

CHAPTER 4

Results and Discussion

This chapter reports and discusses the experimental results from the proposed algorithms which consist of five parts. The first part is the string generation process. The second part is the experiments on standard databases. The third part is the experiments on face recognition and the experimental results from facial expression recognition. We set $m = 2$ in all of our experiments. Finally, the discussion on advantages and disadvantages of the proposed algorithms is presented. The proposed algorithms are applied to standard datasets. The performances are compared to the previous classification algorithms and evaluated by indirect comparison with the existing algorithms.

4.1 String generation process.

For one image, we generated a string consisting of 5 steps as follows:

1. We resized each image in the datasets to 200×200 . If the image is a color image, it will be converted into a gray-scale image using colormap to grayscale [27]. The `rgb2gray` converts RGB values to grayscale values by forming a weighted sum of the R, G, and B components as

$$0.2989 * R + 0.5870 * G + 0.1140 * B \quad (4.1)$$

2. The difference between each image (Ori_f_i) in the dataset and the average of all training images in the dataset (Ave_f) is calculated as

$$Dif_f_i = Ori_f_i - Ave_f \quad \text{for } i = 1, \dots, N \quad (4.2)$$

where N is the total number of images in the dataset. Figures 4.1 and 4.2 show example images from the training dataset and its Ave_f , whereas the corresponding Dif_f_i of each image in figure 4.1 is shown in figure 4.3.

3. The Ori_{f_i} is convolving with the Gaussian kernel with $\sigma=1$ to provide the i^{th} blurred image ($Blur_{f_i}$). Then Dif_{f_i} is divided by $Blur_{f_i}$ (resulting in Fi_{f_i}) to reduce the effect of the variation of illumination [37] as shown in figure 4.4.
4. A symbol of each nonoverlapping subimage of Fi_{f_i} is created and concatenated into a string. For example, if we divide Fi_{f_i} into 100 subimages with the size of 20×20 , then a string of this image will have 100 symbols.
5. The orientation of each pixel (x,y) in the r^{th} subimage according to the gradient direction is calculated as

$$\theta_r(x,y) = 360 - \tan^{-1} \left(\frac{Fi_{f_{ir}}(x,y+1) - Fi_{f_{ir}}(x,y-1)}{Fi_{f_{ir}}(x+1,y) - Fi_{f_{ir}}(x-1,y)} \right) \quad (4.3)$$



Figure 4.1 Examples of original images in the training dataset.

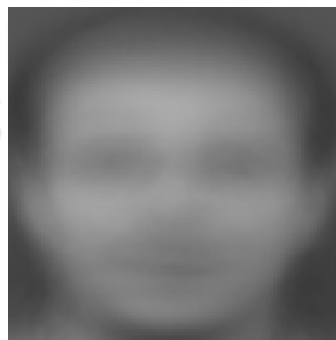


Figure 4.2 Avg_f from the training dataset.



Figure 4.3 Dif_f_i between original images in figure 4.1 and Ave_f in figure 4.2

We reduce the effect of the variation of illumination using the self-quotient normalization [34] by dividing the Dif_f_i with the blurred version of the i^{th} image ($Blur_f_i$). The blurred image of the i^{th} person or ($Blur_f_i$) is created by the Ori_f_i where it is convolved with the Gaussian kernel with $\sigma = 1$. Then we get the final image, (Fi_fi). An example of this process is shown in figure 4.4.

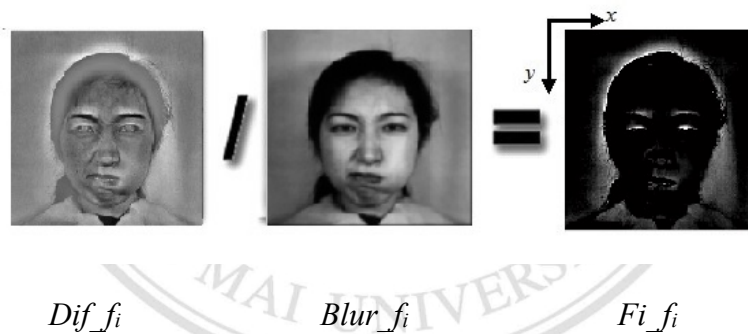


Figure 4.4 An example of self-quotient normalization

ลิขสิทธิ์มหาวิทยาลัยเชียงใหม่
 Copyright© by Chiang Mai University
 All rights reserved

Table 4.1 Bin orientation

Orientation	Bin No.
$0 \leq \theta(x,y) < 45$	1
$45 \leq \theta(x,y) < 90$	2
$90 \leq \theta(x,y) < 135$	3
$135 \leq \theta(x,y) < 180$	4
$180 \leq \theta(x,y) < 225$	5
$225 \leq \theta(x,y) < 270$	6
$270 \leq \theta(x,y) < 315$	7
$315 \leq \theta(x,y) < 360$	8

The string generation process consists of first dividing F_i into nonoverlapping subimages, each has a size of 20×20 . Hence, we have 100 subimages in the most of experiments and 1600 in some experiments which we have the description on each experiment. Then, the Histogram of Gradients (HoG) with 8 bins is implemented in each subimage [35 - 37]. The orientation in each bin is shown in table 4.1.

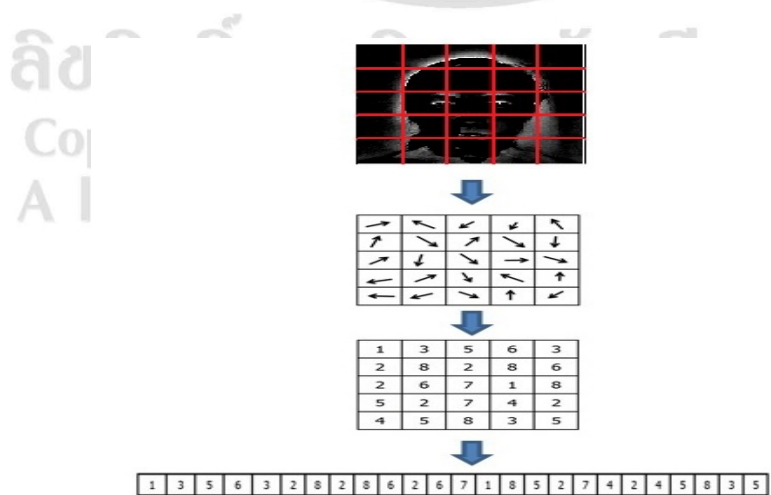


Figure 4.5 String generation process.

The Histogram of Gradients (HoG) [35 – 37] with 8 bins as shown in table 4.1 is implemented in each subimage. The bin with maximum frequency will be a representative of that subimage. The bin number was utilized as a character in the string representing the image, e.g., within the 20th subimage, the bin number 2 has the maximum frequency, and then the character of that subimage will be 2 as well. This step is repeated for all subimages to produce a string of that image. The time complexity of string generation process is approximately $O(N^2)$. An example of this step is shown in figure 4.5.

The leave-one-out cross validation (LOOCV) was implemented and we used the same setting for all of the datasets.

4.2 Standard dataset experiment

In standard dataset experiment, we used 3 datasets consist of Kimia-216, Image Hjpg and USPS datasets. The description these datasets are described as follows:

Figure 4.6 shows the example of each class in Kimia-216 Database (https://www.researchgate.net/figure/260215391_fig5_Fig-10-Kimia-216-database).

This dataset has some objects and some animal images represented as black and white synthetic images and it consists of 216 shapes, grouped into 18 classes with 12 shapes in each class.

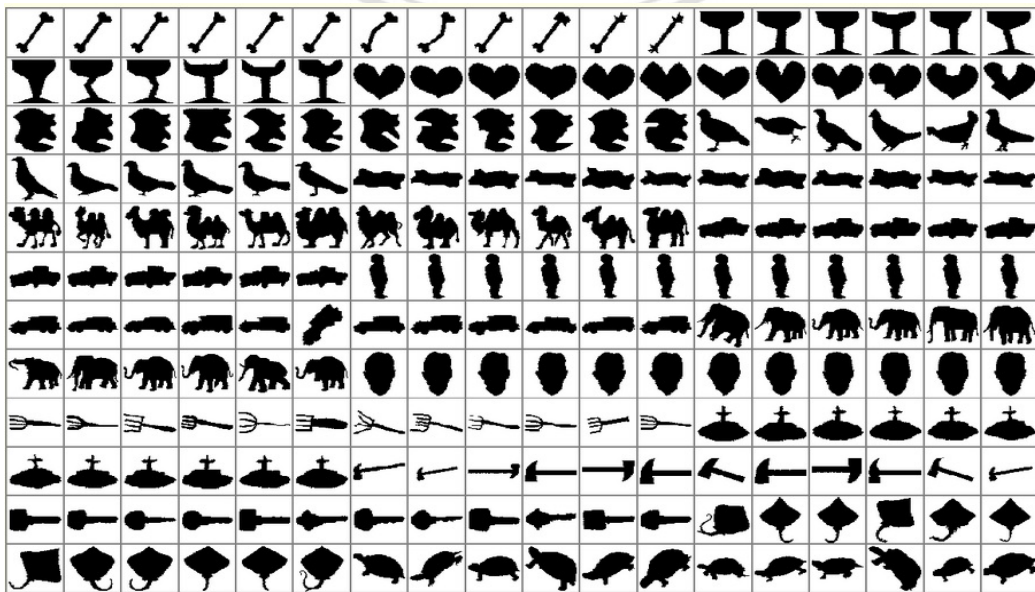


Figure 4.6 Example of each class in Kimia-216 Database.

Figure 4.7 shows the example of each class in Image Hjpg Database (http://www-cvr.ai.uiuc.edu/ponce_grp/data/objects/imagesHjpg/). This dataset has some real objects and real animal images which consist of 16 images for each of 8 objects.

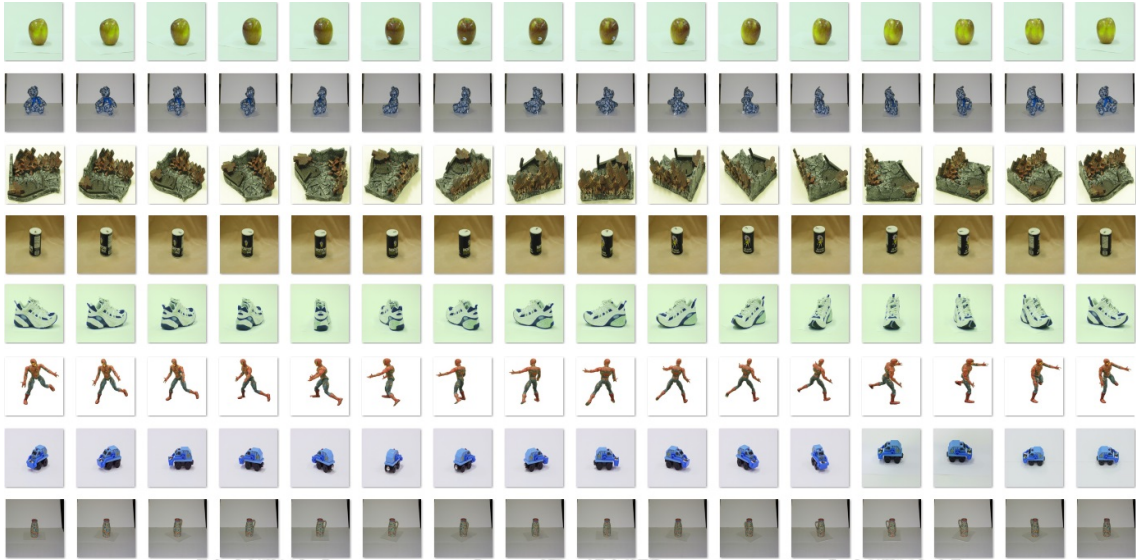


Figure 4.7 Example of each class in Image Hjpg Database.

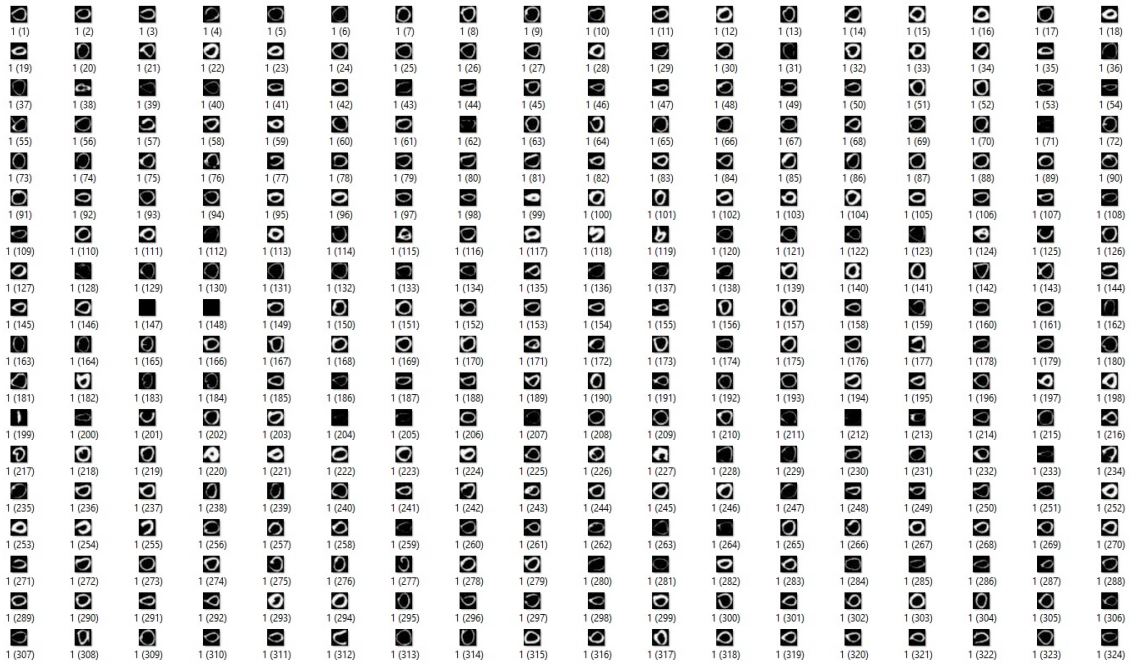


Figure 4.8 Example of class zero in USPS Dataset.

Figure 4.8 shows the example class zero in USPS dataset (<http://www-stat.stanford.edu/~tibs/ElemStatLearn/data.html>). The USPS dataset refers to numeric data obtained from the scanning of handwritten digits from envelopes by the U.S. Postal Service. The original scanned digits have different sizes and orientations; the images here have been normalized in size, resulting in 16 x 16 grayscale images. There are 2007 test observations, consisting of 359 observations for class 0, 264 observations for class 1, 198 observations for class 2, 166 observations for class 3, 200 observations for class 4, 160 observations for class 5, 170 observations for class 6, 147 observations for class 7, 166 observations for class 8, and 177 observations for class 9. We used the same method for string generation process, except for the size of each image which was re sized to 20×20 which using manual resized. From tables 4.2 to 4.8, we show the experimental results when using crisp initialization from sgFKNN1 to sgFKNN3, and sgFKNN7 on each standard dataset when using the K values from 1 to 10 and the minimum number of images for each object minus 1. Nevertheless, we show the experimental results when using crisp initialization from sgFKNN4, sgFKNN5, and sgFKNN6 on each standard dataset when using the K values from 1 to 10 and C and we have N/A in some results because the dataset has number of classes less than number of K .

Table 4.2 The experimental results from sgFKNN1 for standard datasets when using crisp initialization.

Dataset	$K=1$	$K=2$	$K=3$	$K=4$	$K=5$	$K=6$	$K=7$	$K=8$	$K=9$	$K=10$	$K=\min(\text{nperclass})-1$
Kimia-216	100%	100%	100%	100%	98.72%	96.50%	98.72%	96.50%	98.72%	96.50%	98.72%
Image Hjpg	96.87%	96.87%	96.87%	95.31%	96.87%	95.31%	96.87%	95.31%	96.87%	95.31%	96.87%
USPS	88.98%	88.62%	89.60%	88.83%	91.58%	90.85%	91.58%	90.85%	91.58%	90.85%	89.60%

Table 4.3 The experimental results from sgFKNN2 for standard datasets when using crisp initialization.

Dataset	K=1	K=2	K=3	K=4	K=5	K=6	K=7	K=8	K=9	K=10	K=min((nperclass)-1)
Kimia-216	100%	100%	100%	100%	96.50%	94.33%	98.72%	94.33%	98.72%	94.33%	98.72%
Image Hjpg	96.87%	96.87%	96.09%	95.31%	96.09%	95.31%	96.87%	95.31%	96.87%	95.31%	96.87%
USPS	84.63%	84.29%	85.22%	84.49%	87.13%	86.41%	87.13%	86.41%	87.13%	86.41%	85.22%

Table 4.4 The experimental results from sgFKNN3 for standard datasets when using crisp initialization.

Dataset	K=1	K=2	K=3	K=4	K=5	K=6	K=7	K=8	K=9	K=10	K=min((nperclass)-1)
Kimia-216	100%	100%	100%	100%	96.50%	94.33%	98.72%	94.33%	98.72%	94.33%	98.72%
Image Hjpg	96.87%	96.87%	96.09%	95.31%	96.09%	95.31%	96.87%	95.31%	96.87%	95.31%	96.87%
USPS	83.38%	83.04%	83.96%	83.24%	85.85%	85.13%	85.85%	85.13%	85.85%	85.13%	83.96%

Table 4.5 The experimental results from sgFKNN4 for standard datasets when using crisp initialization.

Dataset	K=1	K=2	K=3	K=4	K=5	K=6	K=7	K=8	K=9	K=10	K=C
Kimia-216	98.72%	98.72%	98.72%	98.72%	94.70%	92.53%	96.92%	81.52%	96.92%	81.52%	96.92%
Image Hjpg	96.87%	92.97%	96.09%	93.75%	96.09%	93.75%	96.09%	N/A	N/A	N/A	96.09%
USPS	77.36%	77.05%	77.90%	77.23%	79.65%	78.98%	79.65%	78.98%	79.65%	78.98%	78.98%

Note. We have N/A in some results because the dataset has number of classes less than number of K .

Table 4.6 The experimental results from sgFKNN5 for standard datasets when using crisp initialization.

Dataset	K=1	K=2	K=3	K=4	K=5	K=6	K=7	K=8	K=9	K=10	K=C
Kimia-216	100%	100%	100%	100%	96.50%	94.33%	98.72%	82%	98.72%	82%	98.72%
Image Hjpg	96.87%	92.97%	96.09%	93.75%	96.09%	95.31%	96.09%	N/A	N/A	N/A	96.09%
USPS	82.15%	81.82%	82.72%	82.02%	84.58%	83.87%	84.58%	83.87%	84.58%	83.87%	83.87%

Note. We have N/A in some results because the dataset has number of classes less than number of K .

Table 4.7 The experimental results from sgFKNN6 for standard datasets when using crisp initialization.

Dataset	K=1	K=2	K=3	K=4	K=5	K=6	K=7	K=8	K=9	K=10	K=C
Kimia-216	100%	100%	100%	96%	96.50%	94.33%	98.72%	82%	98.72%	82%	98.72%
Image Hjpg	96.87%	91.41%	94.53%	91.41%	94.53%	91.41%	91.41%	N/A	N/A	N/A	91.41%
USPS	81.08%	80.75%	81.64%	80.95%	83.48%	82.78%	83.48%	82.78%	83.48%	82.78%	82.78%

Note. We have N/A in some results because the dataset has number of classes less than number of K .

Table 4.8 The experimental results from sgFKNN7 for standard datasets when using crisp initialization.

Dataset	K=1	K=2	K=3	K=4	K=5	K=6	K=7	K=8	K=9	K=10	$K=\min(\text{npclass})-1$
Kimia-216	100%	100%	100%	100%	96.50%	94.33%	98.72%	96.50%	98.72%	96.50%	98.72%
Image Hjpg	96.87%	96.87%	96.09%	95.31%	96.09%	95.31%	96.87%	95.31%	96.87%	95.31%	96.87%
USPS	83.38%	83.04%	83.96%	83.24%	85.85%	85.13%	85.85%	85.13%	85.85%	85.13%	83.96%

From the result of the experiment on sgFKNN1 to sgFKNN7, we found that the accuracy rate in sgFKNN1 is higher than the other algorithms, sgFKNN2, sgFKNN7 are

the second and third highest, respectively and sgFKNN4 is the worst accuracy rate than the other algorithm.

Nevertheless, we also show the membership values in appendix A.1

The experimental results from the proposed algorithms on the standard datasets are shown in table 4.9.

Table 4.9 The maximum accuracy rates for the standard datasets on 7 sgFKNNs.

Dataset	sgFKNN1	sgFKNN2	sgFKNN3	sgFKNN4	sgFKNN5	sgFKNN6	sgFKNN7
Kimia-216 [38]	100%	100%	100%	98.72%	100%	100%	100%
Image Hjpg	96.87%	96.87%	96.87%	96.87%	96.87%	96.87%	96.87%
USPS	91.58%	87.13%	85.85%	79.65%	84.58%	83.48%	85.85%

From the result of the experiments on sgFKNN1 to sgFKNN7, we found that the accuracy rate of sgFKNN1 is higher than the other algorithms, but the sgFKNN4 is a little bit lower than the others.

However, the sgFKNN4 on the USPS dataset is lower than the others because the USPS dataset refers to numeric data obtained from the scanning of handwritten digits. The original scanning digits have different sizes and orientations which the properties of sgFKNN4 may not be well as it should be. For instance, the sgFKNN4 has a prototype in each class which the testing sample may be similar in the other classes, because of the similarity of sizes and orientations. We show the misclassification on the USPS dataset when using sgFKNN4 as following below:

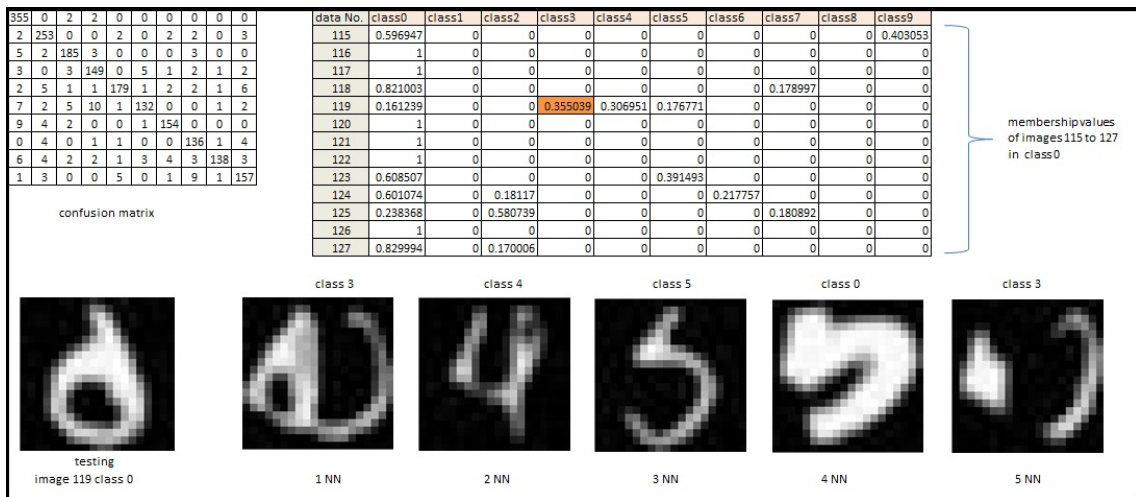


Figure 4.9 The misclassification in data 119th for sgFKNN4 algorithm in USPS database.

From table 4.10 to 4.16, we show the experimental results when using fuzzy initialization from equation 2.7 with sgFKNN1, sgFKNN2, sgFKNN3, and sgFKNN7 on each standard datasets when using the K values from 1 to 10 and the minimum number of images for each object minus 1.

Next, we show the experimental results when using fuzzy initialization from equation 2.7 with sgFKNN4, sgFKNN5, and sgFKNN6 on each standard dataset when using the K values from 1 to 10 and C .

Table 4.10 The experimental results from sgFKNN1 when using fuzzy initialization.

Dataset	$K=1$	$K=2$	$K=3$	$K=4$	$K=5$	$K=6$	$K=7$	$K=8$	$K=9$	$K=10$	$K=\min(\text{upperclass})-1$
Kimia-216	100%	100%	100%	100%	98.72%	96.50%	98.72%	96.50%	96.12%	93.96%	97.40%
Image Hjpg	96.87%	95.31%	96.87%	95.31%	96.87%	95.31%	96.87%	95.31%	94.32%	92.80%	95.58%
USPS	88.98%	87.19%	89.60%	87.40%	91.58%	89.39%	91.58%	89.39%	89.17%	87.03%	88.41%

Table 4.11 The experimental results from sgFKNN2 when using fuzzy initialization.

Dataset	K=1	K=2	K=3	K=4	K=5	K=6	K=7	K=8	K=9	K=10	K=min(nperclass)-1
Kimia-216	100%	99.17%	100%	98.61%	93.89%	91.75%	97.37%	93.02%	97.37%	90.51%	97.35%
Image Hjpg	96.87%	96.06%	96.09%	93.99%	93.49%	92.70%	95.54%	93.99%	95.54%	91.45%	95.52%
USPS	84.63%	83.59%	85.22%	83.32%	84.78%	84.04%	85.94%	85.21%	85.94%	82.91%	84.04%

Table 4.12 The experimental results from sgFKNN3 when using fuzzy initialization.

Dataset	K=1	K=2	K=3	K=4	K=5	K=6	K=7	K=8	K=9	K=10	K=min(nperclass)-1
Kimia-216	100%	99.17%	97.26%	98.61%	92.59%	91.75%	97.37%	93.02%	97.37%	90.51%	97.35%
Image Hjpg	96.87%	96.87%	96.09%	93.99%	93.49%	92.70%	95.54%	93.99%	95.54%	93.99%	95.52%
USPS	83.38%	83.04%	83.96%	82.08%	83.53%	82.80%	84.67%	83.95%	84.67%	83.95%	82.79%

Table 4.13 The experimental results from sgFKNN4 when using fuzzy initialization.

Dataset	K=1	K=2	K=3	K=4	K=5	K=6	K=7	K=8	K=9	K=10	K=C
Kimia-216	98.72%	97.35%	97.61%	96.50%	93.13%	89.99%	95.59%	80.39%	95.59%	78.22%	95.57%
Image Hjpg	96.87%	91.68%	95.01%	91.64%	94.49%	91.32%	94.49%	N/A	N/A	N/A	94.49%
USPS	77.36%	77.05%	77.90%	76.16%	77.50%	76.82%	78.56%	77.88%	78.56%	77.88%	77.88%

Note. We have N/A in some results because the dataset has number of classes less than number of K .

Table 4.14 The experimental results from sgFKNN5 when using fuzzy initialization.

Dataset	K=1	K=2	K=3	K=4	K=5	K=6	K=7	K=8	K=9	K=10	K=C
Kimia-216	100%	98.61%	98.87%	97.75%	94.90%	91.75%	97.37%	80.86%	97.37%	78.68%	97.35%
Image Hjpg	96.87%	91.68%	95.01%	91.64%	94.49%	93.72%	94.49%	N/A	N/A	N/A	94.49%
USPS	82.15%	81.82%	82.72%	80.88%	82.29%	81.57%	83.42%	82.71%	83.42%	82.71%	82.71%

Note. We have N/A in some results because the dataset has number of classes less than number of K .

Table 4.15 The experimental results from sgFKNN6 when using fuzzy initialization.

Dataset	K=1	K=2	K=3	K=4	K=5	K=6	K=7	K=8	K=9	K=10	K=C
Kimia-216	100%	98.61%	98.87%	93.84%	94.90%	91.75%	97.37%	80.86%	97.37%	78.68%	97.35%
Image Hjpg	96.87%	90.14%	93.46%	89.35%	92.96%	89.04%	89.89%	N/A	N/A	N/A	89.89%
USPS	81.08%	80.75%	81.64%	79.83%	81.22%	80.51%	82.34%	81.63%	82.34%	81.63%	81.63%

Note. We have N/A in some results because the dataset has number of classes less than number of K .

Table 4.16 The experimental results from sgFKNN7 when using fuzzy initialization.

Dataset	K=1	K=2	K=3	K=4	K=5	K=6	K=7	K=8	K=9	K=10	$K=\min(\text{nperclass})-1$
Kimia-216	100%	100%	100%	98.61%	93.89%	91.75%	97.37%	95.16%	97.37%	95.16%	97.35%
Image Hjpg	96.87%	96.87%	96.09%	93.99%	93.49%	92.70%	95.54%	93.99%	95.54%	93.99%	95.52%
USPS	83.38%	83.04%	83.96%	82.08%	83.53%	82.80%	84.67%	83.95%	84.67%	83.95%	82.79%

From the result of the experiment on sgFKNN1 to sgFKNN7, we found that the accuracy rate in sgFKNN1 is higher than the other algorithms.

The experimental results from the proposed algorithms when using fuzzy initialization on the standard datasets are shown in table 4.17.

Table 4.17 The maximum accuracy rates for the standard datasets on 7 sgFKNNs when using fuzzy initialization.

Dataset	sgFKNN1	sgFKNN2	sgFKNN3	sgFKNN4	sgFKNN5	sgFKNN6	sgFKNN7
Kimia-216	100%	100%	100%	98.72%	100%	100%	100%
Image Hjpg	96.87%	96.87%	96.87%	96.87%	96.87%	96.87%	96.87%
USPS	91.58%	85.94%	84.67%	78.56%	83.42%	82.34%	84.67%

Table 4.18 The average of accuracy rates for the standard datasets on 7 sgFKNNs when using crisp initialization.

Dataset	sgFKNN1	sgFKNN2	sgFKNN3	sgFKNN4	sgFKNN5	sgFKNN6	sgFKNN7
Kimia-216	98.58%	97.79%	97.79%	94.17%	95.54%	95.18%	98.18%
Image Hjpg	96.30%	96.16%	96.16%	95.21%	95.60%	92.87%	96.16%
USPS	90.27%	85.86%	84.59%	78.58%	83.45%	82.36%	84.59%

Table 4.19 The average of accuracy rates for the standard datasets on 7 sgFKNNs when using fuzzy initialization.

Dataset	sgFKNN1	sgFKNN2	sgFKNN3	sgFKNN4	sgFKNN5	sgFKNN6	sgFKNN7
Kimia-216	97.99%	96.28%	95.91%	92.60%	93.95%	93.60%	96.97%
Image Hjpg	95.58%	94.66%	94.96%	93.75%	94.13%	91.45%	94.96%
USPS	89.07%	84.51%	83.53%	77.60%	82.40%	81.33%	83.53%

Moreover, when we compare the experimental results between crisp initialization with fuzzy initialization, we found that the accuracy rates in crisp initialization are higher than fuzzy initialization in every dataset which show in figure 4.19.

Figure 4.10 shows the average of accuracy rates of every K when using crisp initialization (a) and fuzzy initialization (b).

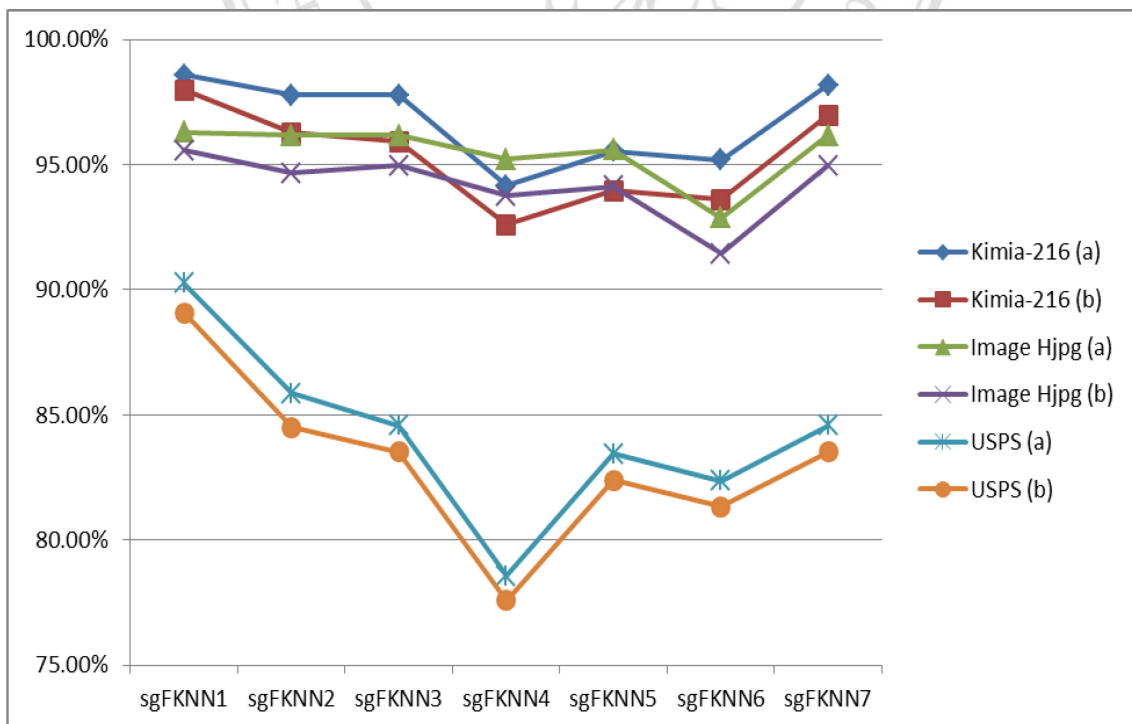


Figure 4.10 The average of accuracy rates of every K when using crisp initialization (a) and fuzzy initialization (b).

Figure 4.11 shows that the membership values when using crisp initialization (a) has higher than fuzzy initialization (b) when correct classification which it makes sure that the testing data stays in the right class. On the other hand, the misclassification in data number 824 on fuzzy initialization shows that the membership values are close to between class 3 and class 4. In crisp initialization, it is also the right classification but in the fuzzy initialization is the misclassification.

Data No	Class	Class 0	Class 1	Class 2	Class 3	Class 4	Class 5	Class 6	Class 7	Class 8	Class 9
1	0	0.97805402	0	0	0	0.02194598	0	0	0	0	0
2	0	0.966767804	0	0	0	0.033232196	0	0	0	0	0
3	0	0.890554309	0	0	0	0.109445691	0	0	0	0	0
4	0	0.969171654	0	0	0	0.030828346	0	0	0	0	0
5	0	0.973957258	0	0	0	0.026042742	0	0	0	0	0
360	1	0.027490457	0.972509543	0	0	0	0	0	0	0	0
361	1	0.010276189	0.989723811	0	0	0	0	0	0	0	0
362	1	0.020171898	0.979828102	0	0	0	0	0	0	0	0
363	1	0.030911021	0.969088979	0	0	0	0	0	0	0	0
364	1	0.014284137	0.985715863	0	0	0	0	0	0	0	0
623	2	0	0	0.973167364	0.026832636	0	0	0	0	0	0
624	2	0	0	0.934187367	0	0	0	0	0	0	0.065812633
625	2	0	0	0.970220528	0	0	0	0	0	0	0.029779472
626	2	0	0	0.973740186	0	0	0	0	0	0	0.026259814
627	2	0	0	0.946450481	0.053549519	0	0	0	0	0	0
821	3	0	0	0.146503033	0.853496967	0	0	0	0	0	0
822	3	0	0	0	0.968570621	0	0	0	0	0.031429379	0
823	3	0.033630147	0	0	0.966369853	0	0	0	0	0	0
824	3	0	0	0	0.526208219	0.473791781	0	0	0	0	0
825	3	0.096070615	0	0	0.903929385	0	0	0	0	0	0

(a)

Data No	Class	Class 0	Class 1	Class 2	Class 3	Class 4	Class 5	Class 6	Class 7	Class 8	Class 9
1	0	0.899423438	0	0	0	0.100576562	0	0	0	0	0
2	0	0.891349007	0	0	0	0.108650993	0	0	0	0	0
3	0	0.831459845	0	0	0	0.168540155	0	0	0	0	0
4	0	0.895362925	0	0	0	0.104637075	0	0	0	0	0
5	0	0.887900406	0	0	0	0.112099594	0	0	0	0	0
360	1	0.137320849	0.862679151	0	0	0	0	0	0	0	0
361	1	0.113185794	0.886814206	0	0	0	0	0	0	0	0
362	1	0.122741014	0.877258986	0	0	0	0	0	0	0	0
363	1	0.135743076	0.864256924	0	0	0	0	0	0	0	0
364	1	0.120635147	0.879364853	0	0	0	0	0	0	0	0
623	2	0	0	0.868529926	0.131470074	0	0	0	0	0	0
624	2	0	0	0.840546697	0	0	0	0	0	0	0.159453303
625	2	0	0	0.864717311	0	0	0	0	0	0	0.135282689
626	2	0	0	0.87106715	0	0	0	0	0	0	0.12893285
627	2	0	0	0.845963633	0.154036367	0	0	0	0	0	0
821	3	0	0	0.188893459	0.811106541	0	0	0	0	0	0
822	3	0	0	0	0.85163486	0	0	0	0	0.14836514	0
823	3	0.151702124	0	0	0.848297876	0	0	0	0	0	0
824	3	0	0	0	0.415023974	0.584976026	0	0	0	0	0
825	3	0.176627242	0	0	0.823372758	0	0	0	0	0	0

(b)

Figure 4.11 The membership values of USPS dataset when using crisp initialization (a) and fuzzy initialization (b).

Additionally, we implemented the multi-prototypes on three algorithms, namely sgFKNN4, sgFKNN5, and sgKNN6, for standard datasets using string grammar fuzzy K-nearest neighbor on the above datasets. We set the number of prototypes on each dataset to about 1%, 5%, 10%, 15%, and 20% of the minimum number of images for each class. The minimum number of images for each class in USPS dataset is 147; we

set the number of prototypes to be 1, 7, 14, 21, and 28. The minimum numbers of images for each class in Kimia-216 and Image Hjpg datasets are 12 and 16, respectively; we set the number of prototypes to be 1, 2, 3, 4, and 5. We used the K values from 1 to 10 and C on each dataset.

Table 4.20 The experimental results from sgFKNN4 for standard datasets when using the different number of prototypes.

Dataset	Number of prototypes	1	2	3	4	5
Kimia-216	$K=1$	98.72%	99.14%	98.84%	96.56%	96.38%
	$K=2$	98.72%	99.14%	98.84%	96.56%	96.38%
	$K=3$	98.72%	99.14%	98.84%	96.56%	96.38%
	$K=4$	98.72%	99.14%	98.84%	96.56%	96.38%
	$K=5$	94.70%	95.10%	94.82%	92.63%	92.46%
	$K=6$	92.53%	92.92%	92.64%	90.51%	90.34%
	$K=7$	96.92%	97.33%	97.04%	94.80%	94.62%
	$K=8$	81.52%	81.87%	81.62%	79.74%	79.59%
	$K=9$	96.92%	97.33%	97.04%	94.80%	94.62%
	$K=10$	81.52%	81.87%	81.62%	79.74%	79.59%
	$K=C$	96.92%	97.33%	97.04%	94.80%	94.62%
Image Hjpg	Number of prototypes	1	2	3	4	5
	$K=1$	96.87%	96.45%	96.03%	95.83%	95.41%
	$K=2$	92.97%	92.57%	92.16%	91.97%	91.57%
	$K=3$	96.09%	95.67%	95.26%	95.06%	94.64%
	$K=4$	93.75%	93.34%	92.94%	92.74%	92.34%
	$K=5$	96.09%	95.67%	95.26%	95.06%	94.64%
	$K=6$	93.75%	93.34%	92.94%	92.74%	92.34%
	$K=7$	96.09%	95.67%	95.26%	95.06%	94.64%
	$K=8$	N/A	N/A	N/A	N/A	N/A
	$K=9$	N/A	N/A	N/A	N/A	N/A
	$K=10$	N/A	N/A	N/A	N/A	N/A
		$K=C$	96.09%	95.67%	95.26%	95.06%
USPS	Number of prototypes	1	7	14	28	32
	$K=1$	77.36%	77.63%	78.02%	77.46%	77.11%
	$K=2$	77.05%	77.32%	77.71%	77.15%	76.80%
	$K=3$	77.90%	78.17%	78.57%	78.00%	77.65%
	$K=4$	77.23%	77.50%	77.89%	77.33%	76.98%
	$K=5$	79.65%	79.93%	80.33%	79.75%	79.39%
	$K=6$	78.98%	79.26%	79.65%	79.08%	78.72%
	$K=7$	79.65%	79.93%	80.33%	79.75%	79.39%
	$K=8$	78.98%	79.26%	79.65%	79.08%	78.72%
	$K=9$	79.65%	79.93%	80.33%	79.75%	79.39%
	$K=10$	78.98%	79.26%	79.65%	79.08%	78.72%
	$K=C$	78.98%	79.26%	79.65%	79.08%	78.72%

Note. We have N/A in some results because the dataset has number of classes less than number of K .

The experiment results from sgFKNN4 when using the different number of prototypes. We found that the best accuracy rate on Kimia-216, Image Hjpg and USPS dataset are 99.14%, 96.87% and 80.33%, respectively, when using $K = 1, 2, 3, 4$ and number of prototype is 2 for Kimia-216 database, $K = 1$ and number of prototypes is 1 for Image Hjpg database, $K = 5, 7, 9$ and number of prototypes is 14 for USPS database. For Image Hjpg has 8 classes then we have $K = 1$ to 7 in our experiment.

Table 4.21 The experimental results from sgFKNN5 for standard datasets when using the different number of prototypes.

Dataset	Number of prototypes	1	2	3	4	5
Kimia-216	$K=1$	100%	100%	99.74%	97.29%	97.02%
	$K=2$	100%	100%	99.74%	97.29%	97.02%
	$K=3$	100%	100%	99.74%	97.29%	97.02%
	$K=4$	96%	96%	95.75%	93.40%	93.14%
	$K=5$	96.50%	96.50%	96.25%	93.88%	93.62%
	$K=6$	94.33%	94.33%	94.08%	91.77%	91.52%
	$K=7$	98.72%	98.72%	98.46%	96.04%	95.78%
	$K=8$	82%	82%	81.79%	79.78%	79.56%
	$K=9$	98.72%	98.72%	98.46%	96.04%	95.78%
	$K=10$	82%	82%	81.79%	79.78%	79.56%
	$K=C$	98.72%	98.72%	98.46%	96.04%	95.78%
Image Hjpg	Number of prototypes	1	2	3	4	5
	$K=1$	96.87%	96.45%	96.03%	95.83%	95.41%
	$K=2$	92.97%	92.57%	92.16%	91.97%	91.57%
	$K=3$	96.09%	95.67%	95.26%	95.06%	94.64%
	$K=4$	93.75%	93.34%	92.94%	92.74%	92.34%
	$K=5$	96.09%	95.67%	95.26%	95.06%	94.64%
	$K=6$	95.31%	94.90%	94.48%	94.29%	93.87%
	$K=7$	96.09%	95.67%	95.26%	95.06%	94.64%
	$K=8$	N/A	N/A	N/A	N/A	N/A
	$K=9$	N/A	N/A	N/A	N/A	N/A
	$K=10$	N/A	N/A	N/A	N/A	N/A
	$K=C$	96.09%	95.67%	95.26%	95.06%	94.64%
USPS	Number of prototypes	1	7	14	28	32
	$K=1$	82.15%	82.44%	82.85%	82.26%	81.88%
	$K=2$	81.82%	82.11%	82.52%	81.93%	81.55%
	$K=3$	82.72%	83.01%	83.42%	82.83%	82.45%
	$K=4$	82.02%	82.31%	82.72%	82.13%	81.75%
	$K=5$	84.58%	84.88%	85.30%	84.69%	84.30%
	$K=6$	83.87%	84.17%	84.58%	83.98%	83.59%
	$K=7$	84.58%	84.88%	85.30%	84.69%	84.30%
	$K=8$	83.87%	84.17%	84.58%	83.98%	83.59%
	$K=9$	84.58%	84.88%	85.30%	84.69%	84.30%
	$K=10$	83.87%	84.17%	84.58%	83.98%	83.59%
	$K=C$	83.87%	84.17%	84.58%	83.98%	83.59%

Note. We have N/A in some results because the dataset has number of classes less than number of K .

The experiment results from sgFKNN5 when using the different number of prototypes. We found that the best accuracy rate on Kimia-216, Image Hjpg and USPS dataset are 100%, 96.87% and 85.30%, respectively, when using $K = 1, 2, 3$ and number of prototypes are 1, 2 for Kimia-216 database, $K = 1$ and number of prototypes is 1 for Image Hjpg database, $K = 5, 7, 9$ and number of prototypes is 14 for USPS database.

Table 4.22 The experimental results from sgFKNN6 for standard datasets when using the different number of prototypes.

Dataset	Number of prototypes	1	2	3	4	5
Kimia-216	$K=1$	100%	100%	99.74%	97.29%	97.02%
	$K=2$	100%	100%	99.74%	97.29%	97.02%
	$K=3$	100%	100%	99.74%	97.29%	97.02%
	$K=4$	96%	96%	95.75%	93.40%	93.14%
	$K=5$	96.50%	96.50%	96.25%	93.88%	93.62%
	$K=6$	94.33%	94.33%	94.08%	91.77%	91.52%
	$K=7$	98.72%	98.72%	98.46%	96.04%	95.78%
	$K=8$	82%	82%	81.79%	79.78%	79.56%
	$K=9$	98.72%	98.72%	98.46%	96.04%	95.78%
	$K=10$	82%	82%	81.79%	79.78%	79.56%
	$K=C$	98.72%	98.72%	98.46%	96.04%	95.78%
Image Hjpg	Number of prototypes	1	2	3	4	5
	$K=1$	96.87%	96.45%	96.03%	95.83%	95.41%
	$K=2$	91.41%	91.01%	90.62%	90.43%	90.03%
	$K=3$	94.53%	94.12%	93.71%	93.52%	93.11%
	$K=4$	91.41%	91.01%	90.62%	90.43%	90.03%
	$K=5$	94.53%	94.12%	93.71%	93.52%	93.11%
	$K=6$	91.41%	91.01%	90.62%	90.43%	90.03%
	$K=7$	91.41%	91.01%	90.62%	90.43%	90.03%
	$K=8$	N/A	N/A	N/A	N/A	N/A
	$K=9$	N/A	N/A	N/A	N/A	N/A
	$K=10$	N/A	N/A	N/A	N/A	N/A
	$K=C$	91.41%	91.01%	90.62%	90.43%	90.03%
USPS	Number of prototypes	1	7	14	28	32
	$K=1$	81.08%	81.36%	81.77%	81.18%	80.82%
	$K=2$	80.75%	81.03%	81.44%	80.85%	80.49%
	$K=3$	81.64%	81.92%	82.33%	81.74%	81.38%
	$K=4$	80.95%	81.23%	81.64%	81.05%	80.69%
	$K=5$	83.48%	83.77%	84.19%	83.58%	83.21%
	$K=6$	82.78%	83.07%	83.48%	82.88%	82.51%
	$K=7$	83.48%	83.77%	84.19%	83.58%	83.21%
	$K=8$	82.78%	83.07%	83.48%	82.88%	82.51%
	$K=9$	83.48%	83.77%	84.19%	83.58%	83.21%
	$K=10$	82.78%	83.07%	83.48%	82.88%	82.51%
		$K=C$	82.78%	83.07%	83.48%	82.88%

Note. We have N/A in some results because the dataset has number of classes less than number of K .

The experiment results from sgFKNN6 when using the different number of prototypes. We found that the best accuracy rate on Kimia-216, Image Hjpg and USPS dataset are 100%, 96.87% and 84.19%, respectively, when using $K = 1, 2, 3$ and number of prototypes are 1, 2 for Kimia-216 database, $K = 1$ and number of prototype is 1 for Image Hjpg database, $K = 5, 7, 9$ and number of prototype is 14 for USPS database.

We found that the optimal number of prototypes which provides the highest accuracy rate should be 5% to 15% of data samples, otherwise the accuracy rate will be dropped because some prototypes may be outliers.

4.3 Face recognition and expression experiment

In face recognition, we used 10 public standard datasets. The descriptions of these datasets are described as following:

1. ORL [39]

This dataset has 10 different images of each 40 distinct subjects. For some subjects, the images were taken at different times, varying the lighting, facial expressions (open / closed eyes, smiling / not smiling) and facial details (glasses / no glasses). All the images were taken against a dark homogeneous background with the subjects in an upright, frontal position (with tolerance for some side movement). The URL of dataset is <http://www.cl.cam.ac.uk/research/dtg/attarchive/facedatabase.html>.



Figure 4.12 Example of 1st class from ORL dataset



Figure 4.13 Example of all 40 classes from ORL dataset

2. FEI [40]

The FEI face database is a Brazilian face database that contains a set of face images taken between June 2005 and March 2006 at the Artificial Intelligence Laboratory of FEI in Sao Bernardo do Campo, Sao Paulo, Brazil. There are 14 images for each of 200 individuals, with the total of 2800 images. All images are colorful and taken against a white homogenous background in an upright frontal position with profile rotation of up to about 180 degrees. Scale might vary about 10% and the original size of each image is 640x480 pixels. All faces are mainly represented by students and staffs at FEI, between 19 and 40 years old with distinct appearance, hairstyle, and adorns. The numbers of male and female subjects are exactly the same and equal to 100 which the URL of dataset is <http://fei.edu.br/~cet/facedatabase.html>.



Figure 4.14 Example of FEI dataset in class 1.

3. Yale [41]

The Yale database contains 165 gray scale images in GIF format from 15 individuals. There are 11 images per subject, one per different facial expression or configuration, i.e., center-light, with glasses, happy, left-light, without glasses, normal, light, sad, sleepy, surprised and wink.



Figure 4.15 Example of Yale dataset

4. JAFFE [42]

Japanese Female Facial Expression (JAFFE) database consists of 213 images of Japanese female facial expressions, where each image corresponds to one of the seven categories of expression, i.e., anger, disgust, fear, happiness, neutral, sadness, and surprise which the URL of dataset is <http://www.kasrl.org/jaffe.html>.

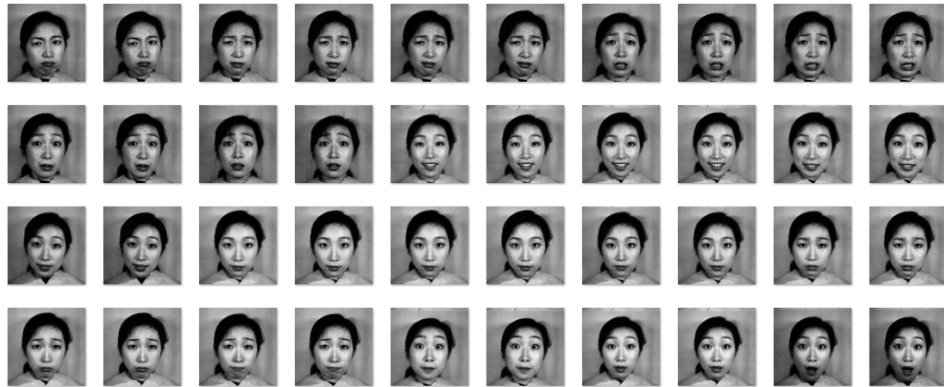


Figure 4.16 Example of JAFFE dataset

5. Pain expressions [43]

This database is the cropped versions, fixed eye location, 7 expressions (not sad) from each of 12 women. The resolution is 181 x 241. There are 26 images for each of individuals which the URL of dataset is http://pics.psych.stir.ac.uk/2D_face_sets.htm.

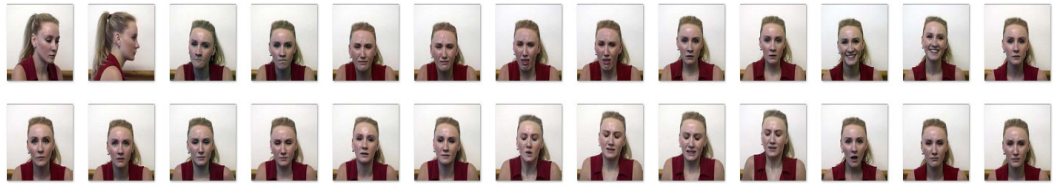


Figure 4.17 Example of Pain expressions dataset

6. Senthilkumar [44]

The database contains 80 color face images of 5 persons (all men), including frontal views of faces with different facial expressions, oclusions, and brightness conditions. Each person has 16 different images which the URL of dataset is <http://www.geocities.ws/senthilirtt/Senthil%20Face%20Database%20Version1>.



Figure 4.18 Example of Senthilkumar dataset

7. PICS - Psychological Image Collection at Stirling [45]

This is a collection of images useful for conducting experiments in psychology, primarily faces, though other submissions are welcome. They are free for research use. The database contains 9 images for each of 36 individuals which the URL of dataset is <http://pics.psych.stir.ac.uk/>.



Figure 4.19 Example of PICS dataset

8. MIT CBCL [46]

There are 200 images for each of 10 individuals (poses and scale variation of facial expressions) which the URL of dataset is <http://cbcl.mit.edu/software-datasets/heisele/facerecognition-database.html>.



Figure 4.20 Example of MIT original dataset

9. CMU AMP [47]

CMU AMP database, there are 13 subjects in the database, each with 75 images, and all of the face images are collected in the same lighting condition, allowing only human expression changes.

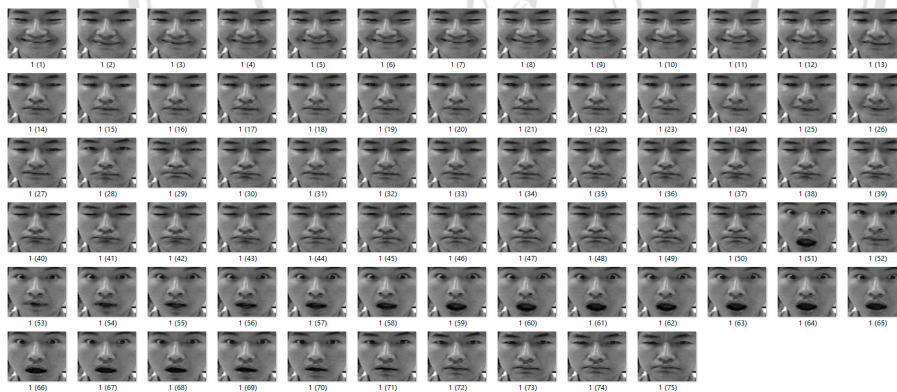


Figure 4.21 Example of CMU AMP dataset

10. Georgia Tech [48]

The database contains images of 50 people and is stored in JPEG format. For each individual, there are 15 color images captured between 06/01/99 and 11/15/99. Most of the images were taken in two different sessions to take into account the variations in illumination conditions, facial expression, and appearance. In addition, the faces were captured at different scales and orientations which the URL of dataset is http://www.anefian.com/research/face_reco.htm.



Figure 4.22 Example of Georgia Tech dataset

4.3.1 Face recognition experiment result

We implement the seven algorithms for face recognition using string grammar fuzzy K-nearest neighbor (sgFKNN1 to sgKNN7) on the ten standard face recognition datasets.

From tables 4.23 to 4.29, we show the experimental results from sgFKNN1, sgFKNN2, sgFKNN3, and sgFKNN7 on 10 face recognition datasets when using the K values from 1 to 10 and the minimum number of images for each person minus 1.

Nevertheless, we show the experimental results from sgFKNN4, sgFKNN5, and sgFKNN6 on 10 face recognition datasets when using the K values from 1 to 10 and C .

Moreover, in the next, we show the experimental results of multi-prototypes from sgFKNN4, sgFKNN5, and sgFKNN6 on 10 face recognition datasets when using the K values from 1 to 10 and C .

Table 4.23 The experimental results from sgFKNN1 for face recognition datasets.

Dataset	k=1	k=2	k=3	k=4	k=5	k=6	k=7	k=8	k=9	k=10	$k=\min(\text{perclass})-1$
ORL	96.70%	96.70%	99.25%	98.65%	99.25%	98.65%	99.25%	98.65%	99.25%	98.65%	99.25%
MIT-CBCL	99.70%	99.70%	100%	99.70%	100%	99.70%	100%	99.70%	100%	99.70%	99.70%
Georgia Tech	77.37%	77.37%	79.57%	79.33%	79.57%	79.33%	79.57%	79.33%	79.57%	79.33%	79.31%
FEI	91.73%	91.73%	93.85%	93.57%	93.85%	93.57%	93.85%	93.57%	93.85%	93.57%	93.83%
JAFFE	100%	100%	100%	100%	100%	100%	100%	100%	100%	100%	98.49%
Pain Expression	89.16%	89.16%	90%	89.73%	90%	89.73%	90%	89.73%	90%	89.73%	90%
Senthil Kumar	96.25%	96.25%	93.75%	93.75%	95%	93.75%	93.75%	93.75%	95%	95%	93.75%
PICS	94.27%	94.27%	95.23%	94.94%	95.23%	94.94%	95.23%	94.94%	95.23%	94.94%	95.23%
Yale	96.34%	96.34%	94.85%	94.04%	96.98%	96.17%	96.98%	96.17%	96.98%	96.17%	94.85%
CMU AMP	98.67%	98.67%	100%	99.70%	100%	99.70%	100%	99.70%	100%	99.70%	100%

From the experimental results, the best accuracy rate is sgFKNN1 for face recognition datasets. We found that the best accuracy rate on ORL, Pain Expression, PICS and CMU AMP database are 99.25%, 90%, 95.23% and 100%, respectively. For MIT-CBCL, Georgia Tech and FEI databases the accuracy rates are 100%, 79.57%, and 93.85%, respectively.

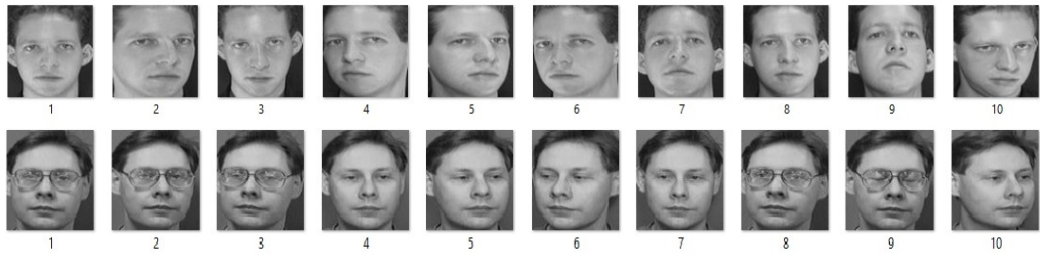


Figure 4.23 The example of 10 images for the two classes in ORL dataset.

Figure 4.23 shows 10 examples images for two classes in ORL dataset. We found that 10 images for each class which are scale invariance, rotational invariance, left and right turning faces, up and down faces in the 1st class, with and without glasses, left and right turning faces in the 2nd class have high between-classes variation. Thus, large K values may not be appropriated.

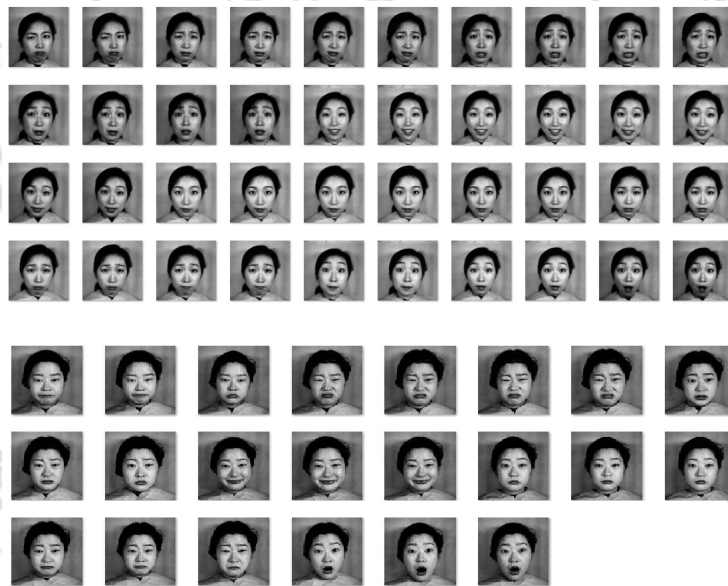


Figure 4.24 The example of the images for each class in JAFFE dataset.

Figure 4.24 shows data of 2 classes which have high between-classes variation. These 2 classes have differences in image size and facial expression. Thus, we expect that the accuracy rate should be high.



Figure 4.25 The example of the images for each class in Yale dataset.

Figure 4.25 shows three different classes in three rows on the Yale database which has a high within-classes variation, such as different emotional faces and lighting. However, it has low distribution of between class, such as containing the same smiling faces, the same lighting direction, and the same backgrounds. Thus, the best of K values are 5, 7, and 9.



Figure 4.26 The example of the images for each class in Senthilkumar dataset.

Figure 4.26 shows the Senthilkumar database has a high within-classes variation where it has different poses in each image for each class. However, it has low between-classes variation. Considering that the 1st row represents the 1st class and the 2nd row represents the 2nd class, we can see that each class has a similar look (low between-classes variation). Thus, the $K=1$ may be appropriated.

		Person No.																																											
		1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31	32	33	34	35	36	37	38	39	40				
Person 28th (Picture No.)	28.1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0.01	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0		
	28.2	0	0	0	0	0	0	0	0	0	0	0	0	0	0.1	0	0	0	0	0	0	0	0	0	0	0	0	0	0.8	0	0	0	0	0	0	0	0	0.1	0	0	0	0	0		
	28.3	0	0.1	0	0	0	0	0	0	0	0	0	0	0	0.3	0	0	0	0	0	0	0	0	0	0	0	0	0.6	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
	28.4	0	0	0	0	0	0	0	0	0	0	0.2	0	0	0.1	0	0	0	0	0.31	0	0.1	0	0	0	0	0	0.2	0	0	0	0	0	0	0	0	0	0	0.1	0	0	0	0	0	
	28.5	0	0	0	0	0	0	0	0	0	0	0	0	0	0.1	0	0	0	0	0.06	0	0	0	0	0	0	0	0	0.7	0	0	0	0	0	0	0	0	0.1	0	0	0	0	0		
	28.6	0	0	0	0	0	0	0	0	0	0	0	0.01	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	
	28.7	0	0	0	0	0	0	0	0	0	0	0	0	0	0.1	0	0	0	0	0.12	0	0	0	0	0	0	0	0.5	0	0	0	0	0.1	0	0	0.2	0	0	0	0	0	0	0	0	
	28.8	0	0	0.2	0.1	0	0	0	0	0	0	0	0.22	0	0.1	0	0.1	0	0.1	0	0	0	0	0	0	0	0	0.2	0	0	0	0	0	0	0	0	0.1	0.1	0	0	0	0	0	0	0
	28.9	0	0	0	0	0	0	0	0	0	0	0	0.1	0	0.2	0	0	0	0	0	0	0.2	0	0	0	0	0	0	0.4	0	0	0	0	0.1	0.1	0	0	0	0	0	0	0	0	0	0
	28.10	0	0	0	0	0	0	0	0	0	0	0	0	0.05	0	0	0	0	0	0.14	0	0	0	0	0	0	0	0	0.7	0	0	0	0	0	0	0	0	0	0	0	0.1	0	0	0	0

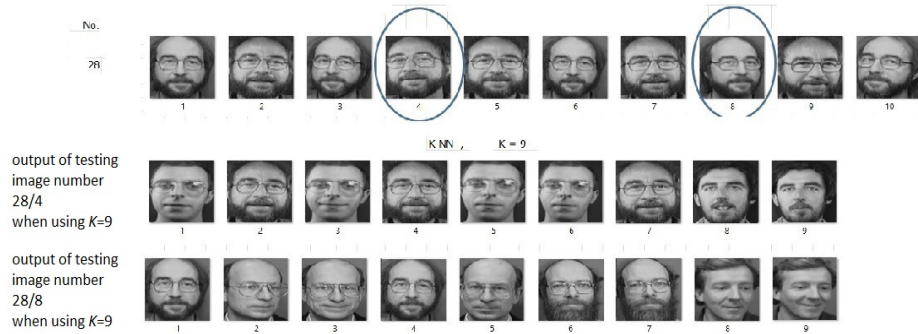


Figure 4.27 The misclassification in person 28th for sgFKNN1 algorithm in ORL database.

		Person No.																																															
		1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31	32	33	34	35	36	37	38	39	40								
Person 16th (Picture No.)	16.1	0	0	0	0	0	0	0	0	0	0	0	0	0	0.1	0.1	0.5	0	0.1	0	0	0	0	0.1	0	0	0	0	0	0	0	0	0	0	0.1	0	0	0	0	0	0	0	0	0					
	16.2	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0			
	16.3	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0.9	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0		
	16.4	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0.9	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0		
	16.5	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0.9	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0		
	16.6	0	0.2	0	0	0	0	0	0	0	0	0	0	0	0	0.3	0	0	0.4	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0.1	0	0	0	0	0	0	0	0	0	0	0			
	16.7	0.2	0.2	0	0	0	0	0	0	0	0	0	0	0	0	0.5	0	0	0	0	0	0	0	0	0	0	0	0.1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	
	16.8	0.1	0	0	0	0	0	0	0	0	0	0	0	0	0.1	0	0.6	0	0.1	0	0	0	0	0	0	0	0	0.1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	
	16.9	0.2	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0.6	0	0	0	0	0	0	0	0	0	0.1	0	0	0	0	0	0	0	0.1	0	0	0	0	0	0	0.1	0	0	0	0	0		
	16.10	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0.7	0	0	0.1	0	0	0	0	0	0.1	0	0	0.1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

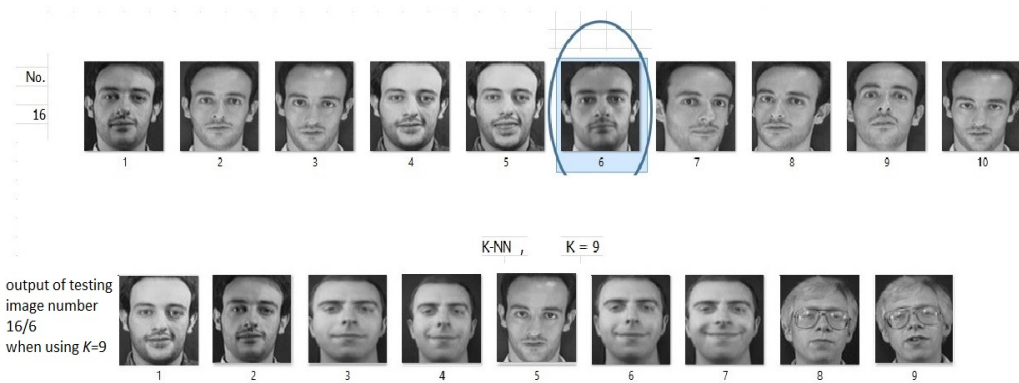


Figure 4.28 The misclassification in person 16th for sgFKNN1 algorithm in ORL database.

Person No.	Person No.																																							
	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31	32	33	34	35	36	37	38	39	40
36.1	0	0	0	0	0	0.1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0.82	0	0	0	0
36.2	0	0	0	0	0	0.1	0	0	0	0	0	0	0	0	0	0.1	0	0	0	0	0	0.1	0	0	0	0	0	0	0	0	0	0	0	0	0	0.7	0	0	0	0
36.3	0	0	0	0	0	0.2	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0.79	0	0	0	0	
36.4	0	0	0	0	0	0.4	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0.59	0	0	0	0		
36.5	0	0	0	0	0	0.1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0.85	0	0	0	0		
36.6	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0.83	0	0	0	0		
36.7	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0.89	0	0	0	0		
36.8	0	0	0	0	0	0.1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0.85	0	0	0	0		
36.9	0.1	0.2	0	0	0	0	0	0	0	0	0	0	0	0	0	0.1	0	0	0.3	0.1	0	0	0	0	0	0	0	0	0	0	0	0	0	0.22	0	0	0	0		
36.10	0	0	0	0	0	0	0	0	0	0.1	0.1	0	0	0	0.1	0.1	0	0.1	0	0	0	0	0	0	0	0	0	0	0	0	0	0.17	0.1	0	0	0.22	0	0	0	0

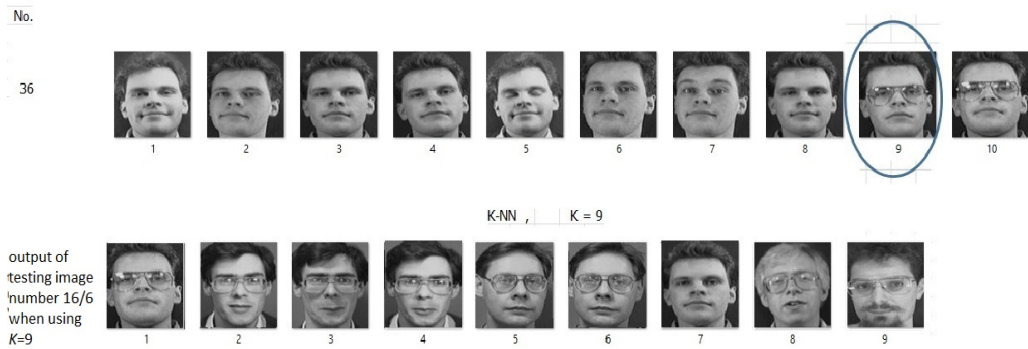


Figure 4.29 The misclassification in person 36th for sgFKNN1 algorithm in ORL database.

The misclassifications from sgFKNN1 algorithm are shown in figures 4.27-4.29. Figure 4.29 shows the image number 9 of the 36th person wearing glasses when using $K = 9$. It found that the testing image was similar to another group who wearing glasses. Figure 4.30 shows the example of Georgia Tech dataset. The misclassification could be another reason that a small distance was created between strings which usually are far from each other, by the process of string transformation, when calculating the Levenshtein distance.

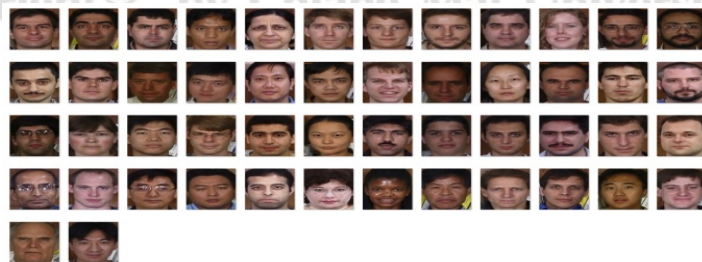


Figure 4.30 Examples from Georgia Tech dataset

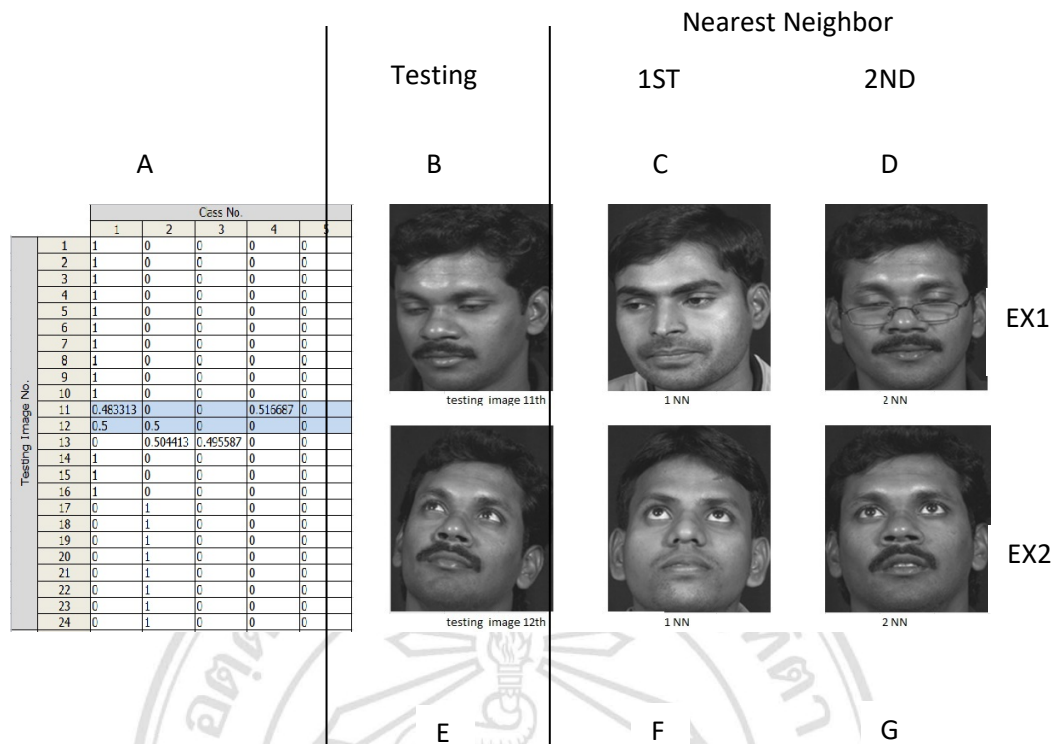


Figure 4.31 The misclassification in person 11th and 12th for sgFKNN1 algorithm in Senthilkumar database when using $K=2$.

The misclassifications from sgFKNN1 algorithm are shown in figure 4.31. For person 11th (B), the membership values in class 1 is 0.48 and in class 4 is 0.52. It is found that the image in 1st-NN (C) has similarity in pose and the image in 2nd-NN (D) has similarity in the person. Nevertheless, the distance between test image with 1st-NN is 976 and with 2nd-NN is 979. These values are considered rather close which the membership values are rather close too.

Moreover, for the 12th testing image (E), the membership values in class 1 and class 2 are 0.5. It is shown that if we use $K=2$ or even values, then the membership values have an equal opportunity and the accuracy rate will be dropped.

Table 4.24 The experimental results from sgFKNN2 for face recognition datasets.

Dataset	K=1	K=2	K=3	K=4	K=5	K=6	K=7	K=8	K=9	K=10	K=min(n(perclass)-1)
ORL	94.74%	94.74%	97.24%	95.65%	97.24%	95.65%	97.24%	95.65%	96.00%	95.65%	96%
MIT-CBCL	99.45%	99.45%	99.75%	99.45%	99.75%	99.45%	99.75%	99.45%	99.75%	99.45%	98.05%
Georgia Tech	73.51%	73.51%	75.60%	75.37%	75.60%	75.37%	75.60%	75.37%	75.60%	75.37%	73.16%
FEI	88.80%	88.80%	90.85%	90.58%	90.85%	90.58%	90.85%	90.58%	90.85%	90.58%	89.26%
JAFFE	100%	100%	99.53%	98.59%	100%	99.06%	99.53%	99.53%	99.53%	99.53%	97.48%
Pain Expression	76.99%	76.99%	77.72%	77.49%	77.72%	77.49%	77.72%	77.49%	77.72%	77.49%	77.72%
Senthikumar	96.25%	90%	91.25%	90.00%	92.50%	91.25%	91.25%	90%	91.25%	92.50%	90%
PICS	87.09%	87.09%	87.98%	87.72%	87.98%	87.72%	87.98%	87.72%	87.98%	87.72%	87.98%
Yale	85.26%	85.26%	83.94%	83.13%	85.73%	85.01%	85.73%	85.01%	85.73%	85.01%	83.94%
CMU AMP	96.84%	96.84%	98.15%	97.85%	98.15%	97.85%	98.15%	97.85%	98.15%	97.85%	98.15%

From the experimental results, we found that the best accuracy rate on ORL is 97.24% when using $K = 3, 5,$ and 7 . For Pain Expression, PICS and CMU AMP database are 77.72%, 87.98% and 98.15%, respectively. For MIT-CBCL, Georgia Tech and FEI databases the accuracy rates are 99.75%, 75.60%, and 90.85%, respectively.

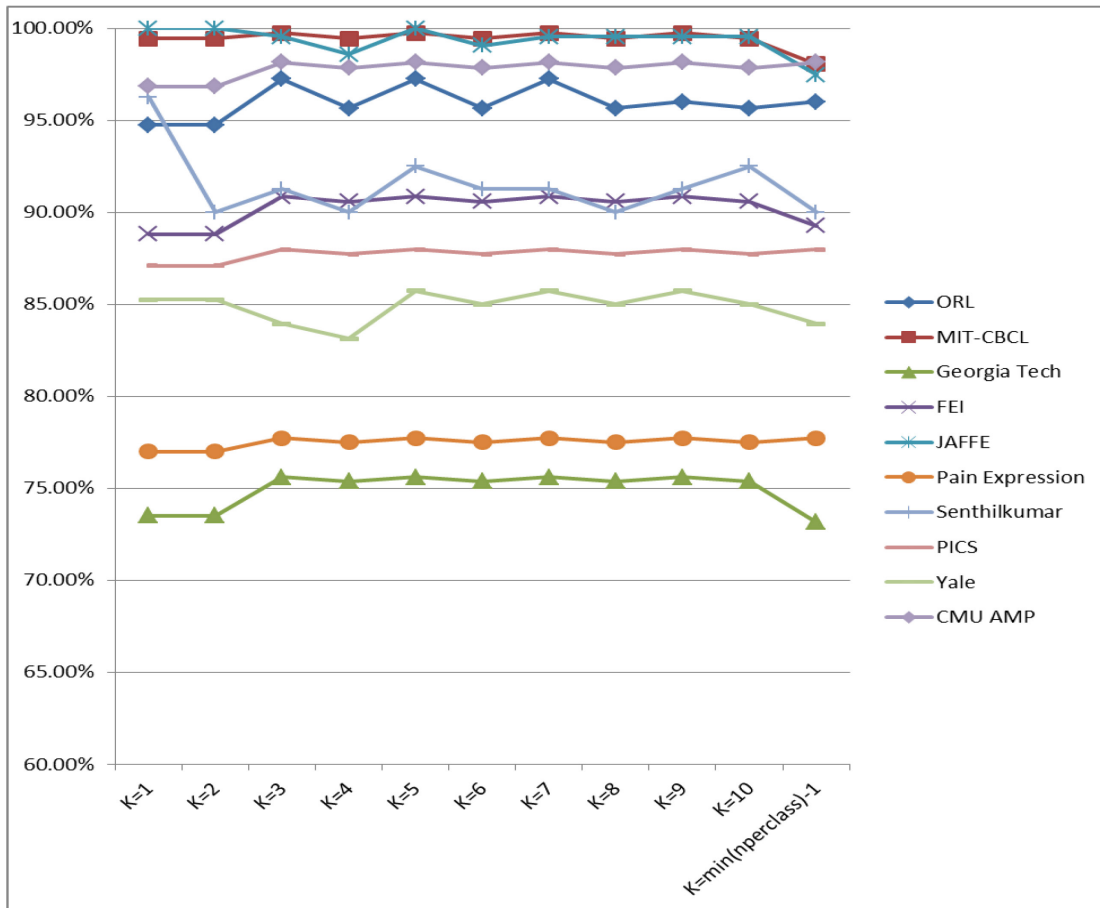


Figure 4.32 The accuracy rate of sgFKNN2 on 10 standard datasets

In addition, the experimental results show that the accuracy rate is similar to sgFKNN1 and has similar pattern when varying K values but a little bit less than those of sgFKNN1.

For each image in the classes of every dataset, if we generate string in the same process then we have the same string in same image on 7 sgFKNNs and same distance in sgFKNN1, sgFKNN2, sgFKNN3 and sgFKNN7 as group 1 and same distance in sgFKNN4, sgFKNN5 and sgFKNN6 as group 2. From this reason, the misclassifications depend on each equation in 7 sgFKNNs. The misclassifications from sgFKNN2 algorithm are similar to sgFKNN1. Although we use the different datasets which also make the accuracy rate changed. However, using the same dataset will have the same pattern of accuracy rate, although different algorithms which variable on K

values were used. However, if we use $K=2$ or even values, then the membership values will have an equal opportunity which make the accuracy rate dropped.

Table 4.25 The experimental results from sgFKNN3 for face recognition datasets.

Dataset	$K=1$	$K=2$	$K=3$	$K=4$	$K=5$	$K=6$	$K=7$	$K=8$	$K=9$	$K=10$	$K=\min(\text{nperclass})-1$
ORL	96.46%	96.46%	99.00%	96.81%	97.74%	96.81%	97.74%	96.81%	97.74%	96.81%	97.74%
MIT-CBCL	99.45%	99.45%	99.75%	99.45%	99.75%	99.45%	99.75%	99.45%	99.75%	99.45%	98.05%
Georgia Tech	73.51%	73.51%	75.60%	75.37%	75.60%	75.37%	75.60%	75.37%	75.60%	75.37%	73.16%
FEI	84.34%	84.34%	86.29%	86.03%	86.29%	86.03%	86.29%	86.03%	86.29%	86.03%	84.78%
JAFFE	100%	100%	99.53%	98.59%	99.53%	99.06%	99.53%	99.53%	99.53%	99.53%	97.48%
Pain Expression	81.97%	81.97%	82.75%	82.50%	82.75%	82.50%	82.75%	82.50%	82.75%	82.50%	82.75%
Senthikumar	96.25%	90%	91.25%	90.00%	92.50%	91.25%	91.25%	90%	91.25%	92.50%	90%
PICS	82.57%	82.57%	83.41%	83.16%	83.41%	83.16%	83.41%	83.16%	83.41%	83.16%	83.41%
Yale	80.98%	80.98%	79.73%	78.92%	82.42%	81.61%	82.42%	81.61%	82.42%	81.61%	79.73%
CMU AMP	84.98%	84.98%	86.13%	85.87%	86.13%	85.87%	86.13%	85.87%	86.13%	85.87%	86.13%

From the experimental results, we found that the best accuracy rate on ORL is 99% when using $K = 3$. For Pain Expression, PICS and CMU AMP database the accuracy rates are 82.75%, 83.41% and 86.13%, respectively, when using $K = 3, 5, 7, 9$ and $\min(\text{nperclass})-1$. For MIT-CBCL, Georgia Tech and FEI databases the accuracy rates are 99.75%, 75.60%, and 86.29%, respectively, when using $K = 3, 5, 7, 9$.

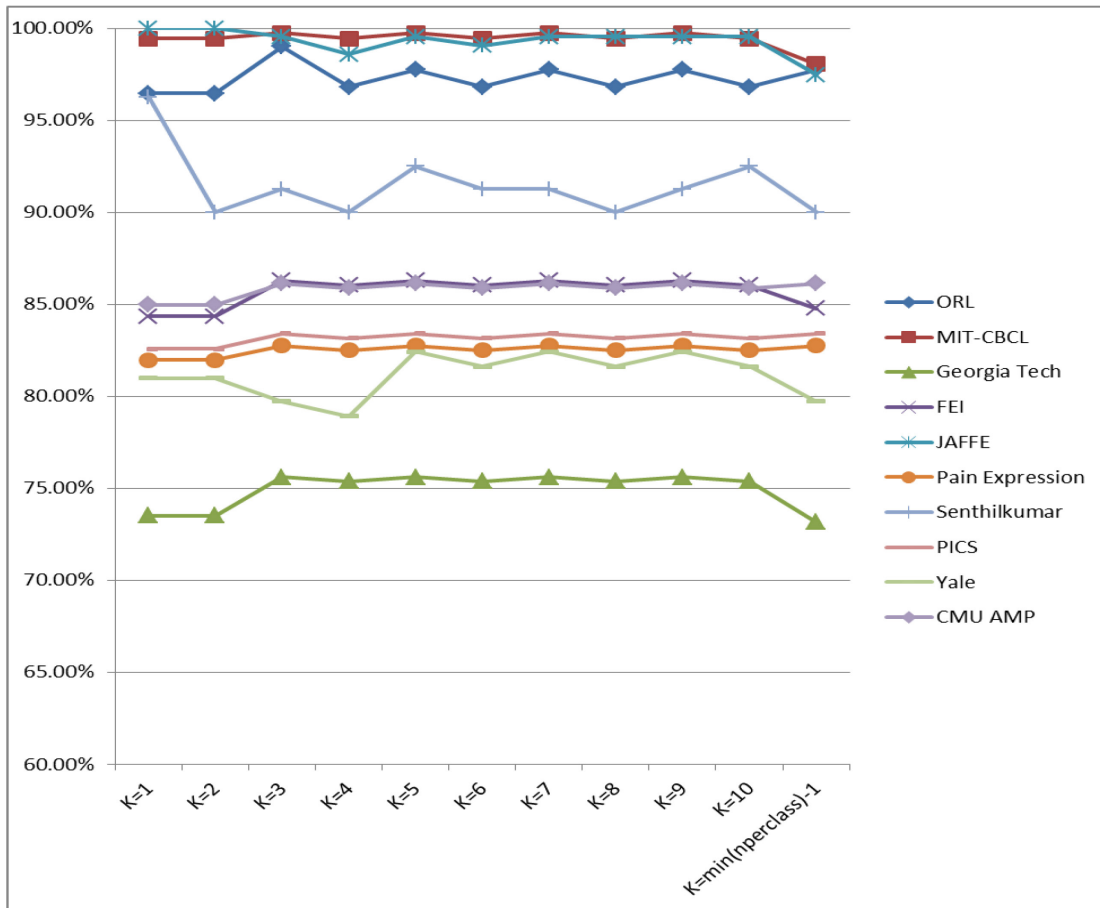


Figure 4.33 The accuracy rate of sgFKNN3 on 10 standard datasets

In addition, the experimental results show that the accuracy rate is similar to sgFKNN1 and has the same pattern when vary on K values but a little bit less than those of sgFKNN1. The misclassifications from sgFKNN3 algorithm are similar to sgFKNN1 and sgFKNN2. They show that although we use different dataset which also make the accuracy rate changed. However, using the same dataset will have the same pattern of accuracy rate, although different algorithm which variable on K values was used. If we use $K=2$ or even values, then the membership values will have an equal opportunity which make the accuracy rate dropped.

Table 4.26 The experimental results from sgFKNN4 for face recognition datasets.

Dataset	K=1	K=2	K=3	K=4	K=5	K=6	K=7	K=8	K=9	K=10	K=C
ORL	86.20%	86.20%	88.47%	86.51%	87.34%	86.51%	87.34%	86.51%	87.34%	86.51%	87.34%
MIT-CBCL	99.20%	98.50%	99.50%	98.50%	99.50%	98.50%	99.50%	98.50%	99.50%	98.50%	97.80%
Georgia Tech	60.80%	59.03%	62.53%	61.90%	62.53%	61.90%	62.53%	61.90%	62.53%	61.90%	60.51%
FEI	74.98%	72.68%	76.71%	75.94%	76.71%	75.94%	76.71%	75.94%	76.71%	75.94%	75.37%
JAFFE	100%	99.06%	99.53%	98.59%	99.53%	99.06%	99.53%	99.06%	99.06%	99.06%	97.48%
Pain Expression	72.87%	71.39%	73.56%	72.82%	73.56%	72.82%	73.56%	72.82%	73.56%	72.82%	73.56%
Senthikumar	96.25%	90%	90%	88.75%	90.25%	N/A	N/A	N/A	N/A	N/A	90.25%
PICS	73.40%	72.76%	74.15%	73.40%	74.15%	73.40%	74.15%	73.40%	74.15%	73.40%	74.15%
Yale	71.99%	71.99%	70.88%	70.06%	73.03%	71.21%	73.03%	71.21%	73.03%	71.21%	70.88%
CMU AMP	75.55%	74.52%	76.57%	75.80%	76.57%	75.80%	76.57%	75.80%	76.57%	75.80%	76.57%

Note. We have N/A in some results because the dataset has number of classes less than number of K .

From the experimental results, we found that the best accuracy rate on ORL is 88.47% when using $K = 3$. For Pain Expression, PICS and CMU AMP database the accuracy rates are 73.56%, 74.15% and 76.57%, respectively, when using $K = 3, 5, 7, 9$ and $\min(\text{nperclass})-1$. For MIT-CBCL, Georgia Tech and FEI databases the accuracy rates are 99.50%, 62.53%, and 76.71%, respectively, when using $K = 3, 5, 7, \text{ and } 9$.

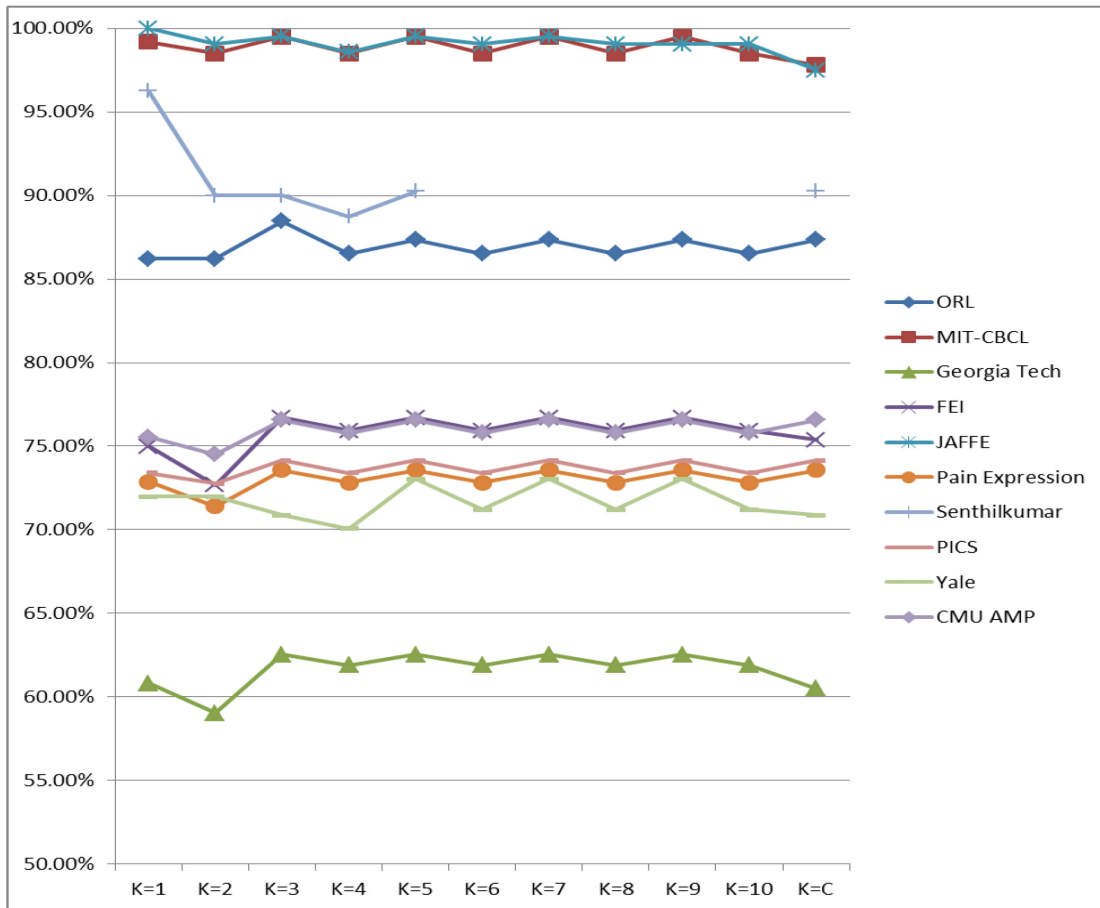


Figure 4.34 The accuracy rate of sgFKNN4 on 10 standard datasets

In addition, the experimental results show that the accuracy rates are dropped from sgFKNN1, sgFKNN2 and sgFKNN3 but still the same pattern when vary on K values.

Nevertheless, we implemented the multi-prototypes on sgFKNN4. We set the number of prototypes on each dataset to about 1%, 5%, 10%, 15%, and 20% of the minimum number of images for each class. For ORL, Georgia Tech, FEI, JAFFE, Pain Expression, Senthilkumar, PICS and Yale datasets, the minimum number of images for each class are 10, 15, 14, 19, 26, 16, 9 and 11, respectively; we set the number of prototypes to be 1, 2, 3, 4 and 5. For MIT-CBCL dataset datasets, the minimum number of images for each class is 200; we set the number of prototypes to be 1, 10, 20, 30 and 40. We chose K value according to the highest accuracy rate on each dataset. For CMU AMP dataset, the minimum number of images for each class is 75; we set the number of prototypes to be 1, 5, 10, 15 and 20.

Table 4.27 The experimental results from sgFKNN4 for face recognition datasets when using the different number of prototypes.

Dataset	Number of prototypes	1	2	3	4	5
ORL	$K=1$	86.20%	88.02%	89.89%	86.40%	85.63%
	$K=2$	86.20%	88.02%	89.89%	86.40%	85.63%
	$K=3$	88.47%	90.34%	92.26%	88.68%	87.89%
	$K=4$	86.51%	88.34%	90.22%	86.72%	85.94%
	$K=5$	87.34%	89.19%	91.08%	87.55%	86.77%
	$K=6$	86.51%	88.34%	90.22%	86.72%	85.94%
	$K=7$	87.34%	89.19%	91.08%	87.55%	86.77%
	$K=8$	86.51%	88.34%	90.22%	86.72%	85.94%
	$K=9$	87.34%	89.19%	91.08%	87.55%	86.77%
	$K=10$	86.51%	88.34%	90.22%	86.72%	85.94%
	$K=C$	87.34%	89.19%	91.08%	87.55%	86.77%
MIT-CBCL	Number of prototypes	1	10	20	30	40
	$K=1$	99.20%	99.20%	99.20%	98.50%	98.50%
	$K=2$	98.50%	98.50%	98.50%	97.81%	97.81%
	$K=3$	99.50%	99.50%	99.50%	98.80%	98.80%
	$K=4$	98.50%	98.50%	98.50%	97.81%	97.81%
	$K=5$	99.50%	99.50%	99.50%	98.80%	98.80%
	$K=6$	98.50%	98.50%	98.50%	97.81%	97.81%
	$K=7$	99.50%	99.50%	99.50%	98.80%	98.80%
	$K=8$	98.50%	98.50%	98.50%	97.81%	97.81%
	$K=9$	99.50%	99.50%	99.50%	98.80%	98.80%
	$K=10$	98.50%	98.50%	98.50%	97.81%	97.81%
$K=C$	97.80%	97.80%	97.80%	97.11%	97.11%	
Georgia Tech	Number of prototypes	1	2	3	4	5
	$K=1$	60.80%	61.35%	60.65%	60.07%	59.20%
	$K=2$	59.03%	59.57%	58.89%	58.32%	57.47%
	$K=3$	62.53%	63.10%	62.38%	61.78%	60.88%
	$K=4$	61.90%	62.46%	61.75%	61.16%	60.27%
	$K=5$	62.53%	63.10%	62.38%	61.78%	60.88%
	$K=6$	61.90%	62.46%	61.75%	61.16%	60.27%
	$K=7$	62.53%	63.10%	62.38%	61.78%	60.88%
	$K=8$	61.90%	62.46%	61.75%	61.16%	60.27%
	$K=9$	62.53%	63.10%	62.38%	61.78%	60.88%
	$K=10$	61.90%	62.46%	61.75%	61.16%	60.27%
$K=C$	60.51%	61.06%	60.36%	59.78%	58.91%	

Table 4.27 The experimental results from sgFKNN4 for face recognition datasets when using the different number of prototypes (Cont.).

Dataset	Number of prototypes	1	2	3	4	5
FEI	$K=1$	74.98%	74.81%	74.30%	74.12%	73.51%
	$K=2$	72.68%	72.52%	72.02%	71.85%	71.26%
	$K=3$	76.71%	76.54%	76.01%	75.83%	75.21%
	$K=4$	75.94%	75.77%	75.25%	75.07%	74.46%
	$K=5$	76.71%	76.54%	76.01%	75.83%	75.21%
	$K=6$	75.94%	75.77%	75.25%	75.07%	74.46%
	$K=7$	76.71%	76.54%	76.01%	75.83%	75.21%
	$K=8$	75.94%	75.77%	75.25%	75.07%	74.46%
	$K=9$	76.71%	76.54%	76.01%	75.83%	75.21%
	$K=10$	75.94%	75.77%	75.25%	75.07%	74.46%
	$K=C$	75.37%	75.20%	74.68%	74.51%	73.90%
JAFPE	Number of prototypes	1	2	3	4	5
	$K=1$	100%	100.00%	99.53%	98.64%	97.48%
	$K=2$	99.06%	99.06%	98.59%	97.71%	96.56%
	$K=3$	99.53%	99.53%	99.06%	98.18%	97.02%
	$K=4$	98.59%	98.59%	98.13%	97.25%	96.11%
	$K=5$	99.53%	99.53%	99.06%	98.18%	97.02%
	$K=6$	99.06%	99.06%	98.59%	97.71%	96.56%
	$K=7$	99.53%	99.53%	99.06%	98.18%	97.02%
	$K=8$	99.06%	99.06%	98.59%	97.71%	96.56%
	$K=9$	99.06%	99.06%	98.59%	97.71%	96.56%
	$K=10$	99.06%	99.06%	98.59%	97.71%	96.56%
$K=C$	97.48%	97.48%	97.02%	96.15%	95.02%	
Pain Expression	Number of prototypes	1	2	3	4	5
	$K=1$	72.87%	73.48%	74.09%	72.76%	72.28%
	$K=2$	71.39%	71.99%	72.59%	71.28%	70.81%
	$K=3$	73.56%	74.18%	74.79%	73.45%	72.96%
	$K=4$	72.82%	73.43%	74.04%	72.71%	72.23%
	$K=5$	73.56%	74.18%	74.79%	73.45%	72.96%
	$K=6$	72.82%	73.43%	74.04%	72.71%	72.23%
	$K=7$	73.56%	74.18%	74.79%	73.45%	72.96%
	$K=8$	72.82%	73.43%	74.04%	72.71%	72.23%
	$K=9$	73.56%	74.18%	74.79%	73.45%	72.96%
	$K=10$	72.82%	73.43%	74.04%	72.71%	72.23%
$K=C$	73.56%	74.18%	74.79%	73.45%	72.96%	

Table 4.27 The experimental results from sgFKNN4 for face recognition datasets when using the different number of prototypes (Cont.).

Dataset	Number of prototypes	1	2	3	4	5
Senthilkumar	$K=1$	96.25%	97.35%	97.78%	97.93%	95.44%
	$K=2$	90%	91.03%	91.43%	91.57%	89.24%
	$K=3$	90%	91.03%	91.43%	91.57%	89.24%
	$K=4$	88.75%	89.76%	90.16%	90.30%	88.00%
	$K=5$	90.25%	91.28%	91.68%	91.83%	89.49%
	$K=6$	N/A	N/A	N/A	N/A	N/A
	$K=7$	N/A	N/A	N/A	N/A	N/A
	$K=8$	N/A	N/A	N/A	N/A	N/A
	$K=9$	N/A	N/A	N/A	N/A	N/A
	$K=10$	N/A	N/A	N/A	N/A	N/A
	$K=C$	90.25%	91.28%	91.68%	91.83%	89.49%
PICS	Number of prototypes	1	2	3	4	5
	$K=1$	73.40%	74.26%	73.37%	73.06%	72.94%
	$K=2$	72.76%	73.62%	72.73%	72.43%	72.31%
	$K=3$	74.15%	75.02%	74.12%	73.81%	73.69%
	$K=4$	73.40%	74.26%	73.37%	73.06%	72.94%
	$K=5$	74.15%	75.02%	74.12%	73.81%	73.69%
	$K=6$	73.40%	74.26%	73.37%	73.06%	72.94%
	$K=7$	74.15%	75.02%	74.12%	73.81%	73.69%
	$K=8$	73.40%	74.26%	73.37%	73.06%	72.94%
	$K=9$	74.15%	75.02%	74.12%	73.81%	73.69%
	$K=10$	73.40%	74.26%	73.37%	73.06%	72.94%
$K=C$	74.15%	75.02%	74.12%	73.81%	73.69%	
Yale	Number of prototypes	1	2	3	4	5
	$K=1$	71.99%	72.51%	72.90%	71.74%	71.18%
	$K=2$	71.99%	72.51%	72.90%	71.74%	71.18%
	$K=3$	70.88%	71.39%	71.77%	70.64%	70.08%
	$K=4$	70.06%	70.57%	70.94%	69.82%	69.27%
	$K=5$	73.03%	73.56%	73.95%	72.78%	72.21%
	$K=6$	71.21%	71.72%	72.11%	70.97%	70.41%
	$K=7$	73.03%	73.56%	73.95%	72.78%	72.21%
	$K=8$	71.21%	71.72%	72.11%	70.97%	70.41%
	$K=9$	73.03%	73.56%	73.95%	72.78%	72.21%
	$K=10$	71.21%	71.72%	72.11%	70.97%	70.41%
$K=C$	70.88%	71.39%	71.77%	70.64%	70.08%	

Note. We have N/A in some results because the dataset has number of classes less than number of K .

Table 4.27 The experimental results from sgFKNN4 for face recognition datasets when using the different number of prototypes (Cont.).

Dataset	Number of prototypes	1	5	10	15	20
CMU AMP	$K=1$	75.55%	75.80%	76.01%	75.56%	74.73%
	$K=2$	74.52%	74.77%	74.98%	74.53%	73.71%
	$K=3$	76.57%	76.83%	77.04%	76.58%	75.74%
	$K=4$	75.80%	76.05%	76.26%	75.81%	74.98%
	$K=5$	76.57%	76.83%	77.04%	76.58%	75.74%
	$K=6$	75.80%	76.05%	76.26%	75.81%	74.98%
	$K=7$	76.57%	76.83%	77.04%	76.58%	75.74%
	$K=8$	75.80%	76.05%	76.26%	75.81%	74.98%
	$K=9$	76.57%	76.83%	77.04%	76.58%	75.74%
	$K=10$	75.80%	76.05%	76.26%	75.81%	74.98%
	$K=C$	76.57%	76.83%	77.04%	76.58%	75.74%

From table 4.27, we found that the best accuracy rate on ORL, MIT-CBCL, Georgia Tech, FEI, JAFFE, Pain Expression, Senthilkumar, PICS, Yale and CMU AMP dataset are 92.26%, 99.50%, 63.10%, 76.71%, 100.00%, 74.79%, 97.93%, 75.02%, 73.95% and 77.04%, respectively.

For above accuracy rates when using $K = 3$ and number of prototypes is 3 for ORL database, $K = 3, 5, 7, 9$ and number of prototypes are 1, 10, 20 for MIT-CBCL database, $K = 3, 5, 7, 9$ and number of prototypes is 2 for Georgia Tech database, $K = 3, 5, 7, 9$ and number of prototypes is 1 for FEI database, $K = 1$ and number of prototypes are 1 and 2 for JAFFE database, $K = 3, 5, 7, 9, C$ and number of prototypes is 3 for Pain Expression database, $K = 1$ and number of prototype is 4 for Senthilkumar database, $K = 3, 5, 7, 9, C$ and number of prototypes is 2 for PICS database, $K = 5, 7, 9$ and number of prototypes is 3 for Yale database, $K = 3, 5, 7, 9, C$ and number of prototype is 10 for Pain Expression database.

The number of prototypes which provides the highest accuracy rate is about 5% to 15% of the number of samples in each class. However, if the number of prototypes is more than the optimal value, then accuracy will be dropped because some prototypes in one class may be outliers.

However, we found that the accuracy rate of some datasets will be drastically dropped. It is shown that the sgFKNN1 to sgFKNN3 can manage those datasets better than sgFKNN4.

Table 4.28 The experimental results from sgFKNN5 for face recognition datasets.

Dataset	K=1	K=2	K=3	K=4	K=5	K=6	K=7	K=8	K=9	K=10	K=C
ORL	95.73%	93.19%	98.25%	96.08%	97.00%	96.08%	97.00%	96.08%	97.00%	96.08%	97.00%
MIT-CBCL	99.25%	98.65%	99.55%	98.65%	99.55%	98.65%	99.55%	98.65%	99.55%	98.65%	97.85%
Georgia Tech	67.16%	65.21%	69.07%	68.45%	69.07%	68.45%	69.07%	68.45%	69.07%	68.45%	66.84%
FEI	81.55%	79.04%	83.43%	82.68%	83.43%	82.68%	83.43%	82.68%	83.43%	82.68%	81.97%
JAFFE	100%	99.06%	99.53%	98.59%	99.53%	99.06%	99.53%	99.06%	99.06%	99.06%	97.48%
Pain Expression	79.26%	77.65%	80.01%	79.29%	80.01%	79.29%	80.01%	79.29%	80.01%	79.29%	80.01%
Senthikumar	96.25%	90%	90%	88.75%	90%	N/A	N/A	N/A	N/A	N/A	90%
PICS	79.84%	79.14%	80.65%	79.92%	80.65%	79.92%	80.65%	79.92%	80.65%	79.92%	80.65%
Yale	78.29%	78.29%	77.08%	76.39%	81.81%	79.39%	81.81%	79.39%	81.81%	79.39%	77.08%
CMU AMP	82.16%	81.04%	83.27%	82.52%	83.27%	82.52%	83.27%	82.52%	83.27%	82.52%	83.27%

Note. We have N/A in some results because the dataset has number of classes less than number of K .

From the experimental results, we found that the best accuracy rate on ORL is 98.25% when using $K = 3$. For Pain Expression, PICS and CMU AMP database are 80.01%, 80.65% and 83.27%, respectively, when using $K = 3, 5, 7, 9$ and $\min(\text{nperclass})-1$. For

MIT-CBCL, Georgia Tech and FEI databases the accuracy rates are 99.55%, 69.07%, and 83.43%, respectively, when using $K = 3, 5, 7,$ and 9.

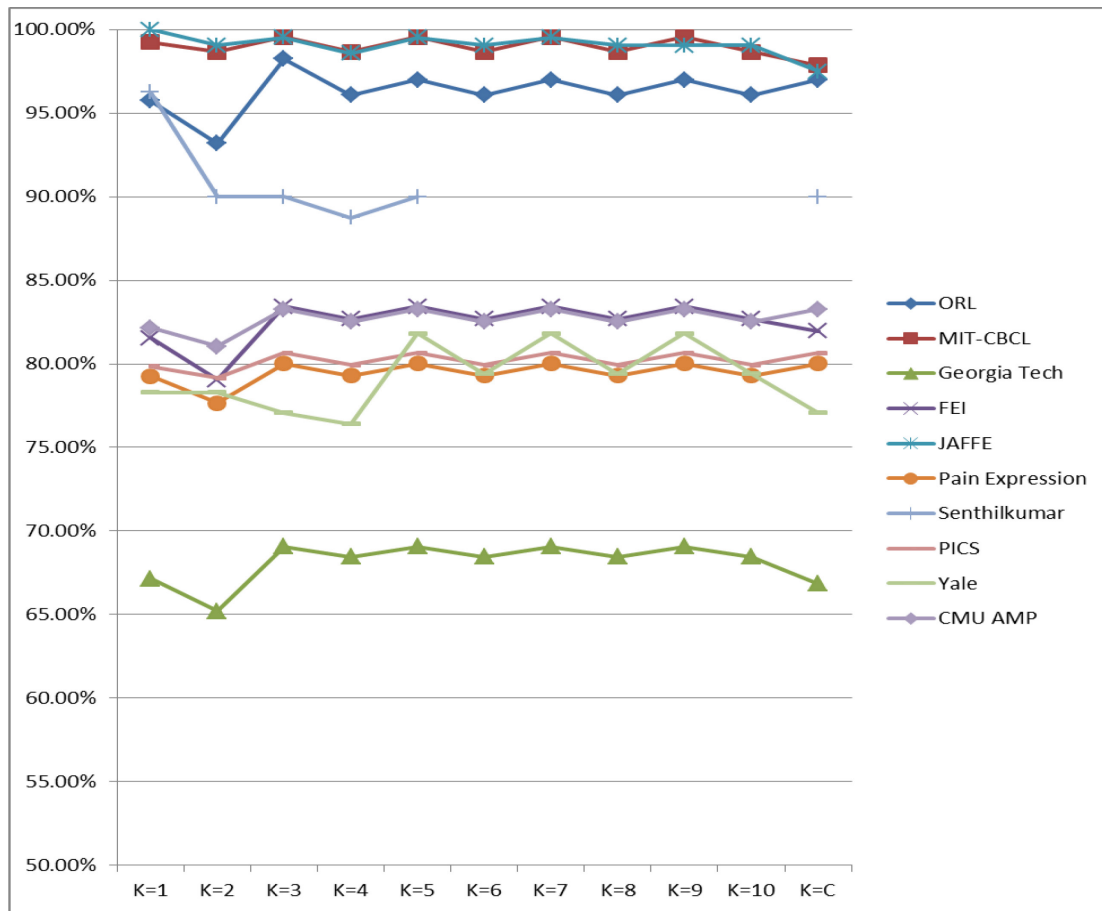


Figure 4.35 The accuracy rate of sgFKNN5 on 10 standard datasets

In addition, the experimental results show that the pattern of accuracy rate is similar to sgFKNN1 when vary on K values. The misclassifications from sgFKNN5 algorithm are similar to sgFKNN1 to sgFKNN4. It shows that if we use the same algorithm but using different datasets, the accuracy rates will be changed as well. Meanwhile, using the different algorithm and the same datasets, the accuracy rates will have same pattern of the change when varying on K values. If we use $K=2$ or even values, then the membership values will have an equal opportunity which make the accuracy rate dropped. Next, we found that the accuracy rate of some datasets will be drastically dropped. It is shown that the sgFKNN1 to sgFKNN3 can manage those datasets better than sgFKNN5. Nevertheless, we implemented the multi-prototypes on sgFKNN5. We

set the number of prototypes on each dataset same with the multi-prototypes on sgFKNN4 as follows:

Table 4.29 The experimental results from sgFKNN5 for face recognition datasets when using the different number of prototypes.

Dataset	Number of prototypes	1	2	3	4	5
ORL	$K=1$	95.73%	96.29%	96.42%	94.49%	93.64%
	$K=2$	93.19%	96.29%	96.42%	94.49%	93.64%
	$K=3$	98.25%	98.83%	98.96%	96.98%	96.11%
	$K=4$	96.08%	96.64%	96.77%	94.83%	93.98%
	$K=5$	97.00%	97.57%	97.70%	95.74%	94.88%
	$K=6$	96.08%	96.64%	96.77%	94.83%	93.98%
	$K=7$	97.00%	97.57%	97.70%	95.74%	94.88%
	$K=8$	96.08%	96.64%	96.77%	94.83%	93.98%
	$K=9$	97.00%	97.57%	97.70%	95.74%	94.88%
	$K=10$	96.08%	96.64%	96.77%	94.83%	93.98%
	$K=C$	97.00%	97.57%	97.70%	95.74%	94.88%
MIT-CBCL	Number of prototypes	1	10	20	30	40
	$K=1$	99.25%	99.25%	99.25%	98.55%	98.55%
	$K=2$	98.65%	98.65%	98.65%	97.96%	97.96%
	$K=3$	99.55%	99.55%	99.55%	98.85%	98.85%
	$K=4$	98.65%	98.65%	98.65%	97.96%	97.96%
	$K=5$	99.55%	99.55%	99.55%	98.85%	98.85%
	$K=6$	98.65%	98.65%	98.65%	97.96%	97.96%
	$K=7$	99.55%	99.55%	99.55%	98.85%	98.85%
	$K=8$	98.65%	98.65%	98.65%	97.96%	97.96%
	$K=9$	99.55%	99.55%	99.55%	98.85%	98.85%
	$K=10$	98.65%	98.65%	98.65%	97.96%	97.96%
$K=C$	97.85%	97.85%	97.85%	97.16%	97.16%	
Georgia Tech	Number of prototypes	1	2	3	4	5
	$K=1$	67.16%	67.77%	66.99%	66.35%	65.39%
	$K=2$	65.21%	65.80%	65.05%	64.43%	63.49%
	$K=3$	69.07%	69.70%	68.90%	68.24%	67.25%
	$K=4$	68.45%	69.07%	68.28%	67.63%	66.65%
	$K=5$	69.07%	69.70%	68.90%	68.24%	67.25%
	$K=6$	68.45%	69.07%	68.28%	67.63%	66.65%
	$K=7$	69.07%	69.70%	68.90%	68.24%	67.25%
	$K=8$	68.45%	69.07%	68.28%	67.63%	66.65%
	$K=9$	69.07%	69.70%	68.90%	68.24%	67.25%
	$K=10$	68.45%	69.07%	68.28%	67.63%	66.65%
$K=C$	66.84%	67.45%	66.68%	66.04%	65.08%	

Table 4.29 The experimental results from sgFKNN5 for face recognition datasets when using the different number of prototypes (Cont.).

Dataset	Number of prototypes	1	2	3	4	5
FEI	$K=1$	81.55%	81.37%	80.81%	80.61%	79.96%
	$K=2$	79.04%	78.88%	78.33%	78.14%	77.50%
	$K=3$	83.43%	83.25%	82.67%	82.47%	81.80%
	$K=4$	82.68%	82.41%	81.84%	81.64%	80.98%
	$K=5$	83.43%	83.25%	82.67%	82.47%	81.80%
	$K=6$	82.68%	82.41%	81.84%	81.64%	80.98%
	$K=7$	83.43%	83.25%	82.67%	82.47%	81.80%
	$K=8$	82.68%	82.41%	81.84%	81.64%	80.98%
	$K=9$	83.43%	83.25%	82.67%	82.47%	81.80%
	$K=10$	82.68%	82.41%	81.84%	81.64%	80.98%
	$K=C$	81.97%	81.80%	81.23%	81.03%	80.37%
JAFPE	Number of prototypes	1	2	3	4	5
	$K=1$	100%	100.00%	99.53%	98.64%	97.48%
	$K=2$	99.06%	99.06%	98.59%	97.71%	96.56%
	$K=3$	99.53%	99.53%	99.06%	98.18%	97.02%
	$K=4$	98.59%	98.59%	98.13%	97.25%	96.11%
	$K=5$	99.53%	99.53%	99.06%	98.18%	97.02%
	$K=6$	99.06%	99.06%	98.59%	97.71%	96.56%
	$K=7$	99.53%	99.53%	99.06%	98.18%	97.02%
	$K=8$	99.06%	99.06%	98.59%	97.71%	96.56%
	$K=9$	99.06%	99.06%	98.59%	97.71%	96.56%
	$K=10$	99.06%	99.06%	98.59%	97.71%	96.56%
$K=C$	97.48%	97.48%	97.02%	96.15%	95.02%	
Pain Expression	Number of prototypes	1	2	3	4	5
	$K=1$	79.26%	79.92%	80.59%	79.14%	78.62%
	$K=2$	77.65%	78.30%	78.95%	77.53%	77.02%
	$K=3$	80.01%	80.68%	81.35%	79.89%	79.36%
	$K=4$	79.29%	79.87%	80.53%	79.09%	78.56%
	$K=5$	80.01%	80.68%	81.35%	79.89%	79.36%
	$K=6$	79.29%	79.87%	80.53%	79.09%	78.56%
	$K=7$	80.01%	80.68%	81.35%	79.89%	79.36%
	$K=8$	79.29%	79.87%	80.53%	79.09%	78.56%
	$K=9$	80.01%	80.68%	81.35%	79.89%	79.36%
	$K=10$	79.29%	79.87%	80.53%	79.09%	78.56%
$K=C$	80.01%	80.68%	81.35%	79.89%	79.36%	

Table 4.29 The experimental results from sgFKNN5 for face recognition datasets when using the different number of prototypes (Cont.).

Dataset	Number of prototypes	1	2	3	4	5
Senthilkumar	$K=1$	96.25%	97.35%	97.78%	97.93%	95.44%
	$K=2$	90%	91.03%	91.43%	91.57%	89.24%
	$K=3$	90%	91.03%	91.43%	91.57%	89.24%
	$K=4$	88.75%	89.76%	90.16%	90.30%	88.00%
	$K=5$	90%	91.28%	91.68%	91.83%	89.49%
	$K=6$	N/A	N/A	N/A	N/A	N/A
	$K=7$	N/A	N/A	N/A	N/A	N/A
	$K=8$	N/A	N/A	N/A	N/A	N/A
	$K=9$	N/A	N/A	N/A	N/A	N/A
	$K=10$	N/A	N/A	N/A	N/A	N/A
	$K=C$	90%	91.28%	91.68%	91.83%	89.49%
PICS	Number of prototypes	1	2	3	4	5
	$K=1$	79.84%	80.77%	79.80%	79.47%	79.34%
	$K=2$	79.14%	80.07%	79.11%	78.78%	78.65%
	$K=3$	80.65%	81.60%	80.62%	80.28%	80.15%
	$K=4$	79.92%	80.77%	79.80%	79.47%	79.34%
	$K=5$	80.65%	81.60%	80.62%	80.28%	80.15%
	$K=6$	79.92%	80.77%	79.80%	79.47%	79.34%
	$K=7$	80.65%	81.60%	80.62%	80.28%	80.15%
	$K=8$	79.92%	80.77%	79.80%	79.47%	79.34%
	$K=9$	80.65%	81.60%	80.62%	80.28%	80.15%
	$K=10$	79.92%	80.77%	79.80%	79.47%	79.34%
$K=C$	80.65%	81.60%	80.62%	80.28%	80.15%	
Yale	Number of prototypes	1	2	3	4	5
	$K=1$	78.29%	81.23%	81.66%	80.37%	79.74%
	$K=2$	78.29%	81.23%	81.66%	80.37%	79.74%
	$K=3$	77.08%	79.97%	80.40%	79.13%	78.51%
	$K=4$	76.39%	79.05%	79.47%	78.21%	77.60%
	$K=5$	81.81%	82.40%	82.84%	81.53%	80.89%
	$K=6$	79.39%	80.35%	80.78%	79.50%	78.87%
	$K=7$	81.81%	82.40%	82.84%	81.53%	80.89%
	$K=8$	79.39%	80.35%	80.78%	79.50%	78.87%
	$K=9$	81.81%	82.40%	82.84%	81.53%	80.89%
	$K=10$	79.39%	80.35%	80.78%	79.50%	78.87%
$K=C$	77.08%	79.97%	80.40%	79.13%	78.51%	

Note. We have N/A in some results because the dataset has number of classes less than number of K .

Table 4.29 The experimental results from sgFKNN5 for face recognition datasets when using the different number of prototypes (Cont.).

Dataset	Number of prototypes	1	5	10	15	20
CMU AMP	$K=1$	82.16%	82.44%	82.66%	82.17%	81.27%
	$K=2$	81.04%	81.31%	81.54%	81.05%	80.16%
	$K=3$	83.27%	83.55%	83.78%	83.27%	82.37%
	$K=4$	82.52%	82.71%	82.94%	82.44%	81.54%
	$K=5$	83.27%	83.55%	83.78%	83.27%	82.37%
	$K=6$	82.52%	82.71%	82.94%	82.44%	81.54%
	$K=7$	83.27%	83.55%	83.78%	83.27%	82.37%
	$K=8$	82.52%	82.71%	82.94%	82.44%	81.54%
	$K=9$	83.27%	83.55%	83.78%	83.27%	82.37%
	$K=10$	82.52%	82.71%	82.94%	82.44%	81.54%
	$K=C$	83.27%	83.55%	83.78%	83.27%	82.37%

The experiment results from sgFKNN5 for face recognition datasets when using the different number of prototypes. We found that the best accuracy rate on ORL, MIT-CBCL, Georgia Tech, FEI, JAFFE, Pain Expression, Senthilkumar, PICS, Yale and CMU AMP dataset are 98.96%, 99.55%, 69.70%, 83.43%, 100.00%, 81.35%, 97.93%, 81.60%, 82.84% and 83.78%, respectively.

For above accuracy rates when using $K = 3$ and number of prototype is 3 for ORL database, $K = 3, 5, 7, 9$ and number of prototypes are 1, 10, 20 for MIT-CBCL database, $K = 3, 5, 7, 9$ and number of prototype is 2 for Georgia Tech database, $K = 3, 5, 7, 9$ and number of prototype is 1 for FEI database, $K = 1$ and number of prototypes are 1 and 2 for JAFFE database, $K = 3, 5, 7, 9, C$ and number of prototype is 3 for Pain Expression database, $K = 1$ and number of prototype is 4 for Senthilkumar database, $K = 3, 5, 7, 9, C$ and number of prototype is 2 for PICS database, $K = 5, 7, 9$ and number of prototype is 3 for Yale database, $K = 3, 5, 7, 9, C$ and number of prototype is 10 for Pain Expression database.

The number of prototypes which provides the highest accuracy rate is about 5% to 15% of the number of sample in each class. However, if the number of prototypes is more than the optimal value, then accuracy will be dropped because some prototypes in one class may be outliers.

Table 4.30 The experimental results from sgFKNN6 for face recognition datasets.

Dataset	K=1	K=2	K=3	K=4	K=5	K=6	K=7	K=8	K=9	K=10	K=C
ORL	95.73%	93.19%	98.25%	96.08%	97.00%	96.08%	97.00%	96.08%	97.00%	96.08%	97.00%
MIT-CBCL	99.20%	98.70%	99.50%	98.70%	99.50%	98.70%	99.50%	98.70%	99.50%	98.70%	97.80%
Georgia Tech	65.08%	63.19%	66.93%	66.39%	66.93%	66.39%	66.93%	66.39%	66.93%	66.39%	64.77%
FEI	80.57%	78.09%	82.43%	81.77%	82.43%	81.77%	82.43%	81.77%	82.43%	81.77%	80.99%
JAFFE	100%	99.06%	99.53%	98.59%	99.53%	99.06%	99.53%	99.06%	99.06%	99.06%	97.48%
Pain Expression	78.31%	76.71%	79.05%	78.41%	79.05%	78.41%	79.05%	78.41%	79.05%	78.41%	79.05%
Senthikumar	96.25%	90%	90%	88.75%	90%	N/A	N/A	N/A	N/A	N/A	90%
PICS	78.89%	78.20%	79.69%	79.05%	79.69%	79.05%	79.69%	79.05%	79.69%	79.05%	79.69%
Yale	77.36%	77.36%	76.16%	75.47%	80.60%	79.39%	80.60%	79.39%	80.60%	79.39%	76.16%
CMU AMP	81.19%	80.07%	82.28%	81.62%	82.28%	81.62%	82.28%	81.62%	82.28%	81.62%	82.28%

Note. We have N/A in some results because the dataset has number of classes less than number of K .

From the experimental results, we found that the best accuracy rate on ORL is 98.25% when using $K = 3$. For Pain Expression, PICS and CMU AMP database the accuracy rates are 79.05%, 79.69% and 82.28%, respectively, when using $K = 3, 5, 7, 9$ and $\min(\text{nperclass})-1$. For MIT-CBCL, Georgia Tech and FEI databases the accuracy rates are 99.50%, 66.93%, and 82.43%, respectively, when using $K = 3, 5, 7, \text{ and } 9$.

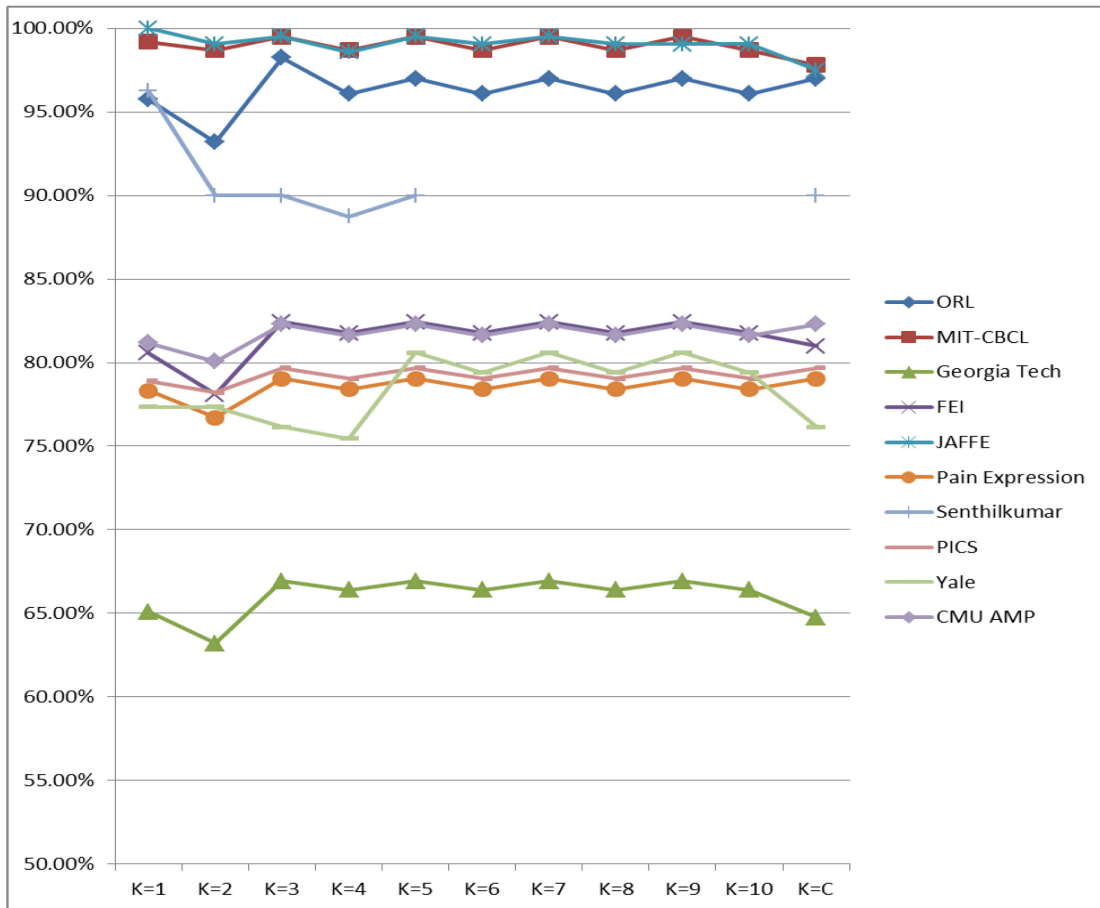


Figure 4.36 The accuracy rate of sgFKNN6 on 10 standard datasets

In addition, the experimental results show that the pattern of accuracy rate is similar to sgFKNN1 when vary on K values. The misclassifications from sgFKNN6 algorithm are similar to sgFKNN5. It shows that if we use the same algorithm but different datasets, the accuracy rates will be also changed. Meanwhile, using the different algorithms and the same datasets, the accuracy rates will have the same patterns when varying on K values. If we use $K=2$ or even values, then the membership values will have an equal opportunity which make the accuracy rate dropped. For instance, in Georgia Tech dataset when $K=1$, accuracy rate is about 65%, $K=2$ accuracy rate is about 63%, $K=3$ accuracy rate is about 67%, $K=4$ accuracy rate is about 66% etc.

Therefore, we found that the accuracy rate of some datasets will be drastically dropped. It is shown that the sgFKNN1 to sgFKNN3 can manage those datasets better than sgFKNN6. Nevertheless, we implemented the multi-prototypes on sgFKNN6. We set

the number of prototypes on each dataset the same values as the multi-prototypes on sgFKNN4 as follows:

Table 4.31 The experimental results from sgFKNN6 for face recognition datasets when using the different number of prototypes.

Dataset	Number of prototypes	1	2	3	4	5
ORL	$K=1$	95.73%	96.29%	96.42%	94.49%	93.64%
	$K=2$	93.19%	96.29%	96.42%	94.49%	93.64%
	$K=3$	98.25%	98.83%	98.96%	96.98%	96.11%
	$K=4$	96.08%	96.64%	96.77%	94.83%	93.98%
	$K=5$	97.00%	97.57%	97.70%	95.74%	94.88%
	$K=6$	96.08%	96.64%	96.77%	94.83%	93.98%
	$K=7$	97.00%	97.57%	97.70%	95.74%	94.88%
	$K=8$	96.08%	96.64%	96.77%	94.83%	93.98%
	$K=9$	97.00%	97.57%	97.70%	95.74%	94.88%
	$K=10$	96.08%	96.64%	96.77%	94.83%	93.98%
	$K=C$	97.00%	97.57%	97.70%	95.74%	94.88%
MIT-CBCL	Number of prototypes	1	10	20	30	40
	$K=1$	99.20%	99.20%	99.20%	98.50%	98.50%
	$K=2$	98.70%	98.70%	98.70%	98.01%	98.01%
	$K=3$	99.50%	99.50%	99.50%	98.80%	98.80%
	$K=4$	98.70%	98.70%	98.70%	98.01%	98.01%
	$K=5$	99.50%	99.50%	99.50%	98.80%	98.80%
	$K=6$	98.70%	98.70%	98.70%	98.01%	98.01%
	$K=7$	99.50%	99.50%	99.50%	98.80%	98.80%
	$K=8$	98.70%	98.70%	98.70%	98.01%	98.01%
	$K=9$	99.50%	99.50%	99.50%	98.80%	98.80%
	$K=10$	98.70%	98.70%	98.70%	98.01%	98.01%
	$K=C$	97.80%	97.80%	97.80%	97.11%	97.11%
Georgia Tech	Number of prototypes	1	2	3	4	5
	$K=1$	65.08%	65.67%	64.92%	64.30%	63.36%
	$K=2$	63.19%	63.77%	63.04%	62.43%	61.52%
	$K=3$	66.93%	67.54%	66.77%	66.13%	65.16%
	$K=4$	66.39%	67.00%	66.23%	65.60%	64.63%
	$K=5$	66.93%	67.54%	66.77%	66.13%	65.16%
	$K=6$	66.39%	67.00%	66.23%	65.60%	64.63%
	$K=7$	66.93%	67.54%	66.77%	66.13%	65.16%
	$K=8$	66.39%	67.00%	66.23%	65.60%	64.63%
	$K=9$	66.93%	67.54%	66.77%	66.13%	65.16%
	$K=10$	66.39%	67.00%	66.23%	65.60%	64.63%
	$K=C$	64.77%	65.36%	64.62%	64.00%	63.06%

Table 4.31 The experimental results from sgFKNN6 for face recognition datasets when using the different number of prototypes (Cont.).

Dataset	Number of prototypes	1	2	3	4	5
FEI	$K=1$	80.57%	80.40%	79.84%	79.64%	79.00%
	$K=2$	78.09%	77.93%	77.39%	77.20%	76.57%
	$K=3$	82.43%	82.25%	81.68%	81.48%	80.82%
	$K=4$	81.77%	81.42%	80.86%	80.66%	80.01%
	$K=5$	82.43%	82.25%	81.68%	81.48%	80.82%
	$K=6$	81.77%	81.42%	80.86%	80.66%	80.01%
	$K=7$	82.43%	82.25%	81.68%	81.48%	80.82%
	$K=8$	81.77%	81.42%	80.86%	80.66%	80.01%
	$K=9$	82.43%	82.25%	81.68%	81.48%	80.82%
	$K=10$	81.77%	81.42%	80.86%	80.66%	80.01%
	$K=C$	80.99%	80.81%	80.25%	80.06%	79.41%
JAFFE	Number of prototypes	1	2	3	4	5
	$K=1$	100%	100.00%	99.53%	98.64%	97.48%
	$K=2$	99.06%	99.06%	98.59%	97.71%	96.56%
	$K=3$	99.53%	99.53%	99.06%	98.18%	97.02%
	$K=4$	98.59%	98.59%	98.13%	97.25%	96.11%
	$K=5$	99.53%	99.53%	99.06%	98.18%	97.02%
	$K=6$	99.06%	99.06%	98.59%	97.71%	96.56%
	$K=7$	99.53%	99.53%	99.06%	98.18%	97.02%
	$K=8$	99.06%	99.06%	98.59%	97.71%	96.56%
	$K=9$	99.06%	99.06%	98.59%	97.71%	96.56%
	$K=10$	99.06%	99.06%	98.59%	97.71%	96.56%
$K=C$	97.48%	97.48%	97.02%	96.15%	95.02%	
Pain Expression	Number of prototypes	1	2	3	4	5
	$K=1$	78.31%	78.97%	79.62%	78.19%	77.67%
	$K=2$	76.71%	77.37%	78.00%	76.60%	76.10%
	$K=3$	79.05%	79.72%	80.37%	78.93%	78.41%
	$K=4$	78.41%	78.92%	79.56%	78.14%	77.62%
	$K=5$	79.05%	79.72%	80.37%	78.93%	78.41%
	$K=6$	78.41%	78.92%	79.56%	78.14%	77.62%
	$K=7$	79.05%	79.72%	80.37%	78.93%	78.41%
	$K=8$	78.41%	78.92%	79.56%	78.14%	77.62%
	$K=9$	79.05%	79.72%	80.37%	78.93%	78.41%
	$K=10$	78.41%	78.92%	79.56%	78.14%	77.62%
$K=C$	79.05%	79.72%	80.37%	78.93%	78.41%	

Table 4.31 The experimental results from sgFKNN6 for face recognition datasets when using the different number of prototypes (Cont.).

Dataset	Number of prototypes	1	2	3	4	5
Senthilkumar	$K=1$	96.25%	97.35%	97.78%	97.93%	95.44%
	$K=2$	90%	91.03%	91.43%	91.57%	89.24%
	$K=3$	90%	91.03%	91.43%	91.57%	89.24%
	$K=4$	88.75%	89.76%	90.16%	90.30%	88.00%
	$K=5$	90%	91.28%	91.68%	91.83%	89.49%
	$K=6$	N/A	N/A	N/A	N/A	N/A
	$K=7$	N/A	N/A	N/A	N/A	N/A
	$K=8$	N/A	N/A	N/A	N/A	N/A
	$K=9$	N/A	N/A	N/A	N/A	N/A
	$K=10$	N/A	N/A	N/A	N/A	N/A
	$K=C$	90%	91.28%	91.68%	91.83%	89.49%
PICS	Number of prototypes	1	2	3	4	5
	$K=1$	78.89%	79.81%	78.85%	78.52%	78.40%
	$K=2$	78.20%	79.12%	78.17%	77.83%	77.72%
	$K=3$	79.69%	80.63%	79.66%	79.32%	79.20%
	$K=4$	79.05%	79.81%	78.85%	78.52%	78.40%
	$K=5$	79.69%	80.63%	79.66%	79.32%	79.20%
	$K=6$	79.05%	79.81%	78.85%	78.52%	78.40%
	$K=7$	79.69%	80.63%	79.66%	79.32%	79.20%
	$K=8$	79.05%	79.81%	78.85%	78.52%	78.40%
	$K=9$	79.69%	80.63%	79.66%	79.32%	79.20%
	$K=10$	79.05%	79.81%	78.85%	78.52%	78.40%
$K=C$	79.69%	80.63%	79.66%	79.32%	79.20%	
Yale	Number of prototypes	1	2	3	4	5
	$K=1$	77.36%	80.02%	80.46%	79.18%	78.57%
	$K=2$	77.36%	80.02%	80.46%	79.18%	78.57%
	$K=3$	76.16%	78.79%	79.22%	77.96%	77.35%
	$K=4$	75.47%	77.88%	78.30%	77.05%	76.46%
	$K=5$	80.60%	81.18%	81.62%	80.32%	79.70%
	$K=6$	79.39%	79.16%	79.59%	78.32%	77.71%
	$K=7$	80.60%	81.18%	81.62%	80.32%	79.70%
	$K=8$	79.39%	79.16%	79.59%	78.32%	77.71%
	$K=9$	80.60%	81.18%	81.62%	80.32%	79.70%
	$K=10$	79.39%	79.16%	79.59%	78.32%	77.71%
$K=C$	76.16%	78.79%	79.22%	77.96%	77.35%	

Note. We have N/A in some results because the dataset has number of classes less than number of K .

Table 4.31 The experimental results from sgFKNN6 for face recognition datasets when using the different number of prototypes (Cont.).

Dataset	Number of prototypes	1	5	10	15	20
CMU AMP	$K=1$	81.19%	81.46%	81.69%	81.19%	80.31%
	$K=2$	80.07%	80.35%	80.57%	80.09%	79.21%
	$K=3$	82.28%	82.56%	82.79%	82.29%	81.39%
	$K=4$	81.62%	81.73%	81.96%	81.46%	80.57%
	$K=5$	82.28%	82.56%	82.79%	82.29%	81.39%
	$K=6$	81.62%	81.73%	81.96%	81.46%	80.57%
	$K=7$	82.28%	82.56%	82.79%	82.29%	81.39%
	$K=8$	81.62%	81.73%	81.96%	81.46%	80.57%
	$K=9$	82.28%	82.56%	82.79%	82.29%	81.39%
	$K=10$	81.62%	81.73%	81.96%	81.46%	80.57%
	$K=C$	82.28%	82.56%	82.79%	82.29%	81.39%

The experiment results from sgFKNN6 for face recognition datasets when using the different number of prototypes. We found that the best accuracy rate on ORL, MIT-CBCL, Georgia Tech, FEI, JAFFE, Pain Expression, Senthilkumar, PICS, Yale and CMU AMP dataset are 98.96%, 99.55%, 67.54%, 82.43%, 100.00%, 80.37%, 97.93%, 80.63%, 81.62% and 82.79%, respectively.

For above accuracy rates when using $K = 3$ and number of prototype is 3 for ORL database, $K = 3, 5, 7, 9$ and number of prototypes are 1, 10, 20 for MIT-CBCL database, $K = 3, 5, 7, 9$ and number of prototype is 2 for Georgia Tech database, $K = 3, 5, 7, 9$ and number of prototype is 1 for FEI database, $K = 1$ and number of prototypes are 1 and 2 for JAFFE database, $K = 3, 5, 7, 9, C$ and number of prototype is 3 for Pain Expression database, $K = 1$ and number of prototype is 4 for Senthilkumar database, $K = 3, 5, 7, 9, C$ and number of prototype is 2 for PICS database, $K = 5, 7, 9$ and number of prototype is 3 for Yale database, $K = 3, 5, 7, 9, C$ and number of prototype is 10 for Pain Expression database.

The number of prototypes which provides the highest accuracy rate is about 5% to 15% of the number of sample in each class. However, if the number of prototypes is more than the optimal value, then accuracy will be dropped because some prototypes in one class may be outliers.

From the multi-prototype experiments, we found that the number of prototypes which provides the highest accuracy rate should be the optimal value. However, if the number of prototypes is more than the optimal value, then accuracy will be dropped because some prototypes in one class may be outliers.

Table 4.32 The experimental results from sgFKNN7 for face recognition datasets.

Dataset	$K=1$	$K=2$	$K=3$	$K=4$	$K=5$	$K=6$	$K=7$	$K=8$	$K=9$	$K=10$	$K=\min(\text{nperclass})-1$
ORL	94.74%	94.74%	97.24%	95.09%	96.00%	95.09%	96.00%	95.09%	96.00%	95.09%	96.00%
MIT-CBCL	99.45%	99.45%	99.75%	99.45%	99.75%	99.45%	99.75%	99.45%	99.75%	99.45%	98.05%
Georgia Tech	73.51%	73.51%	75.60%	75.37%	75.60%	75.37%	75.60%	75.37%	75.60%	75.37%	73.16%
FEI	84.34%	84.34%	86.29%	86.03%	86.29%	86.03%	86.29%	86.03%	86.29%	86.03%	84.78%
JAFFE	100%	100%	99.53%	98.59%	100%	99.06%	99.53%	99.53%	99.53%	99.53%	97.48%
Pain Expression	81.98%	81.98%	82.75%	82.50%	82.75%	82.50%	82.75%	82.50%	82.75%	82.50%	82.75%
Senthikumar	96.25%	93.75%	91.25%	90%	92.50%	91.25%	91.25%	90%	91.25%	90%	90%
PICS	82.57%	82.57%	83.41%	83.16%	83.41%	83.16%	83.41%	83.16%	83.41%	83.16%	83.41%
Yale	80.98%	80.98%	79.73%	78.92%	83.63%	82.82%	83.63%	82.82%	83.63%	82.82%	79.73%
CMU AMP	84.98%	84.98%	86.13%	85.87%	86.13%	85.87%	86.13%	85.87%	86.13%	85.87%	86.13%

From the experimental results, we found that the best accuracy rate on ORL is 97.24% when using $K = 3$. For Pain Expression, PICS and CMU AMP database are 82.75%, 83.41% and 86.13%, respectively, when using $K = 3, 5, 7, 9$ and $\min(\text{nperclass})-1$. For

MIT-CBCL, Georgia Tech and FEI databases the accuracy rates are 99.75%, 75.60%, and 86.29%, respectively, when using $K = 3, 5, 7,$ and 9.

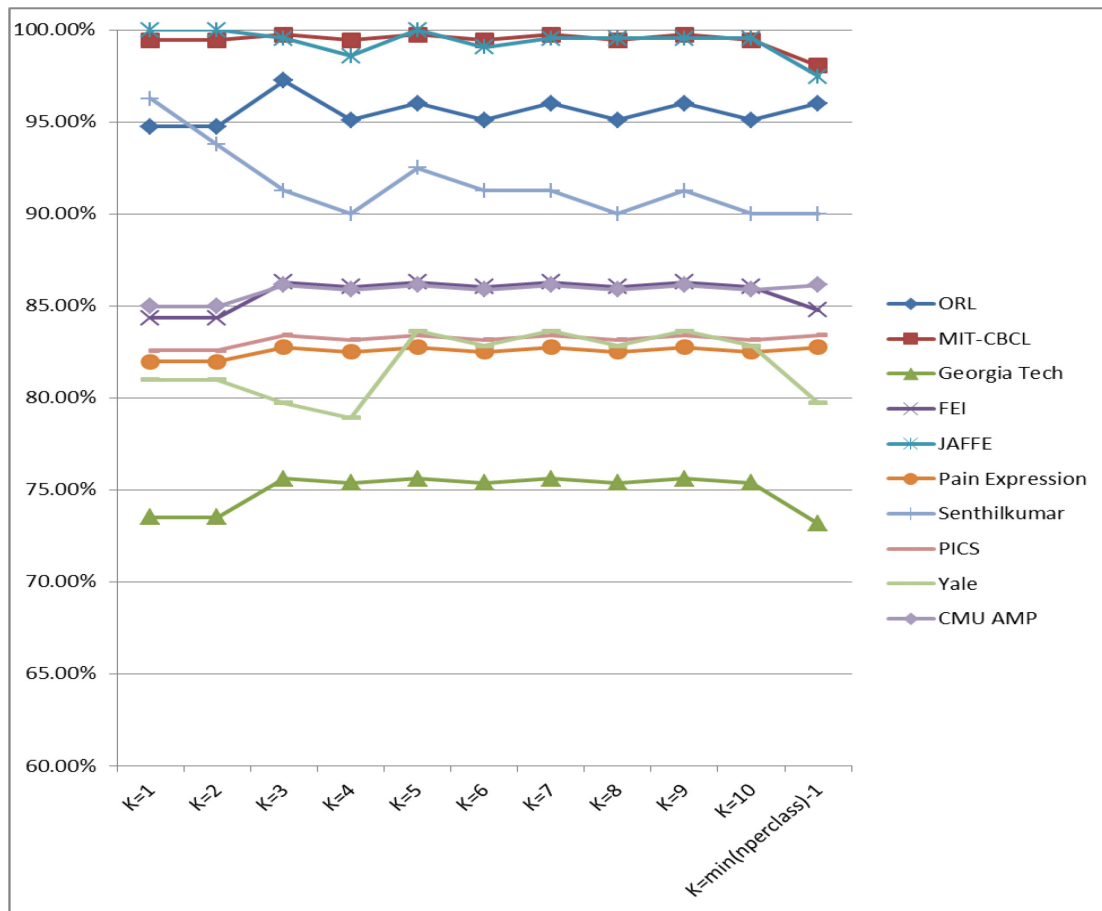


Figure 4.37 The accuracy rate of sgFKNN7 on 10 standard datasets

In addition, the experimental results show that the accuracy rate is similar to sgFKNN1 and has the same pattern when vary on K values but a little bit less than those of sgFKNN1. The misclassifications from sgFKNN7 algorithm are similar to sgFKNN1, sgFKNN2 and sgFKNN3. It shows that if we use the same algorithm but using different datasets, the accuracy rates will be changed as well. Meanwhile, using the different algorithm and the same datasets, the accuracy rates will have same pattern of the change when varying on K values. If we use $K=2$ or even values, then the membership values will have an equal opportunity which make the accuracy rate dropped.

From the above experimental results from sgFKNN1 to sgFKNN7, we found that the accuracy rate in sgFKNN1 is higher than the other algorithms. Moreover, it produces the best accuracy rate in most of the datasets when the K values are 3, 5, 7, 9, and $\min(\text{nperclass})-1$, and the K value is 1 for minority dataset whose reasons can be shown as follows:

1. If the dataset with high within-classes variation and low between-classes variation, $K=1$ should be appropriate as shown in figure 4.26.
2. If the dataset has high between-classes variation, then we receive the highest accuracy rate. The K values should be 3, 5, 7, 9, or $\min(\text{nperclass})-1$ as shown in figure 4.24.
3. If the number of data samples in each class is not large, then the $K = \min(\text{nperclass})-1$ may not be appropriate.
4. We suggest K should be an odd number to prevent prediction of two equal membership values.
5. If we cannot guarantee condition in 1 and 2. We need to find the optimal K empirically.

Nevertheless, the comparison of accuracy rates among 7 sgFKNN algorithms on face recognition against the others is shown in table 4.33.

Table 4.33 The comparison of accuracy rates between our algorithm of face recognition on 7 sgFKNNs with the other's.

Dataset	sgFKNN1	sgFKNN2	sgFKNN3	sgFKNN4	sgFKNN5	sgFKNN6	sgFKNN7	The other's
ORL	99.25%	97.24%	99.00%	88.47%	98.25%	98.25%	97.24%	98% [49]
MIT-CBCL	100.00%	99.75%	99.75%	99.50%	99.55%	99.50%	99.75%	92.7% [46]
Georgia Tech	79.57%	75.60%	75.60%	62.53%	69.07%	66.93%	75.60%	96.6% [50]
FEI	93.85%	90.85%	86.29%	76.71%	83.43%	82.43%	86.29%	96% [51]
JAFFE	100.00%	100.00%	100.00%	100.00%	100.00%	100.00%	100.00%	99.43%[52]
Pain Expression	90.00%	77.72%	82.75%	73.56%	80.01%	79.05%	82.75%	N/A
Senthilkumar	96.25%	96.25%	96.25%	96.25%	96.25%	96.25%	96.25%	83.33%[53]
PICS	95.23%	87.98%	83.41%	74.15%	80.65%	79.69%	83.41%	N/A
Yale	96.98%	85.73%	82.42%	83.03%	81.81%	80.60%	83.63%	86.67%[54]
CMU AMP	100.00%	98.15%	86.13%	76.57%	83.27%	82.28%	86.13%	90.5%[55]

A string grammar fuzzy K -nearest neighbor (sgFKNN) was developed by adjusting the membership value in [8] and [24] and incorporating them into the string grammar K -nearest neighbor. Specifically, the assignment to the test string image to the class can be done by calculating the maximum membership value among the K -nearest neighbor strings. The 7 sgFKNNs were then applied in the face recognition system. Even though we used 1600 symbols in ORL and Yale databases and used 100 symbols for the rest. The results are better than the other previous works on ORL, MIT-CBCL, JAFFE, Senthilkumar, Yale and CMU AMP, and are able to compare to the previous works on FEI. However, the results are worse than that from a previous work on Georgia Tech

database. The reason is possibly that the faces from the dataset of Georgia Tech were captured at different scales and the proposed algorithm was not much expected to be invariant to scaling. The Levenshtein distance could cause some problems which the transformation of the strings used in the calculation of the Levenshtein distance might create a small distance between strings that are actually far apart in the normal sense. This could occur from two different faces with two similar strings. However, it can be suspicious that increasing number of subimages (symbols) could help in this case.

From all experiment, sgFKNN1 to sgFKNN7 are able to compare and better than the previous algorithms in some datasets. However, worse results than the previous methods in some datasets can be seen from the sgFKNN1 to sgFKNN7, because the algorithm's nature is needed to be improved in the future.

4.3.2 Facial expression recognition experiment result

This section demonstrates application of string grammar fuzzy K-nearest neighbor to the facial expression recognition.

We describe 5 public standard databases in “Facial expression recognition using string grammar fuzzy K-nearest neighbor” as follows:

1. JAFFE

Japanese Female Facial Expression (JAFFE) database consists of 213 images of Japanese female facial expressions, where each image corresponds to one of the seven categories of expression, i.e., anger, disgust, fear, happiness, neutral, sadness, and surprise.

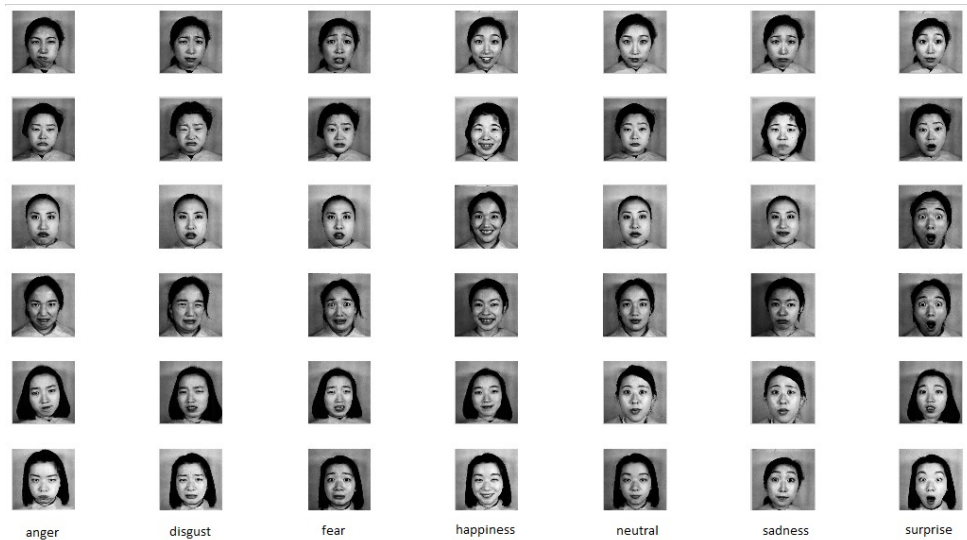


Figure 4.38 Examples of each class in JAFFE Database.

2. Yale

The Yale database contains 165 gray scale images in GIF format from 15 individuals. There are 11 images per subject, one per different facial expression or configuration, i.e., center-light, with glasses, happy, left-light, without glasses, normal, right-light, sad, sleepy, surprised, and wink. There are 3 categories of illumination, i.e., center-light, left-light, and right-light. The facial expression part is divided into 6 expressions, i.e., neutral, happiness, sadness, sleepiness, surprise, and wink. There are 88 images in this part. Since in this paper, we are only interested in facial expression, we use only these 88 facial expression images for this dataset

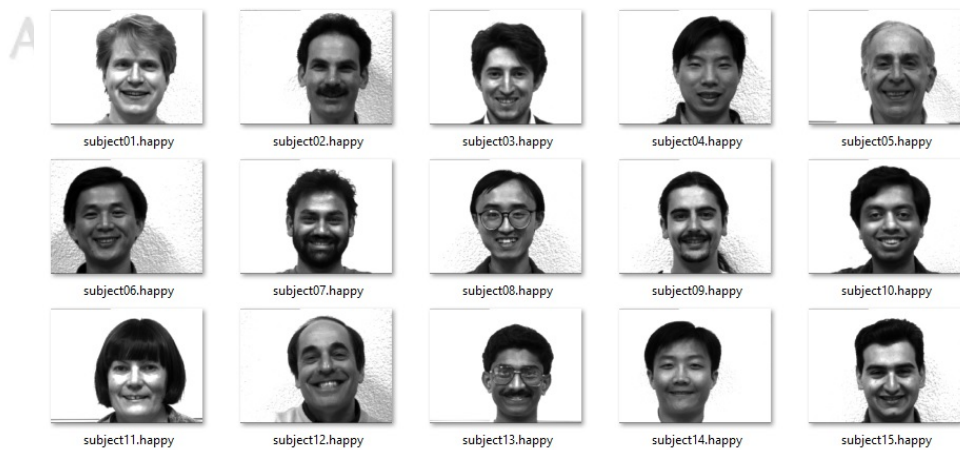


Figure 4.39 Examples of each class in Yale Database.

3. CMU AMP

CMU AMP database, there are 13 subjects in the database, each with 75 images, and all of the face images are collected in the same lighting condition, allowing only human expression changes. There are 5 facial expressions in this dataset, i.e., happy, normal, wink, surprise, and disgust.

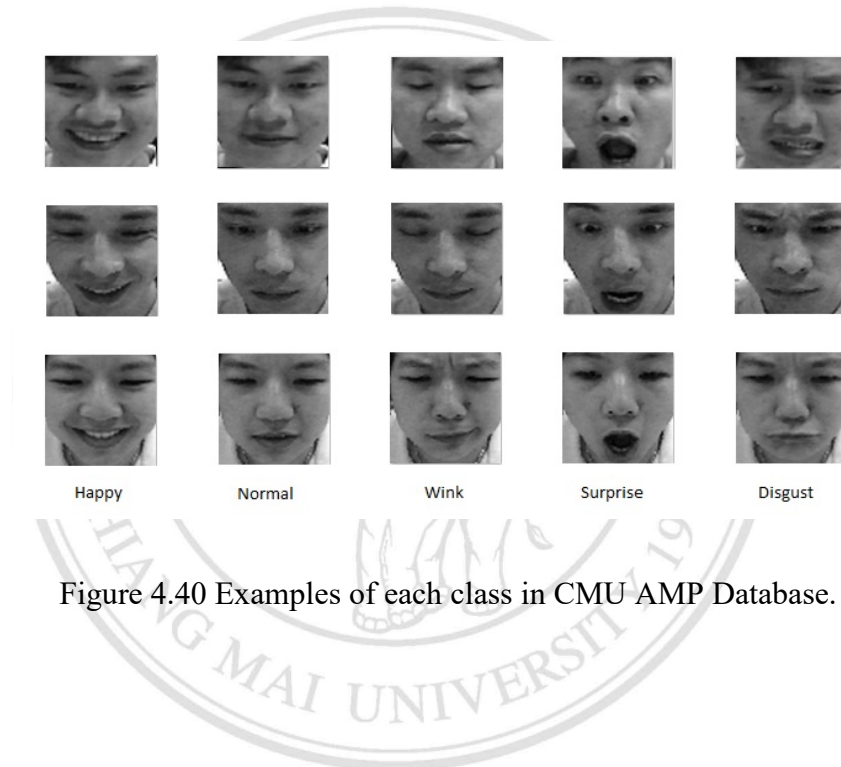


Figure 4.40 Examples of each class in CMU AMP Database.

4. CK+ [56]

Cohn-Kanade (CK) has 593 VDOs sequences across 123 subjects. It consists of 8 categories of facial expressions: 1=anger, 2=contempt, 3=disgust, 4=fear, 5=happy, 6=sadness, 7=surprise, and 8=neutral. Next, CK+ has 327 VDOs sequences. Finally, UNBC-McMaster Shoulder Pain Expression Archive Database has 200 sequences across 25 subjects. The spontaneous expressions of pain from patients with shoulder problems are shown in the image sequences.



Figure 4.41 Examples of each class in CK Database.

5. UNBC [57]

UNBC-McMaster Shoulder Pain Expression Archive Database has 200 sequences across 25 subjects for spontaneous expressions of pain from patients with shoulder problems. However, we want to test this database because we want to check the Glasgow coma scale (GCS) for facial expression of the patients consists of 5 categories, including 1=neutral, 2=happy, 3=wink, 4=mild pain, and 5=pain as shown in figure 4.50. For each category, we selected three clearly seen images per one person in each class which were used as the representatives of the database. This is performed since, for each class, a sequence of images consists of facial expression from normal to pain emotion classes.



Figure 4.42 Example of each class in UNBC Database.

Furthermore, we also developed facial expression recognition based on sgFKNN1 to sgFKNN7 on 5 facial expression recognition datasets which are JAFFE (1600 symbols), Yale (1600 symbols), CMU AMP (1600 symbols), CK+ (327 VDOs) (1600 symbols) and UNBC-McMaster Shoulder Pain Expression Archive Database (100 symbols).

From table 4.37 and 4.39 to 4.45, we show the experimental results from sgFKNN1, sgFKNN2, sgFKNN3, and sgFKNN7 on 5 facial expression recognition datasets when using the K values from 1 to 10 and the minimum number of images for each person minus 1.

Moreover, we show the experimental results from sgFKNN4, sgFKNN5, and sgFKNN6 on 5 facial expression recognition datasets when using the K values from 1 to C . JAFFE has 7 classes, while Yale has 6 classes, CMU AMP has 5 classes, CK+ has 8 classes, and UNBC has 5 classes.

Table 4.34 The experimental results from sgFKNN1 for facial expression recognition datasets.

Dataset	K=1	K=2	K=3	K=4	K=5	K=6	K=7	K=8	K=9	K=10	K=min(nperclass)-1
JAFFE	88.77%	88.77%	89.85%	87.49%	84.21%	82.00%	100%	97.37%	89.67%	87.31%	89.67%
YALE	61.06%	61.06%	61.80%	60.78%	64.57%	63.51%	74.08%	72.85%	69.39%	68.24%	61.80%
CMU_A MP	95.66%	95.66%	96.82%	96.82%	97.93%	97.93%	97.93%	97.93%	96.82%	96.82%	96.82%
CK+ (327 VDOs)	98.79%	98.79%	100%	98.37%	93.14%	91.63%	96.14%	94.57%	96.14%	94.57%	96.14%
UNBC	100%	100%	100%	100%	100%	100%	100%	100%	100%	100%	100%

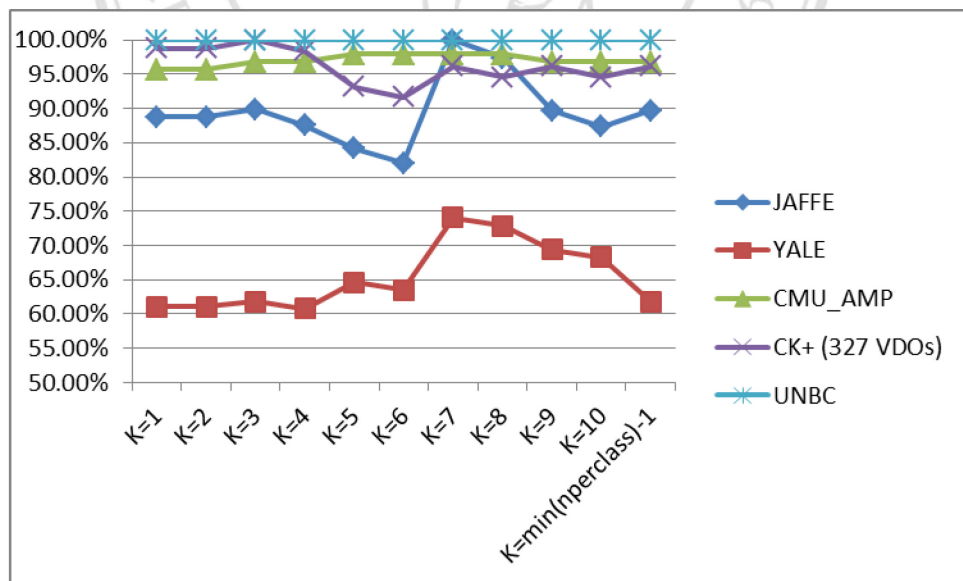


Figure 4.43 Graph of the experimental results from sgFKNN1 for facial expression recognition datasets.

The sgFKNN1 has the best accuracy rate when $K = 7$ for JAFFE database and Yale database, $K = 5, 6, 7, 8$ for CMU AMP database, $K = 3$ for CK+ database and 100% in each K on UNBC database. In JAFFE and YALE database, where

the accuracy rate is less when starting from $K = 1$ and increasing for $K = 3, 5$, and so on until the accuracy reaches the peak and later decreases.

Moreover, the system provided the accuracy rates when using $K = 1, 3, 5, 7, 9$ greater or equal to $K = 2, 4, 6, 8$, and 10, respectively.

Table 4.35 shows the results of the indirect comparison from the Neural-AdaBoost algorithm [58] and the multiple Gabor filter with SR and SVM algorithm [59]. It can be seen that the correct classification rates from [58] and [59] are 96.81% and 89.28%, respectively. Our algorithm is better than the algorithm in [59] but the algorithm in [58] is far better than our algorithm. The reason could be that the algorithm in [58] cropped the whole image into a subimage containing only face area; meanwhile our algorithm does not require such a process.

Table 4.35 Indirect comparison of the results from the JAFFE Database.

Method	Anger	Disgust	Fear	Happiness	Neutral	Sadness	Surprise	Average
Multiple Gabor filter & SR + SVM [59]	90%	95%	80%	90%	90%	80%	100%	89.28%
Neural-AdaBoost [58]	96.10%	99.99%	96.08%	99.72%	92.23%	93.9%	99.87%	96.81%
Our Method	86.67%	89.66%	90.63%	90.32%	90%	83.87%	96.67%	89.67%

All rights reserved

Table 4.36 Confusion matrix of six expression classes of sgFKNN1 in the YALE Database.

		Program output					
		Happy	Normal	Sad	Sleepy	Surprise	Wink
Desired output	Expression						
	Happy	11	1	0	0	0	1
	Normal	1	10	1	2	0	1
	Sad	2	2	8	2	1	1
	Sleepy	0	3	1	8	0	2
	Surprise	1	1	1	0	12	0
Wink	3	3	0	3	0	6	

Table 4.37 Indirect comparison of the results from the Yale Database.

Method	Normal	Sad	Happy	Surprise	Average
Neural-AdaBoost [58]	86.16%	86.79%	97.60%	98.33%	91.52%
Our algorithm	83.33%	61.54%	91.67%	80.00%	78.85%

Table 4.37 shows the indirect comparison between our results with that from the algorithm in [58]. Because the algorithm in [58] only implemented for the dataset in 4 classes, i.e., normal, sad, happy, and surprise, we solely compare our results from these 4 classes. It can be seen that the neural-AdaBoost [58] produced around 96.81% classification rate, while our algorithm produced 89.67%. Even

though, our algorithm does not produce accuracy as high as that from [58], we do not crop any image in advance.

Regarding to the indirect comparison, the system found that our results are comparable with the existing algorithms, though we do not crop the image in advance, whereas most of the existing algorithms do. It is clear that our algorithm provides a good result if there is only a face area in the image.

Table 4.38 Confusion matrix of six expression classes of sgFKNN1 in the CMU AMP Database.

		Program output				
		Expression	Happy	Normal	Wink	Surprise
Desired output	Happy	193	2	1	1	0
	Normal	2	277	3	1	1
	Wink	1	9	172	1	1
	Surprise	0	3	1	172	0
	Disgust	0	1	3	1	129

Table 4.39 Indirect comparison of the results from the CMU AMP Database.

Method	Happy	Disgust	Surprise	Average
Nonlinear decomposable generative model [60]	≈95%	≈98%	100%	N/A
Our algorithm	99.48%	99.23%	100%	99.60%

The indirect comparison between our result and the nonlinear decomposable generative model [60] is shown in table 4.39. From this case, the results from [60]

are from 9 persons with only 3 face expressions, i.e., happy, disgust, and surprise. Only these face expression classes for comparison are considered. Since the 9 persons used in [60] are unknown to us, we report the result from our algorithm implemented on all 13 persons. However, our algorithm gives a better correct classification than the method in [60].

Nevertheless, we implemented sgFKNN1 on more datasets. For example, CK dataset, CK+ dataset and UNBC-McMaster Shoulder Pain Expression Archive Database.

Table 4.40 Confusion matrix in 8 facial expressions of CK database when using 1600 symbols using sgFKNN1 and $K=1$.

		Program output							
		Anger	Contempt	Disgust	fear	Happy	Sadness	Surprise	Neutral
Desired output	Anger	145	1	0	0	1	1	0	2
	Contempt	2	179	1	0	1	1	0	5
	Disgust	0	2	173	0	0	0	0	2
	fear	1	0	0	166	0	0	0	1
	Happy	1	2	1	2	252	0	0	3
	Sadness	0	2	0	0	1	170	1	3
	Surprise	0	1	0	2	1	0	297	2
	Neutral	3	4	2	3	3	5	3	346

Table 4.41 Confusion matrix percentage of 8 expression classes of sgFKNN1 in CK Database.

		Program output							
Expression	Anger	Contempt	Disgust	fear	Happy	Sadness	Surprise	Neutral	
Desired output	Anger	96.67%	0.67%	0%	0%	0.67%	0.67%	0%	1.33%
	Contempt	1.06%	94.71%	0.53%	0%	0.53%	0.53%	0%	2.65%
	Disgust	0%	1.13%	97.74%	0%	0%	0%	0%	1.13%
	fear	0.6%	0%	0%	98.81%	0%	0%	0%	0.6%
	Happy	0.38%	0.77%	0.38%	0.77%	96.55%	0%	0%	1.15%
	Sadness	0%	1.13%	0%	0%	0.56%	96.05%	0.56%	1.69%
	Surprise	0%	0.33%	0%	0.66%	0.33%	0%	98.02%	0.66%
	Neutral	0.81%	1.08%	0.54%	0.81%	0.81%	1.36%	0.81%	93.77%

ลิขสิทธิ์มหาวิทยาลัยเชียงใหม่
 Copyright© by Chiang Mai University
 All rights reserved

Table 4.42 Confusion matrix in 5 facial expressions of UNBC database when using 100 symbols and $K=1$.

		Program output				
		Neutral	Happy	Wink	Mild pain	Pain
Desired output	Expression	Neutral	Happy	Wink	Mild pain	Pain
	Neutral	75	0	0	0	0
	Happy	0	30	0	0	0
	Wink	0	0	54	0	0
	Mild pain	0	0	0	12	0
Pain	0	0	0	0	24	

Table 4.43 Confusion matrix percentage of 5 expression classes of sgFKNN1 in UNBC database.

		Program output				
		Neutral	Happy	Wink	Mild pain	Pain
Desired output	Expression	Neutral	Happy	Wink	Mild pain	Pain
	Neutral	100%	0%	0%	0%	0%
	Happy	0%	100%	0%	0%	0%
	Wink	0%	0%	100%	0%	0%
	Mild pain	0%	0%	0%	100%	0%
Pain	0%	0%	0%	0%	100%	

Table 4.44 Confusion matrix of seven expression classes of sgFKNN1 in the JAFFE Database.

		Program output						
		Anger	Disgust	Fear	Happiness	Neutral	Sadness	Surprise
Desired output	Expression	Anger	Disgust	Fear	Happiness	Neutral	Sadness	Surprise
	Anger	26	2	0	0	1	1	0
	Disgust	2	26	1	0	0	0	0
	Fear	0	1	29	0	0	0	2
	Happiness	0	0	0	28	2	0	1
	Neutral	1	0	0	0	27	0	2
	Sadness	1	1	0	0	2	26	1
Surprise	0	0	0	0	0	1	29	

Table 4.44 shows the result in term of confusion matrix from the JAFFE dataset. The misclassification might be the reason that one facial expression class from some subjects is almost similar to the other class. For instance, figure 4.55 to 4.57 show the membership value of subject number 6 in anger, disgust, fear, happiness, neutral, sadness, and surprise classes (all 28 nearest neighbors as shown in figure 4.56) are 0.118, 0.179, 0.128, 0.134, 0.170, 0.142, and 0.128, respectively. When the desired class is anger, the decision from the algorithm is disgust. It can be seen from the face of subject 6 and all 28 nearest neighbors that this face is very similar to neutral or disgust emotion. Therefore, the membership values of this subject to these 2 classes are higher than the other classes.



Figure 4.44 Subject number 6 who is misclassified into disgust class.

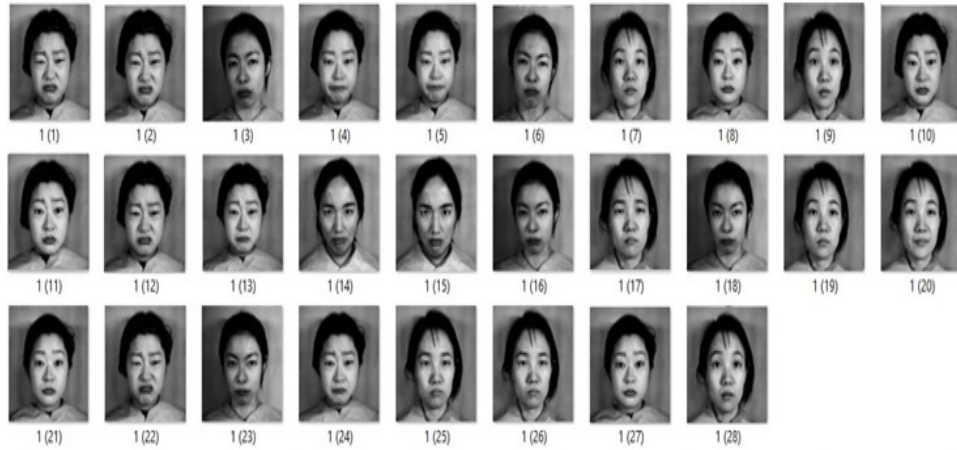


Figure 4.45 28-nearest neighbor of face expression in figure 4.44.

	1	2	3	4	5	6	7
1	0.247330187	0.15088335	0.197025874	0.177444067	0.065497335	0.034224104	0.127615084
2	0.200136793	0.16484339	0.09263898	0.128904188	0.138194244	0.141528976	0.133753428
3	0.236172149	0.161684086	0.137413977	0.162617472	0.084205612	0.100873987	0.117032718
4	0.229007157	0.209818441	0.112789729	0.144498127	0.075158148	0.148378435	0.080349963
5	0.219884676	0.200510639	0.130133827	0.111830034	0.087960138	0.17075991	0.078920776
6	0.118569174	0.178745782	0.127878675	0.134127489	0.170104062	0.141731693	0.128843114
7	0.22941637	0.086560871	0.074611598	0.166085051	0.263935901	0.146273117	0.033117092
8	0.239701323	0.073588725	0.086827301	0.154067821	0.197812402	0.231402916	0.016599511
9	0.221727012	0.04108597	0.087535266	0.150102512	0.216005326	0.247908852	0.035635063
10	0.244914313	0.192019955	0.143593233	0.07725839	0.105089599	0.119006779	0.11811773
11	0.264575936	0.187318109	0.138394375	0.081129885	0.106228579	0.130823664	0.091529452
12	0.132505123	0.195470559	0.146367639	0.094828434	0.100060734	0.135558488	0.191209023
13	0.37424317	0.242105289	0.077580266	0.099063767	0.057357062	0.163291489	0.026358957
14	0.431721937	0.280641155	0.025735252	0.053667001	0.055965128	0.152269527	0
15	0.383581223	0.301113067	0.053171188	0.026515926	0.056085786	0.152174589	0.027358621
16	0.223699148	0.157007349	0.115249385	0.130549255	0.077475961	0.202431011	0.093587892
17	0.292812637	0.206619071	0.056159686	0.028195631	0.080434529	0.242045259	0.093733186
18	0.252538878	0.123899405	0.09573199	0.052187003	0.129097063	0.251678793	0.094866868
19	0.298497037	0.151494773	0.07858992	0.1105106	0.174501921	0.110517038	0.07588881
20	0.319239725	0.144540782	0.105826212	0.098731347	0.148337774	0.110349791	0.072984369
21	0.314591578	0.129833517	0.130928516	0.091674699	0.11048875	0.112895082	0.109587858
22	0.247437439	0.161506081	0.121901304	0.085023364	0.16352157	0.173039528	0.047570713
23	0.269566014	0.186695692	0.112769304	0.133566631	0.134565333	0.136954176	0.025882851
24	0.26294467	0.197562843	0.102327427	0.133496503	0.131936611	0.146886978	0.024844969
25	0.236441217	0.173145267	0.106562939	0.110057256	0.084972411	0.193790505	0.095030404
26	0.227004631	0.191023785	0.137137051	0.114284275	0.092533464	0.147325099	0.090691695
27	0.263300809	0.20909581	0.107518078	0.098869935	0.078260111	0.146543898	0.09641176
28	0.482282665	0.137222187	0.026049385	0.102071787	0.082052129	0.082968961	0.087352886
29	0.453120802	0.194278246	0.028703344	0.108564868	0.077141862	0.08546717	0.052723708
30	0.465657876	0.160508603	0.026977664	0.101647392	0.085658679	0.081408148	0.078141637

Figure 4.46 The membership values when using $K=28$ on JAFFE database.

Nevertheless, figure 4.47 shows that the membership values decrease when the K values increase.

The result from the Yale dataset shows that the misclassification reason could be that the HoG was utilized in the string generation process. The HoG could create a string from one emotion that is similar to the other emotions because of background or a person's shadow in the background.

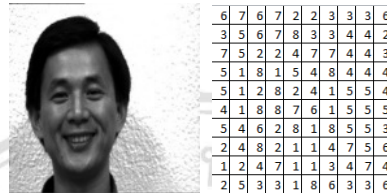


Figure 4.47 Original image of a person in happy class with its string.

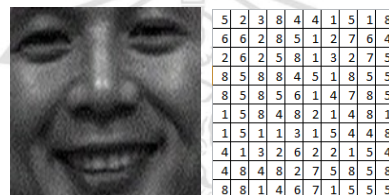


Figure 4.48 Cropped image of (Figure 4.47) with its string.

Figure 4.47 and figure 4.48 show the samples of the original image with the string created versus its corresponding cropped image with the string created. It can be seen that strings from both cases are different. Since we do not crop the image, our algorithm assigns this image to a wrong emotion class.

It is found that the misclassifications are from similar facial expressions among different facial expression classes. The similar strings of images from different classes might be produced by HoG because of the background and the shadow of the person in the background of the image.

For the result from our algorithm on the CMU AMP dataset, the misclassification happens when the subjects have similar faces in different classes. For instance, the subject 85 is supposed to be in the happy class but is classified into the normal class. The reason is that the membership values of this subject in the happy and normal classes are 0.41 and 0.58, respectively. This subject in the happy face and 5-nearest neighbors are shown in Figure 4.49.



(a)

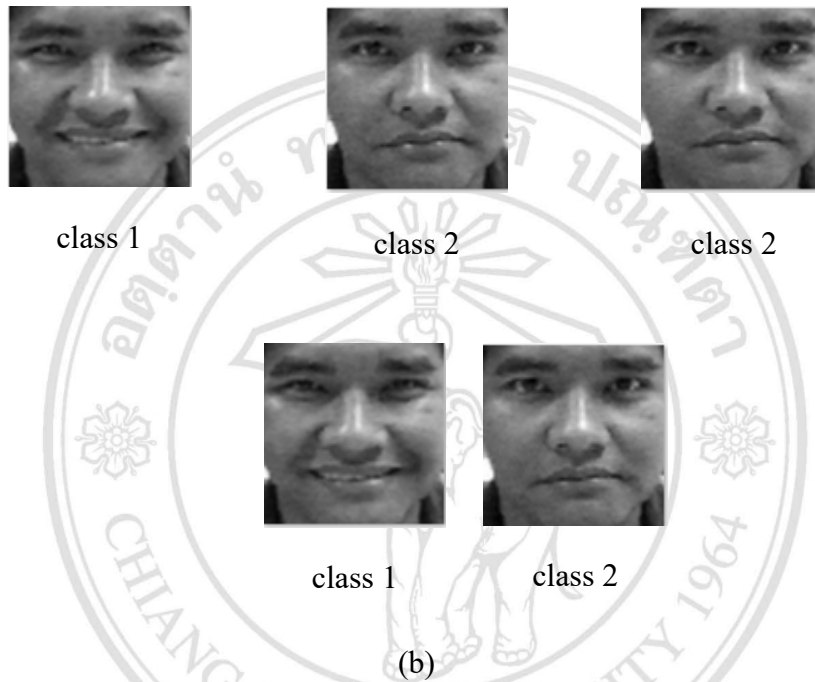


Figure 4.49 (a) subject 85 in happy face and (b) 5-nearest neighbors.

It can be seen that, from all 5 experiments, for some datasets that have images with background, sgFKNN1 gives comparable results with the existing algorithms, even though those algorithms have a cropping preprocessing step. For the dataset that comes with only faces without background (CMU AMP), our algorithm outperforms the existing algorithm significantly.

Table 4.45 The experimental results from sgFKNN2 for facial expression recognition datasets.

Dataset	K=1	K=2	K=3	K=4	K=5	K=6	K=7	K=8	K=9	K=10	K=min(nperclass)-1
JAFFE	87.52%	87.52%	88.58%	86.09%	82.33%	80.02%	99.53%	96.73%	88.40%	86.02%	88.40%
YALE	60.20%	60.20%	60.93%	59.92%	63.67%	62.61%	73.04%	71.83%	63.67%	62.61%	60.93%
CMU AMP	88.87%	88.87%	89.95%	89.95%	90.90%	90.90%	90.68%	90.68%	89.95%	89.95%	89.95%
CK+ (327 VDOs)	95.80%	95.80%	96.97%	95.80%	94.23%	93.09%	93.23%	92.10%	93.23%	92.10%	93.23%
UNBC	100%	100%	100%	100%	100%	100%	100%	100%	100%	100%	100%

From the experimental results, we found that the best accuracy rates on JAFFE and YALE are 99.53% and 73.04%, respectively, when using $K = 7$. For CMU AMP database is 90.90%, when using $K = 5, 6$. For CK+ databases is 96.97% when using $K = 3$. For UNBC database is 100% for every K .

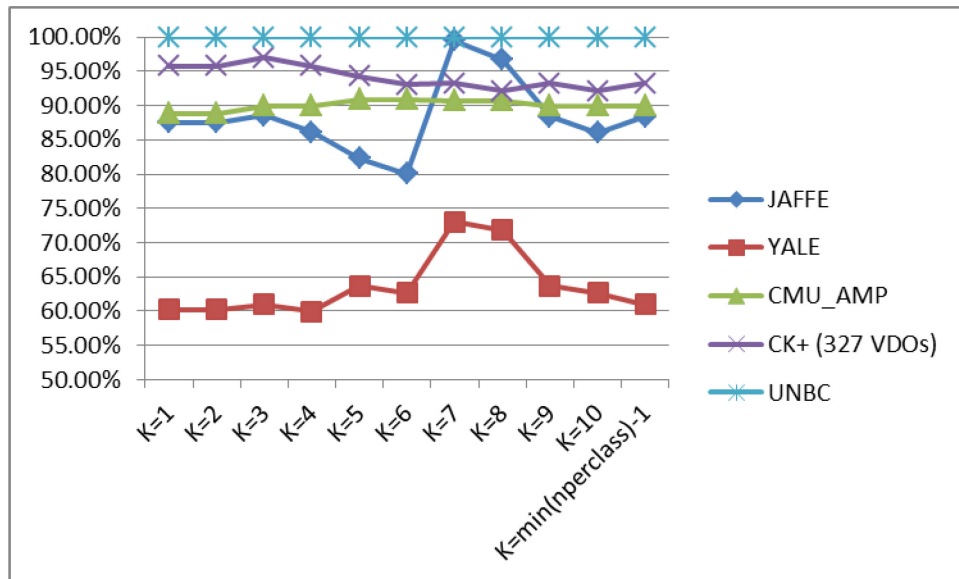


Figure 4.50 The accuracy rate of sgFKNN2 on 5 standard datasets

In addition, the experimental results show that the accuracy rate is similar to sgFKNN1 and has same pattern when vary on K values but a little bit less than those of sgFKNN1.

For each image in the classes of every datasets, if we generate string in same of process then we have the same string in same image on 7 sgFKNNs and same distance in sgFKNN1, sgFKNN2, sgFKNN3 and sgFKNN7 as group 1 and same distance in sgFKNN4, sgFKNN5 and sgFKNN6 as group 2. From this reason, the misclassifications depend on each equation in each sgFKNNs.

The misclassifications from sgFKNN2 algorithm are similar to sgFKNN1. They show that if we use different datasets, then accuracy rate surely differ. In converse, if using the same datasets, it will have the same pattern of accuracy rate, but different algorithm which variable on K values. If we use $K=2$ or even values, then the membership values will have an equal opportunity which make the accuracy rate dropped.

Table 4.46 The experimental results from sgFKNN3 for facial expression recognition datasets.

Dataset	K=1	K=2	K=3	K=4	K=5	K=6	K=7	K=8	K=9	K=10	K=min(nperclass)-1
JAFFE	87.52%	87.52%	88.58%	86.09%	82.33%	80.02%	99.53%	96.73%	88.40%	86.02%	88.40%
YALE	60.20%	60.20%	60.93%	59.92%	63.67%	62.61%	73.04%	71.83%	63.67%	62.61%	60.93%
CMU AMP	88.87%	88.87%	89.95%	89.95%	90.90%	90.90%	90.68%	90.68%	89.95%	89.95%	89.95%
CK+ (327 VDOs)	95.35%	95.35%	96.52%	95.36%	93.79%	93.23%	92.79%	91.23%	92.79%	91.23%	92.79%
UNBC	100%	100%	100%	100%	100%	100%	100%	100%	100%	100%	100%

From the experimental results, we found that the best accuracy rates on JAFFE and YALE are 99.53% and 73.04%, respectively, when using $K = 7$. For CMU AMP database is 90.90%, when using $K = 5, 6$. For CK+ databases is 96.52% when using $K = 3$. For UNBC database is 100% for every K .

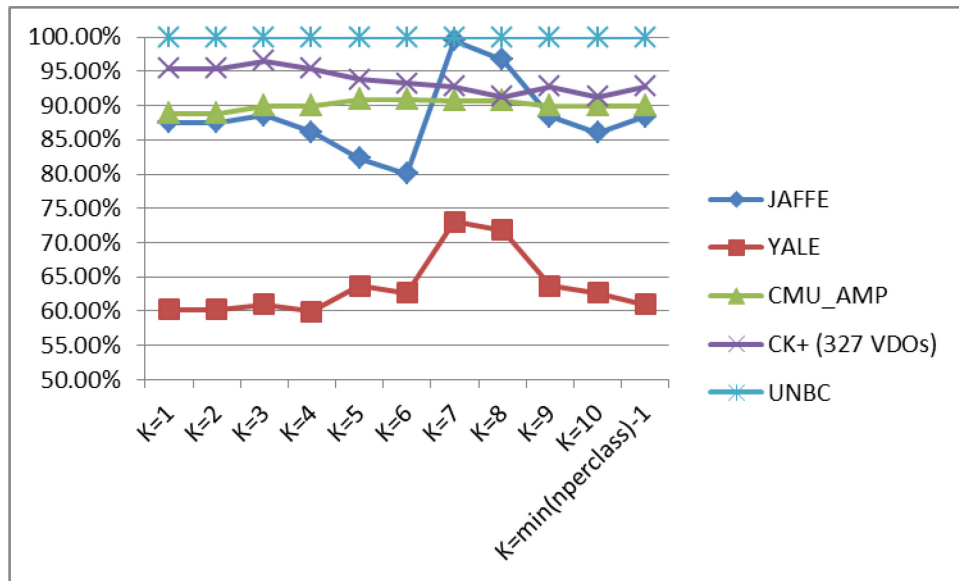


Figure 4.51 The accuracy rate of sgFKNN3 on 5 standard datasets

In addition, the experimental results show that the accuracy rate is similar to sgFKNN1 and has same pattern when vary on K values but a little bit less than those of sgFKNN1.

The misclassifications from sgFKNN3 algorithm are similar to sgFKNN1 and sgFKNN2.

Although we use different datasets which make the accuracy rate also changed. However using the same dataset will have the same pattern of accuracy rate, although different algorithms which variable on K values were used. If we use $K=2$ or even values, then the membership values will have an equal opportunity which make the accuracy rate dropped.

Table 4.47 The experimental results from sgFKNN4 for facial expression recognition datasets.

Dataset	K=1	K=2	K=3	K=4	K=5	K=6	K=7	K=8	K=9	K=10	K=C
JAFFE	74.38%	73.16%	75.28%	74.38%	71.13%	70.28%	75.13%	N/A	N/A	N/A	75.13%
YALE	51.16%	51.31%	51.78%	50.92%	54.10%	51.78%	N/A	N/A	N/A	N/A	51.78%
CMU_A MP	88.47%	87.02%	89.54%	88.08%	89.54%	N/A	N/A	N/A	N/A	N/A	89.54%
CK+ (327 VDOs)	91.03%	89.54%	92.14%	91.03%	89.53%	89.00%	88.58%	87.09%	N/A	N/A	87.09%
UNBC	97.95%	97.95%	97.95%	97.95%	97.95%	N/A	N/A	N/A	N/A	N/A	97.95%

Note. We have N/A in some results because the dataset has number of classes less than number of K .

From the experimental results, we found that the best accuracy rates on JAFFE is 75.28% when using $K = 3$. For YALE database is 54.10% when using $K = 5$ For CMU AMP database is 89.54%, when using $K = 3, 5$. For CK+ databases is 92.14% when using $K = 3$. For UNBC database is 97.95% for every K .

Copyright © by Chiang Mai University
All rights reserved

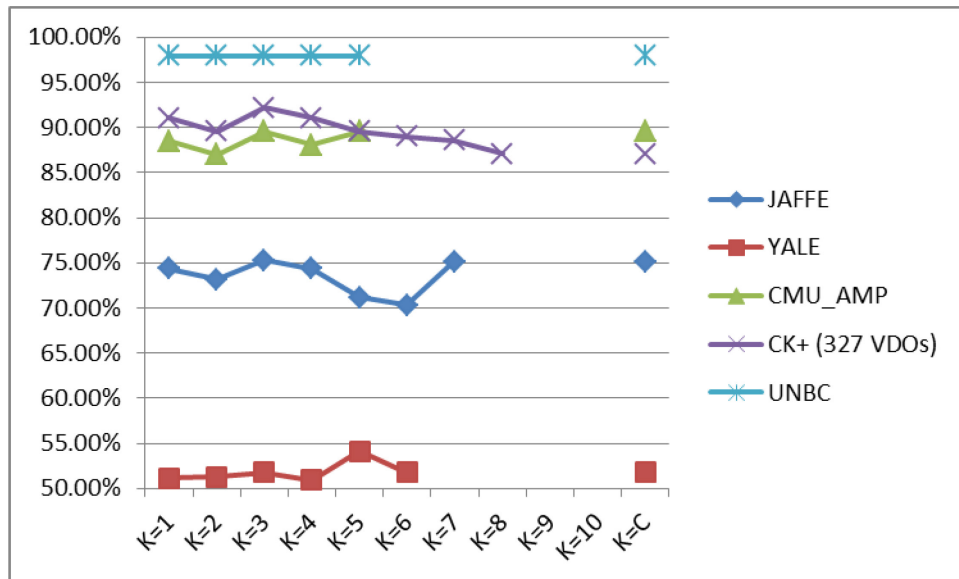


Figure 4.52 The accuracy rate of sgFKNN4 on 5 standard datasets

Next, the experimental results show that the accuracy rate is similar to sgFKNN1 and has the same pattern when vary on K values but a little bit less than those of sgFKNN1.

The misclassifications from sgFKNN4 algorithm are similar to sgFKNN1. They show that although we use different datasets which make the accuracy rate also changed. However using the same dataset will have the same pattern of accuracy rate, although different algorithms which variable on K values were used. If we use $K=2$ or even values, then the membership values will have an equal opportunity which make the accuracy rate dropped.

Furthermore, we implemented the multi-prototypes on sgFKNN4. We set the number of prototypes on each dataset to about 1%, 5%, 10%, 15%, and 20% of the minimum number of images for each class. For JAFFE dataset, the minimum number of images for each class is 30; we set the number of prototypes to be 1, 3, 5, 7, and 9. For Yale and UNBC datasets, the minimum numbers of images for each class is 28 and 20, respectively; we set the number of prototypes to be 1, 2, 3, 4, and 5. For CMU_AMP dataset, the minimum number of images for each class is 200; we set the number of prototypes to be 1, 10, 20, 30, and 40. For CK+ dataset, the minimum number of images for each class is 100; we set the number of prototypes to be 1, 5, 10, 15, and 20.

Table 4.48 The experimental results from sgFKNN4 for facial expression datasets when using the different number of prototypes.

Dataset	Number of prototypes	1	3	5	7	9
JAFPE	$K=1$	74.38%	75.64%	76.75%	75.41%	74.34%
	$K=2$	73.16%	74.40%	75.49%	74.17%	73.12%
	$K=3$	75.28%	76.56%	77.68%	76.32%	75.24%
	$K=4$	74.38%	75.64%	76.75%	75.41%	74.34%
	$K=5$	71.13%	72.34%	73.40%	72.11%	71.09%
	$K=6$	70.28%	71.47%	72.52%	71.25%	70.24%
	$K=7$	75.13%	76.41%	77.53%	76.17%	75.09%
	$K=8$	N/A	N/A	N/A	N/A	N/A
	$K=9$	N/A	N/A	N/A	N/A	N/A
	$K=10$	N/A	N/A	N/A	N/A	N/A
	$K=C$	75.13%	76.41%	77.53%	76.17%	75.09%
YALE	Number of prototypes	1	2	3	4	5
	$K=1$	51.16%	52.85%	53.65%	54.51%	54.68%
	$K=2$	51.31%	53.01%	53.80%	54.67%	54.84%
	$K=3$	51.78%	53.49%	54.30%	55.17%	55.34%
	$K=4$	50.92%	52.60%	53.40%	54.25%	54.42%
	$K=5$	54.10%	55.89%	56.73%	57.64%	57.82%
	$K=6$	51.78%	53.49%	54.30%	55.17%	55.34%
	$K=7$	N/A	N/A	N/A	N/A	N/A
	$K=8$	N/A	N/A	N/A	N/A	N/A
	$K=9$	N/A	N/A	N/A	N/A	N/A
	$K=10$	N/A	N/A	N/A	N/A	N/A
$K=C$	51.78%	53.49%	54.30%	55.17%	55.34%	
CMU AMP	Number of prototypes	1	10	20	30	40
	$K=1$	88.47%	88.66%	88.80%	88.80%	88.66%
	$K=2$	87.02%	87.20%	87.34%	87.34%	87.20%
	$K=3$	89.54%	89.73%	89.87%	89.87%	89.73%
	$K=4$	88.08%	88.27%	88.40%	88.40%	88.27%
	$K=5$	89.54%	89.73%	89.87%	89.87%	89.73%
	$K=6$	N/A	N/A	N/A	N/A	N/A
	$K=7$	N/A	N/A	N/A	N/A	N/A
	$K=8$	N/A	N/A	N/A	N/A	N/A
	$K=9$	N/A	N/A	N/A	N/A	N/A
	$K=10$	N/A	N/A	N/A	N/A	N/A
$K=C$	89.54%	89.73%	89.87%	89.87%	89.73%	

Note. We have N/A in some results because the dataset has number of classes less than number of K .

Table 4.48 The experimental results from sgFKNN4 for facial expression datasets when using the different number of prototypes (Cont.).

Dataset	Number of prototypes	1	5	10	15	20
CK+	$K=1$	91.03%	91.67%	91.67%	91.43%	91.00%
	$K=2$	89.54%	90.17%	90.17%	89.93%	89.51%
	$K=3$	92.14%	92.79%	92.79%	92.54%	92.11%
	$K=4$	91.03%	91.67%	91.67%	91.43%	91.00%
	$K=5$	89.53%	90.16%	90.16%	89.92%	89.50%
	$K=6$	89.00%	89.63%	89.63%	89.39%	88.97%
	$K=7$	88.58%	89.20%	89.20%	88.96%	88.55%
	$K=8$	87.09%	87.70%	87.70%	87.47%	87.06%
	$K=9$	N/A	N/A	N/A	N/A	N/A
	$K=10$	N/A	N/A	N/A	N/A	N/A
	$K=C$	87.09%	87.70%	87.70%	87.47%	87.06%
	UNBC	Number of prototypes	1	2	3	4
$K=1$		97.95%	97.95%	97.43%	97.43%	96.97%
$K=2$		97.95%	97.95%	97.43%	97.43%	96.97%
$K=3$		97.95%	97.95%	97.43%	97.43%	96.97%
$K=4$		97.95%	97.95%	97.43%	97.43%	96.97%
$K=5$		97.95%	97.95%	97.43%	97.43%	96.97%
$K=6$		N/A	N/A	N/A	N/A	N/A
$K=7$		N/A	N/A	N/A	N/A	N/A
$K=8$		N/A	N/A	N/A	N/A	N/A
$K=9$		N/A	N/A	N/A	N/A	N/A
$K=10$		N/A	N/A	N/A	N/A	N/A
$K=C$		97.95%	97.95%	97.43%	97.43%	96.97%

Note. We have N/A in some results because the dataset has number of classes less than number of K .

The experiment results from sgFKNN4 for facial expression datasets when using the different number of prototypes. We found that the best accuracy rate on JAFFE, YALE, CMU_AMP, CK+ and UNBC dataset are 77.68%, 57.82%, 89.87%, 92.79% and 97.95%, respectively, when using $K = 3$ and the number of prototypes is 5 for JAFFE database, $K = 5$ and the number of prototypes is 5 for YALE database, $K = 3, 5$ and the number of prototypes are 20, 30 for CMU_AMP database, $K = 3$ and the number of prototypes are 5, 10 for CK+ database, $K =$ every values and the number of prototypes are 1 and 2 for UNBC database.

The number of prototypes which provides the highest accuracy rate is about 5% to 15% of the number of samples in each class. However, if the number of prototypes is more

than the optimal value, then accuracy will be dropped because some prototypes in one class may be outliers.

Table 4.49 The experimental results from sgFKNN5 for facial expression recognition datasets.

Dataset	$K=1$	$K=2$	$K=3$	$K=4$	$K=5$	$K=6$	$K=7$	$K=8$	$K=9$	$K=10$	$K=C$
JAFFE	77.51%	76.24%	78.45%	74.65%	71.39%	67.93%	78.29%	N/A	N/A	N/A	78.29%
YALE	53.31%	52.43%	53.96%	53.07%	56.38%	53.96%	N/A	N/A	N/A	N/A	53.96%
CMU_A MP	88.57%	87.11%	89.64%	88.17%	89.64%	N/A	N/A	N/A	N/A	N/A	89.64%
CK+ (327 VDOs)	91.69%	90.19%	92.81%	91.69%	90.18%	89.65%	89.23%	87.72%	N/A	N/A	87.72%
UNBC	100%	100%	100%	100%	100%	N/A	N/A	N/A	N/A	N/A	100%

Note. We have N/A in some results because the dataset has number of classes less than number of K .

From the experimental results, we found that the best accuracy rates on JAFFE is 78.45% when using $K = 3$. For YALE database is 56.38% when using $K = 5$ For CMU AMP database is 89.64%, when using $K = 3, 5$. For CK+ databases is 92.81% when using $K = 3$. For UNBC database is 100% for every K .

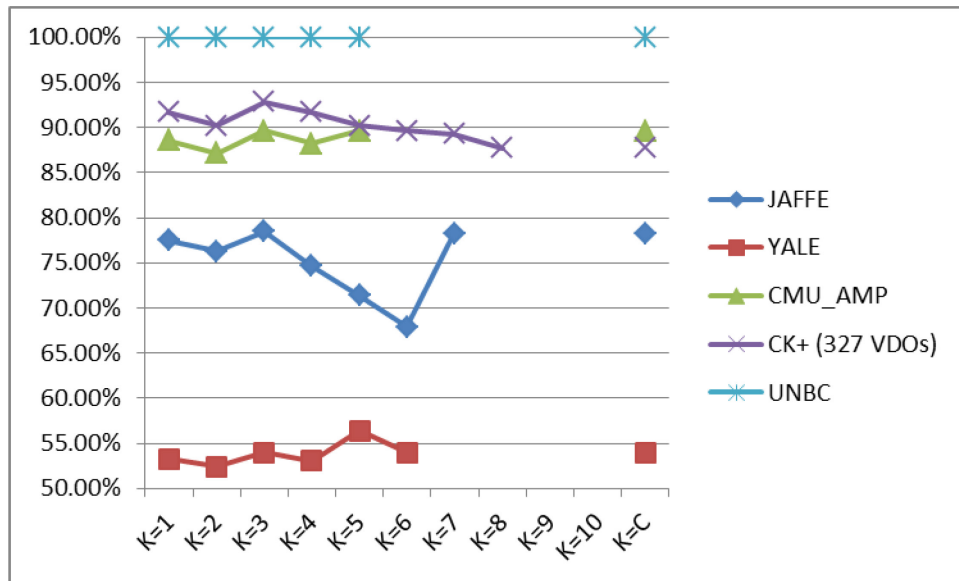


Figure 4.53 The accuracy rate of sgFKNN5 on 5 standard datasets

In addition, the experimental results show that the accuracy rate is similar to sgFKNN1 and has same pattern when vary on K values but a little bit less than those of sgFKNN1.

The misclassifications from sgFKNN4 algorithm are similar to sgFKNN1. They show that although we use different dataset which make accuracy rate also change. However using the same dataset will have the same pattern of accuracy rate, although using the different algorithm which variable on K values. If we use $K=2$ or even values, then the membership values will have an equal opportunity which make the accuracy rate dropped.

Nevertheless, we implemented the multi-prototypes on sgFKNN5. We set the number of prototypes on each dataset to about 1%, 5%, 10%, 15%, and 20% of the minimum number of images for each class. For JAFFE dataset, the minimum number of images for each class is 30; we set the number of prototypes to be 1, 3, 5, 7, and 9. For Yale and UNBC datasets, the minimum numbers of images for each class is 28 and 20, respectively; we set the number of prototypes to be 1, 2, 3, 4, and 5. For CMU_AMP dataset, the minimum number of images for each class is 200; we set the number of prototypes to be 1, 10, 20, 30, and 40. For CK+ dataset, the minimum number of images for each class is 100; we set the number of prototypes to be 1, 5, 10, 15, and 20.

Table 4.50 The experimental results from sgFKNN5 for facial expression datasets when using the different number of prototypes.

Dataset	Number of prototypes	1	3	5	7	9
JAFPE	$K=1$	77.51%	78.83%	79.98%	78.58%	77.47%
	$K=2$	76.24%	77.53%	78.67%	77.29%	76.20%
	$K=3$	78.45%	79.78%	80.95%	79.53%	78.41%
	$K=4$	74.65%	78.83%	79.98%	78.58%	77.47%
	$K=5$	71.39%	75.38%	76.49%	75.15%	74.09%
	$K=6$	67.93%	74.48%	75.57%	74.25%	73.20%
	$K=7$	78.29%	79.62%	80.79%	79.37%	78.25%
	$K=8$	N/A	N/A	N/A	N/A	N/A
	$K=9$	N/A	N/A	N/A	N/A	N/A
	$K=10$	N/A	N/A	N/A	N/A	N/A
	$K=C$	78.29%	79.62%	80.79%	79.37%	78.25%
YALE	Number of prototypes	1	2	3	4	5
	$K=1$	53.31%	55.08%	55.91%	56.81%	56.99%
	$K=2$	52.43%	55.25%	56.07%	56.97%	57.15%
	$K=3$	53.96%	55.75%	56.58%	57.49%	57.68%
	$K=4$	53.07%	54.83%	55.64%	56.54%	56.72%
	$K=5$	56.38%	58.25%	59.12%	60.07%	60.26%
	$K=6$	53.96%	55.75%	56.58%	57.49%	57.68%
	$K=7$	N/A	N/A	N/A	N/A	N/A
	$K=8$	N/A	N/A	N/A	N/A	N/A
	$K=9$	N/A	N/A	N/A	N/A	N/A
	$K=10$	N/A	N/A	N/A	N/A	N/A
$K=C$	53.96%	55.75%	56.58%	57.49%	57.68%	
CMU AMP	Number of prototypes	1	10	20	30	40
	$K=1$	88.57%	88.76%	88.89%	88.89%	88.76%
	$K=2$	87.11%	87.30%	87.44%	87.44%	87.30%
	$K=3$	89.64%	89.83%	89.97%	89.97%	89.83%
	$K=4$	88.17%	88.37%	88.50%	88.50%	88.37%
	$K=5$	89.64%	89.83%	89.97%	89.97%	89.83%
	$K=6$	N/A	N/A	N/A	N/A	N/A
	$K=7$	N/A	N/A	N/A	N/A	N/A
	$K=8$	N/A	N/A	N/A	N/A	N/A
	$K=9$	N/A	N/A	N/A	N/A	N/A
	$K=10$	N/A	N/A	N/A	N/A	N/A
$K=C$	89.64%	89.83%	89.97%	89.97%	89.83%	

Note. We have N/A in some results because the dataset has number of classes less than number of K .

Table 4.50 The experimental results from sgFKNN5 for facial expression datasets when using the different number of prototypes (Cont.).

Dataset	Number of prototypes	1	5	10	15	20
CK+	$K=1$	91.69%	92.33%	92.33%	92.09%	91.66%
	$K=2$	90.19%	90.82%	90.82%	90.58%	90.16%
	$K=3$	92.81%	93.46%	93.46%	93.21%	92.78%
	$K=4$	91.69%	92.33%	92.33%	92.09%	91.66%
	$K=5$	90.18%	90.81%	90.81%	90.57%	90.15%
	$K=6$	89.65%	90.28%	90.28%	90.03%	89.62%
	$K=7$	89.23%	89.85%	89.85%	89.61%	89.20%
	$K=8$	87.72%	88.34%	88.34%	88.10%	87.69%
	$K=9$	N/A	N/A	N/A	N/A	N/A
	$K=10$	N/A	N/A	N/A	N/A	N/A
	$K=C$	87.72%	88.34%	88.34%	88.10%	87.69%
	UNBC	Number of prototypes	1	2	3	4
$K=1$		100%	100%	99.47%	99.47%	99%
$K=2$		100%	100%	99.47%	99.47%	99%
$K=3$		100%	100%	99.47%	99.47%	99%
$K=4$		100%	100%	99.47%	99.47%	99%
$K=5$		100%	100%	99.47%	99.47%	99%
$K=6$		N/A	N/A	N/A	N/A	N/A
$K=7$		N/A	N/A	N/A	N/A	N/A
$K=8$		N/A	N/A	N/A	N/A	N/A
$K=9$		N/A	N/A	N/A	N/A	N/A
$K=10$		N/A	N/A	N/A	N/A	N/A
$K=C$		100%	100%	99.47%	99.47%	99%

Note. We have N/A in some results because the dataset has number of classes less than number of K .

The experiment results from sgFKNN5 for facial expression datasets when using the different number of prototypes. We found that the best accuracy rate on JAFFE, YALE, CMU_AMP, CK+ and UNBC dataset are 80.95%, 60.26%, 89.97%, 93.46% and 100.00%, respectively, when using $K = 3$ and number of prototypes is 5 for JAFFE database, $K = 5$ and number of prototypes is 5 for YALE database, $K = 3, 5$ and number of prototypes are 20, 30 for CMU_AMP database, $K = 3$ and number of prototypes are 5 and 10 for CK+ database, $K =$ every values and number of prototypes are 1 and 2 for UNBC database.

The number of prototypes which provides the highest accuracy rate is about 5% to 15% of the number of samples in each class. However, if the number of prototypes is more

than the optimal value, then accuracy will be dropped because some prototypes in one class may be outliers.

Table 4.51 The experimental results from sgFKNN6 for facial expression recognition datasets.

Dataset	$K=1$	$K=2$	$K=3$	$K=4$	$K=5$	$K=6$	$K=7$	$K=8$	$K=9$	$K=10$	$K=C$
JAFFE	70.01%	68.86%	70.86%	69.16%	62.31%	57.30%	70.72%	N/A	N/A	N/A	70.72%
YALE	48.16%	47.36%	48.74%	44.82%	54.10%	48.74%	N/A	N/A	N/A	N/A	48.74%
CMU_A MP	88.66%	87.21%	89.74%	88.26%	89.74%	N/A	N/A	N/A	N/A	N/A	89.74%
CK+ (327 VDOs)	90.92%	89.43%	92.03%	91.36%	90.84%	90.30%	88.48%	87.72%	N/A	N/A	87.72%
UNBC	100%	100%	100%	100%	100%	N/A	N/A	N/A	N/A	N/A	100%

Note. We have N/A in some results because the dataset has number of classes less than number of K .

From the experimental results, we found that the best accuracy rates on JAFFE is 70.86% when using $K = 3$. For YALE database is 54.10% when using $K = 5$ For CMU AMP database is 89.74%, when using $K = 3, 5$. For CK+ databases is 92.03% when using $K = 3$. For UNBC database is 100% for every K .

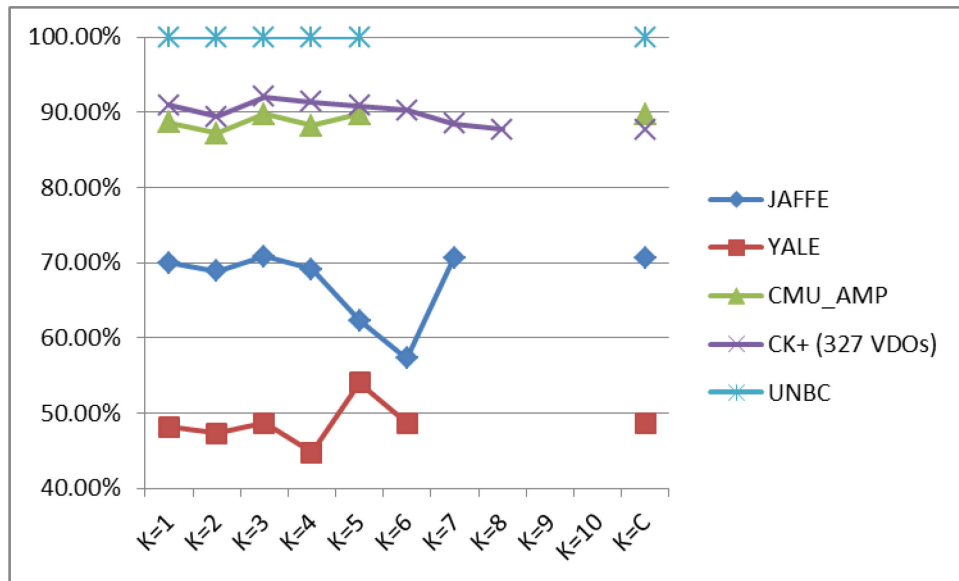


Figure 4.54 The accuracy rate of sgFKNN6 on 5 standard datasets

In addition, the experimental results show that the accuracy rate is similar to sgFKNN1 and has same pattern when vary on K values but a little bit less than those of sgFKNN1.

The misclassifications from sgFKNN6 algorithm are similar to sgFKNN1. They show that although we use different dataset which make accuracy rate also change. However using the same dataset will have the same pattern of accuracy rate, although using the different algorithm which variable on K values. If we use $K=2$ or even values, then the membership values will have an equal opportunity which make the accuracy rate dropped.

Nevertheless, we implemented the multi-prototypes on sgFKNN6. We set the number of prototypes on each dataset to about 1%, 5%, 10%, 15%, and 20% of the minimum number of images for each class. For JAFFE dataset, the minimum number of images for each class is 30; we set the number of prototypes to be 1, 3, 5, 7, and 9. For Yale and UNBC datasets, the minimum numbers of images for each class is 28 and 20, respectively; we set the number of prototypes to be 1, 2, 3, 4, and 5. For CMU_AMP dataset, the minimum number of images for each class is 200; we set the number of prototypes to be 1, 10, 20, 30, and 40. For CK+ dataset, the minimum number of images for each class is 100; we set the number of prototypes to be 1, 5, 10, 15, and 20.

Table 4.52 The experimental results from sgFKNN6 for facial expression datasets when using the different number of prototypes.

Dataset	Number of prototypes	1	3	5	7	9
JAFFE	$K=1$	70.01%	71.20%	72.25%	70.98%	69.97%
	$K=2$	68.86%	70.03%	71.06%	69.82%	68.83%
	$K=3$	70.86%	72.06%	73.12%	71.84%	70.82%
	$K=4$	69.16%	71.20%	72.25%	70.98%	69.97%
	$K=5$	62.31%	68.09%	69.09%	67.88%	66.92%
	$K=6$	57.30%	67.27%	68.26%	67.07%	66.12%
	$K=7$	70.72%	71.92%	72.97%	71.70%	70.68%
	$K=8$	N/A	N/A	N/A	N/A	N/A
	$K=9$	N/A	N/A	N/A	N/A	N/A
	$K=10$	N/A	N/A	N/A	N/A	N/A
	$K=C$	70.72%	71.92%	72.97%	71.70%	70.68%
YALE	Number of prototypes	1	2	3	4	5
	$K=1$	48.16%	52.85%	53.65%	54.51%	54.68%
	$K=2$	47.36%	53.01%	53.80%	54.67%	54.84%
	$K=3$	48.74%	53.49%	54.30%	55.17%	55.34%
	$K=4$	44.82%	52.60%	53.40%	54.25%	54.42%
	$K=5$	54.10%	55.89%	56.73%	57.64%	57.82%
	$K=6$	48.74%	53.49%	54.30%	55.17%	55.34%
	$K=7$	N/A	N/A	N/A	N/A	N/A
	$K=8$	N/A	N/A	N/A	N/A	N/A
	$K=9$	N/A	N/A	N/A	N/A	N/A
	$K=10$	N/A	N/A	N/A	N/A	N/A
$K=C$	48.74%	53.49%	54.30%	55.17%	55.34%	
CMU AMP	Number of prototypes	1	10	20	30	40
	$K=1$	88.66%	88.86%	88.99%	88.99%	88.86%
	$K=2$	87.21%	87.40%	87.54%	87.54%	87.40%
	$K=3$	89.74%	89.93%	90.07%	90.07%	89.93%
	$K=4$	88.26%	88.46%	88.60%	88.60%	88.46%
	$K=5$	89.74%	89.93%	90.07%	90.07%	89.93%
	$K=6$	N/A	N/A	N/A	N/A	N/A
	$K=7$	N/A	N/A	N/A	N/A	N/A
	$K=8$	N/A	N/A	N/A	N/A	N/A
	$K=9$	N/A	N/A	N/A	N/A	N/A
	$K=10$	N/A	N/A	N/A	N/A	N/A
$K=C$	89.74%	89.93%	90.07%	90.07%	89.93%	

Note. We have N/A in some results because the dataset has number of classes less than number of K .

Table 4.52 The experimental results from sgFKNN6 for facial expression datasets when using the different number of prototypes (Cont.).

Dataset	Number of prototypes	1	5	10	15	20
CK+	$K=1$	90.92%	91.56%	91.56%	91.32%	90.89%
	$K=2$	89.43%	90.06%	90.06%	89.82%	89.40%
	$K=3$	92.03%	92.68%	92.68%	92.43%	92.00%
	$K=4$	91.36%	92.01%	92.01%	91.76%	91.33%
	$K=5$	90.84%	91.48%	91.48%	91.23%	90.81%
	$K=6$	90.30%	90.94%	90.94%	90.69%	90.27%
	$K=7$	88.48%	89.10%	89.10%	88.86%	88.45%
	$K=8$	87.72%	88.34%	88.34%	88.10%	87.69%
	$K=9$	N/A	N/A	N/A	N/A	N/A
	$K=10$	N/A	N/A	N/A	N/A	N/A
	$K=C$	87.72%	88.34%	88.34%	88.10%	87.69%
	UNBC	Number of prototypes	1	2	3	4
$K=1$		100%	100%	99.47%	99.47%	99%
$K=2$		100%	100%	99.47%	99.47%	99%
$K=3$		100%	100%	99.47%	99.47%	99%
$K=4$		100%	100%	99.47%	99.47%	99%
$K=5$		100%	100%	99.47%	99.47%	99%
$K=6$		N/A	N/A	N/A	N/A	N/A
$K=7$		N/A	N/A	N/A	N/A	N/A
$K=8$		N/A	N/A	N/A	N/A	N/A
$K=9$		N/A	N/A	N/A	N/A	N/A
$K=10$		N/A	N/A	N/A	N/A	N/A
$K=C$		100%	100%	99.47%	99.47%	99%

Note. We have N/A in some results because the dataset has number of classes less than number of K .

The experiment results from sgFKNN6 for facial expression datasets when using the different number of prototypes. We found that the best accuracy rate on JAFFE, YALE, CMU_AMP, CK+ and UNBC dataset are 73.12%, 57.82%, 90.07%, 92.68% and 100%, respectively, when using $K = 3$ and number of prototypes is 5 for JAFFE database, $K = 5$ and number of prototypes is 5 for YALE database, $K = 3, 5$ and number of prototypes are 20, 30 for CMU_AMP database, $K = 3$ and number of prototypes are 5 and 10 for CK+ database, $K =$ every values and number of prototypes are 1 and 2 for UNBC database.

The number of prototypes which provides the highest accuracy rate is about 5% to 15% of the number of sample in each class. However, if the number of prototypes is more

than the optimal value, then accuracy will be dropped because some prototypes in one class may be outliers.

We found that the number of prototypes which provides the highest accuracy rate should be the optimal value. However, if the number of prototypes is more than the optimal value, then accuracy will be dropped because some prototypes in one class may be outliers.

Table 4.53 The experimental results from sgFKNN7 for facial expression recognition datasets.

Dataset	K=1	K=2	K=3	K=4	K=5	K=6	K=7	K=8	K=9	K=10	K=min(nperclass)-1
JAFFE	87.52%	87.52%	88.58%	86.09%	82.33%	80.02%	99.53%	96.73%	88.40%	86.02%	88.40%
YALE	60.20%	60.20%	60.93%	59.22%	60.93%	59.22%	68.46%	66.54%	60.93%	59.22%	60.93%
CMU_A MP	88.87%	88.87%	89.95%	89.95%	90.98%	90.98%	90.98%	90.98%	89.95%	89.95%	89.95%
CK+ (327 VDOs)	96.02%	96.02%	97.20%	95.62%	90.54%	89.06%	93.45%	91.93%	93.45%	91.93%	93.45%
UNBC	100%	100%	100%	100%	100%	100%	100%	100%	100%	100%	100%

From the experimental results, we found that the best accuracy rates on JAFFE is 99.53% when using $K = 7$. For YALE database is 68.46% when using $K = 7$. For CMU AMP database is 90.98%, when using $K = 5$ to 8. For CK+ databases is 97.20% when using $K = 3$. For UNBC database is 100% for every K .

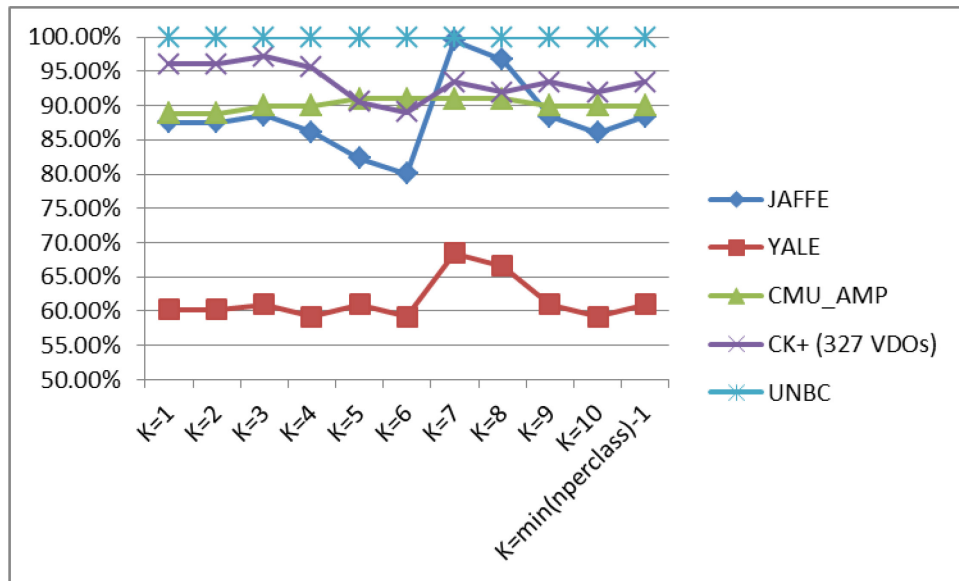


Figure 4.55 The accuracy rate of sgFKNN7 on 5 standard datasets.

In addition, the experimental results show that the accuracy rate is similar to sgFKNN1 and has same pattern when vary on K values but a little bit less than those of sgFKNN1.

The misclassifications from sgFKNN7 algorithm are similar to sgFKNN1. They show that although we use different dataset which make the accuracy rate also changed. However using the same dataset will have the same pattern of accuracy rate, although using the different algorithm which variable on K values. If we use $K=2$ or even values, then the membership values will have an equal opportunity which make the accuracy rate dropped.

From the results of the experiments from sgFKNN1 to sgFKNN7, we found that the accuracy rate from sgFKNN1 is higher than the other algorithms. Moreover, the K values which were the most appropriate and provided the best accuracy rates are $K = 7$ for JAFFE database and Yale database, $K = 5, 6, 7, 8$ for CMU AMP database, $K = 3$ for CK+ database and 100% for all K on UNBC database. Since the UNBC database has very high distribution among classes which is easy for classification. For the most database, the distribution of accuracy rate is lower when starting $K = 1$ and increases when $K = 3, 5$, respectively, until it reaches the peak. The accuracy rate will decrease thereafter, but not for all result.

Moreover, the system provided the greater or equal accuracy rates when using $K = 1, 3, 5, 7,$ and $9,$ respectively, than when using $K = 2, 4, 6, 8,$ and $10.$

In addition, we also developed facial expression recognition based on sgFKNN1 to sgFKNN7 on 5 facial expression recognition datasets which are JAFFE (1600 symbols), Yale (1600 symbols), CMU_AMP (1600 symbols), CK+ (327 VDOs) (1600 symbols), and UNBC-McMaster Shoulder Pain Expression Archive Database (100 symbols).

The experimental results on facial expression recognition from 7 sgFKNNs and the comparison of accuracy rates among our 7 sgFKNN algorithms on facial expression recognition against the others are shown in table 4.45.

Table 4.54 The comparison of accuracy rates between our algorithm of facial expression recognition on 7 sgFKNNs.

Dataset	sgFKNN1	sgFKNN2	sgFKNN3	sgFKNN4	sgFKNN5	sgFKNN6	sgFKNN7	The other's
JAFFE	100%	99.53%	99.53%	75.28%	78.45%	70.86%	99.53%	96.81%[59]
YALE	74.08%	73.04%	73.04%	54.10%	56.38%	54.10%	68.46%	91.52%[60]
CMU_AMP	97.93%	90.90%	90.90%	89.54%	89.64%	89.74%	90.98%	97.67%[61]
CK+ (327 VDOs)	100%	96.97%	96.52%	92.14%	92.81%	92.03%	97.20%	94.14%[62]
UNBC	100%	100%	100%	97.95%	100%	100%	100%	87.40%[63]

From the results of the experiments from sgFKNN1 to sgFKNN7, we found that the accuracy rate from sgFKNN1 is higher than from the other algorithms.

Moreover, the K values that produced the best accuracy rates are $K = 7$ for JAFFE database and Yale database, $K = 5, 6, 7, 8$ for CMU AMP database, $K=3$ for CK+ database, and 100% for all K on UNBC database. For the most database, the distribution of accuracy rate is less when starting $K = 1$ and increases when $K = 3, 5,$ respectively, until it reaches the peak. The accuracy rate will decrease thereafter, but not for all result.

Moreover, the system provided greater or equal accuracy rates when using $K = 1, 3, 5, 7,$ and 9 than when using $K = 2, 4, 6, 8,$ and $10,$ respectively.

From all experiments, sgFKNN1 to sgFKNN7 are able to compare and be better than the previous algorithms in some datasets. However, worse result than the previous methods in some datasets can be seen from the sgFKNN1 to sgFKNN7, because the algorithm's nature is needed to be improved in the future.

According to the results presented in the previous sections, some advantages and some disadvantages of our algorithms can be discussed as follows:

We have the experimental result of 25, 100, 400, 1600 and 2500 symbols per image on Yale database in facial expression recognition when using sgFKNN1 as following:

Table 4.55 The accuracy rate of Yale Database in facial expression recognition.

symbols	$K=1$	$K=9$	$K=14$
25	10.11%	11.24%	10.11%
100	10.11%	22.47%	29.21%
400	14.60%	30.33%	31.46%
1600	61.06%	69.39%	61.80%
2500	57.73%	65.61%	58.43%

From table 4.55, the accuracy rate is higher when we used 1600 symbols. It is confirmed that the system consumed more computing time.

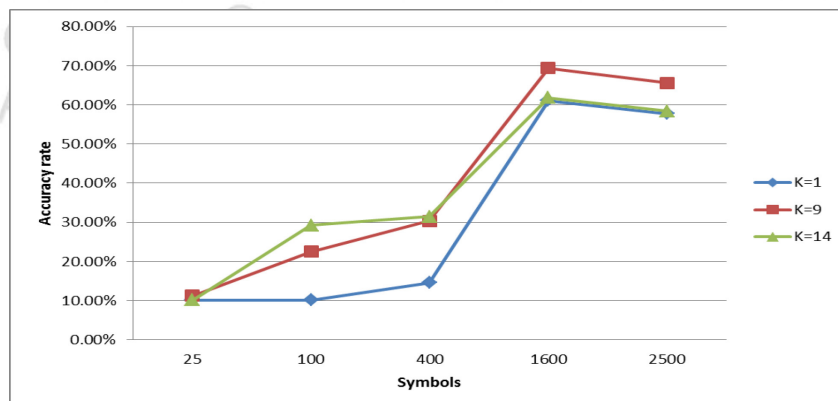





Figure 4.56 The accuracy rate from table 4.55.

In some experiments, we cropped the image or removed padding so that they have a higher accuracy rate.

Table 4.56 The differences between the original and the cropped images

	Original	Crop Level 1	Crop Level 2																																																																																																																																																																																																																																																																																																												
Picture																																																																																																																																																																																																																																																																																																															
String of HOG	<table border="1"> <tr><td>6</td><td>7</td><td>6</td><td>7</td><td>2</td><td>2</td><td>3</td><td>3</td><td>3</td><td>6</td></tr> <tr><td>3</td><td>5</td><td>6</td><td>7</td><td>8</td><td>3</td><td>3</td><td>4</td><td>4</td><td>2</td></tr> <tr><td>7</td><td>5</td><td>2</td><td>2</td><td>4</td><td>7</td><td>7</td><td>4</td><td>4</td><td>3</td></tr> <tr><td>5</td><td>1</td><td>8</td><td>1</td><td>5</td><td>4</td><td>8</td><td>4</td><td>4</td><td>4</td></tr> <tr><td>5</td><td>1</td><td>2</td><td>8</td><td>2</td><td>4</td><td>1</td><td>5</td><td>5</td><td>4</td></tr> <tr><td>4</td><td>1</td><td>8</td><td>8</td><td>7</td><td>6</td><td>1</td><td>5</td><td>5</td><td>5</td></tr> <tr><td>5</td><td>4</td><td>6</td><td>2</td><td>8</td><td>1</td><td>8</td><td>5</td><td>5</td><td>3</td></tr> <tr><td>2</td><td>4</td><td>8</td><td>2</td><td>1</td><td>1</td><td>4</td><td>7</td><td>5</td><td>6</td></tr> <tr><td>1</td><td>2</td><td>4</td><td>7</td><td>1</td><td>1</td><td>3</td><td>4</td><td>7</td><td>4</td></tr> <tr><td>2</td><td>5</td><td>3</td><td>3</td><td>1</td><td>8</td><td>6</td><td>3</td><td>3</td><td>6</td></tr> </table>	6	7	6	7	2	2	3	3	3	6	3	5	6	7	8	3	3	4	4	2	7	5	2	2	4	7	7	4	4	3	5	1	8	1	5	4	8	4	4	4	5	1	2	8	2	4	1	5	5	4	4	1	8	8	7	6	1	5	5	5	5	4	6	2	8	1	8	5	5	3	2	4	8	2	1	1	4	7	5	6	1	2	4	7	1	1	3	4	7	4	2	5	3	3	1	8	6	3	3	6	<table border="1"> <tr><td>7</td><td>6</td><td>2</td><td>2</td><td>2</td><td>3</td><td>3</td><td>3</td><td>1</td><td>1</td></tr> <tr><td>1</td><td>2</td><td>6</td><td>2</td><td>3</td><td>3</td><td>7</td><td>3</td><td>3</td><td>4</td></tr> <tr><td>5</td><td>7</td><td>2</td><td>2</td><td>2</td><td>2</td><td>2</td><td>4</td><td>8</td><td>4</td></tr> <tr><td>5</td><td>1</td><td>7</td><td>7</td><td>6</td><td>6</td><td>7</td><td>4</td><td>4</td><td>8</td></tr> <tr><td>5</td><td>1</td><td>6</td><td>3</td><td>2</td><td>2</td><td>2</td><td>3</td><td>4</td><td>1</td></tr> <tr><td>4</td><td>1</td><td>7</td><td>6</td><td>6</td><td>7</td><td>7</td><td>6</td><td>8</td><td>1</td></tr> <tr><td>4</td><td>1</td><td>6</td><td>6</td><td>3</td><td>2</td><td>3</td><td>8</td><td>7</td><td>3</td></tr> <tr><td>2</td><td>3</td><td>7</td><td>3</td><td>3</td><td>2</td><td>2</td><td>1</td><td>3</td><td>7</td></tr> <tr><td>2</td><td>6</td><td>4</td><td>7</td><td>7</td><td>6</td><td>6</td><td>8</td><td>3</td><td>3</td></tr> <tr><td>2</td><td>2</td><td>2</td><td>2</td><td>3</td><td>2</td><td>2</td><td>4</td><td>8</td><td>3</td></tr> </table>	7	6	2	2	2	3	3	3	1	1	1	2	6	2	3	3	7	3	3	4	5	7	2	2	2	2	2	4	8	4	5	1	7	7	6	6	7	4	4	8	5	1	6	3	2	2	2	3	4	1	4	1	7	6	6	7	7	6	8	1	4	1	6	6	3	2	3	8	7	3	2	3	7	3	3	2	2	1	3	7	2	6	4	7	7	6	6	8	3	3	2	2	2	2	3	2	2	4	8	3	<table border="1"> <tr><td>5</td><td>2</td><td>3</td><td>8</td><td>4</td><td>4</td><td>1</td><td>5</td><td>1</td><td>8</td></tr> <tr><td>6</td><td>6</td><td>2</td><td>8</td><td>5</td><td>1</td><td>2</td><td>7</td><td>6</td><td>4</td></tr> <tr><td>2</td><td>6</td><td>2</td><td>5</td><td>8</td><td>1</td><td>3</td><td>2</td><td>7</td><td>5</td></tr> <tr><td>8</td><td>5</td><td>8</td><td>8</td><td>4</td><td>5</td><td>1</td><td>8</td><td>5</td><td>5</td></tr> <tr><td>8</td><td>5</td><td>8</td><td>5</td><td>6</td><td>1</td><td>4</td><td>7</td><td>8</td><td>5</td></tr> <tr><td>1</td><td>5</td><td>8</td><td>4</td><td>8</td><td>2</td><td>1</td><td>4</td><td>8</td><td>1</td></tr> <tr><td>1</td><td>5</td><td>1</td><td>1</td><td>3</td><td>1</td><td>5</td><td>4</td><td>4</td><td>8</td></tr> <tr><td>4</td><td>1</td><td>3</td><td>2</td><td>6</td><td>2</td><td>2</td><td>1</td><td>5</td><td>4</td></tr> <tr><td>4</td><td>8</td><td>4</td><td>8</td><td>2</td><td>7</td><td>5</td><td>8</td><td>5</td><td>8</td></tr> <tr><td>8</td><td>8</td><td>1</td><td>4</td><td>6</td><td>7</td><td>1</td><td>5</td><td>5</td><td>5</td></tr> </table>	5	2	3	8	4	4	1	5	1	8	6	6	2	8	5	1	2	7	6	4	2	6	2	5	8	1	3	2	7	5	8	5	8	8	4	5	1	8	5	5	8	5	8	5	6	1	4	7	8	5	1	5	8	4	8	2	1	4	8	1	1	5	1	1	3	1	5	4	4	8	4	1	3	2	6	2	2	1	5	4	4	8	4	8	2	7	5	8	5	8	8	8	1	4	6	7	1	5	5	5
6	7	6	7	2	2	3	3	3	6																																																																																																																																																																																																																																																																																																						
3	5	6	7	8	3	3	4	4	2																																																																																																																																																																																																																																																																																																						
7	5	2	2	4	7	7	4	4	3																																																																																																																																																																																																																																																																																																						
5	1	8	1	5	4	8	4	4	4																																																																																																																																																																																																																																																																																																						
5	1	2	8	2	4	1	5	5	4																																																																																																																																																																																																																																																																																																						
4	1	8	8	7	6	1	5	5	5																																																																																																																																																																																																																																																																																																						
5	4	6	2	8	1	8	5	5	3																																																																																																																																																																																																																																																																																																						
2	4	8	2	1	1	4	7	5	6																																																																																																																																																																																																																																																																																																						
1	2	4	7	1	1	3	4	7	4																																																																																																																																																																																																																																																																																																						
2	5	3	3	1	8	6	3	3	6																																																																																																																																																																																																																																																																																																						
7	6	2	2	2	3	3	3	1	1																																																																																																																																																																																																																																																																																																						
1	2	6	2	3	3	7	3	3	4																																																																																																																																																																																																																																																																																																						
5	7	2	2	2	2	2	4	8	4																																																																																																																																																																																																																																																																																																						
5	1	7	7	6	6	7	4	4	8																																																																																																																																																																																																																																																																																																						
5	1	6	3	2	2	2	3	4	1																																																																																																																																																																																																																																																																																																						
4	1	7	6	6	7	7	6	8	1																																																																																																																																																																																																																																																																																																						
4	1	6	6	3	2	3	8	7	3																																																																																																																																																																																																																																																																																																						
2	3	7	3	3	2	2	1	3	7																																																																																																																																																																																																																																																																																																						
2	6	4	7	7	6	6	8	3	3																																																																																																																																																																																																																																																																																																						
2	2	2	2	3	2	2	4	8	3																																																																																																																																																																																																																																																																																																						
5	2	3	8	4	4	1	5	1	8																																																																																																																																																																																																																																																																																																						
6	6	2	8	5	1	2	7	6	4																																																																																																																																																																																																																																																																																																						
2	6	2	5	8	1	3	2	7	5																																																																																																																																																																																																																																																																																																						
8	5	8	8	4	5	1	8	5	5																																																																																																																																																																																																																																																																																																						
8	5	8	5	6	1	4	7	8	5																																																																																																																																																																																																																																																																																																						
1	5	8	4	8	2	1	4	8	1																																																																																																																																																																																																																																																																																																						
1	5	1	1	3	1	5	4	4	8																																																																																																																																																																																																																																																																																																						
4	1	3	2	6	2	2	1	5	4																																																																																																																																																																																																																																																																																																						
4	8	4	8	2	7	5	8	5	8																																																																																																																																																																																																																																																																																																						
8	8	1	4	6	7	1	5	5	5																																																																																																																																																																																																																																																																																																						
Plot Histogram of String	