Darwinian-Based Feature Extraction Using K-Means and Kohonen Clustering

Joshua Adams, Joseph Shelton, Lasanio Small, Sabra Neal, Melissa Venable, Jung Hee Kim, and Gerry Dozier
North Carolina Agricultural and Technical State University
1601 East Market Street
Greensboro, NC 27411
jcadams2@ncat.edu, jashelt1@ncat.edu, lrsmall@ncat.edu, saneal@ncat.edu, mdvenabl@ncat.edu, jungkim@ncat.edu, gvdozier@ncat.edu

Abstract

This paper presents a novel approach to feature extraction for face recognition. This approach extends a previously developed method that incorporated the feature extraction techniques of GEFE$_{ML}$ (Genetic and Evolutionary Feature Extraction – Machine Learning) and Darwinian Feature Extraction). The feature extractors evolved by GEFE$_{ML}$ are superior to traditional feature extraction methods in terms of recognition accuracy as well as feature reduction. From the set of feature extractors created by GEFE$_{ML}$, Darwinian feature extractors are created based on the most consistent pixels processed by the set of feature extractors. Pixels selected during the DFE process are then clustered in an attempt to improve recognition accuracy. Our new approach moves clusters towards large groups of selected pixels using techniques such as k-means clustering and Kohonen clustering. Our results show that DFE clustering (DFE$_C$) has statistically better recognition accuracy than DFE without clustering.

Introduction

This paper presents an improvement on a novel approach to identifying areas of facial images for use by the LBP (Local Binary Pattern) feature extraction technique. The traditional LBP technique computes the frequency of different pixel patterns within an entire image. While traditional LBP deterministically computes a binary pattern from every pixel of the image (Ahonen, Hadid and Pietikinen 2006), our approach identifies highly discriminatory sets of isolated pixels and can provide better discrimination while using only some of the image data.

Our previous work on Genetic and Evolutionary Feature Extraction-Machine Learning (GEFE$_{ML}$) (Shelton et al. 2012a) applied Genetic Algorithms (GAs) (Goldberg 1989; Davis 1991) to discover patches of the image for creating LBP feature vectors. Unlike traditional LBP, which divides an image into a grid of patches, we experimentally determined other sets of image patches we call feature extractors. A feature extractor consists of possibly overlapping rectangles of varying size and position. Our feature extractors have been shown to perform better than the traditional grid in the LBP algorithm for discriminating between faces. GEFE$_{ML}$ also uses cross validation techniques (Mitchell 1997) to ensure that our feature extractors generalize well to unseen datasets.

In (Shelton et al. 2012b), a technique was developed that identified pixel locations in feature extractors that seem the most discriminatory. The feature extractors are used to create a hyper feature extractor. From the hyper feature extractor, a pixel frequency matrix is created. A pixel frequency matrix is a two dimensional matrix containing the number of times each pixel was processed by a set of feature extractors. The pixel frequency matrix is used to determine which pixels will be selected for clustering. Pixel locations are chosen via tournament selection where the most consistently used pixels are chosen without replacement. After the pixel locations are selected, the locations are grouped for feature extraction. This grouping process is performed by randomly placing centers and assigning each pixel location to its nearest center. Clustering the pixels identifies the regions of the image that will form the new patches for the new feature extractors. This technique is referred to as Darwinian Feature Extraction (DFE).

In this paper, we improve upon the DFE technique by performing clustering (Dubes and Jain 1988) on the randomly selected pixels. Our results show that clustering techniques improve the recognition accuracy over traditional DFE. We call this technique DFE$_C$ (Darwinian Feature Extraction using Clustering).

The remainder of this paper is as follows: Section 2 provides an overview of the LBP algorithm, Genetic and Evolutionary Computations (GECs), Genetic and Evolutionary Feature Extractors with Machine Learning (GEFE$_{ML}$), and Darwinian Feature Extraction (DFE). Section 3 gives a description of Darwinian-based Feature Extraction using Clustering (DFE$_C$), Sections 4 describes our experiments and Section 5 provides our results. Our conclusions and future work are presented in Section 6.

Background

This section provides an overview of necessary concepts related to this research. We explain the LBP algorithm; we also explain the GECs used to evolve feature extractors in the hyper feature extractor, specifically an Estimation of Distribution Algorithm (EDA), and we explain GEFE$_{ML}$. Finally, we describe the process of DFE.

Local Binary Patterns

The Local Binary Pattern (LBP) of a pixel is a binary number computed by comparing the difference between the intensity of a pixel, c_p, and its surrounding t pixels. Equation 1 shows how to calculate the binary pattern, where n_t is the intensity of the surrounding pixel and c_p is the intensity of the center pixel.

$$LBP(N, c_p) = \sum_{t=0}^{t} \rho(n_t - c_p) 2^t \qquad (1)$$

$$\rho(x) = \begin{cases} 1, x >= 0 \\ 0, x < 0 \end{cases} \qquad (2)$$

Once the binary pattern for a pixel has been determined, it is then checked for uniformity. We define uniform patterns as binary patters that have fewer than three intensity changes. An intensity change occurs if, when reading the pattern from left to right, a 0 precedes a 1 or a 1 precedes a 0. For example, 00111111 contains one intensity change and 01111000 contains two intensity changes. When $t = 7$, there are exactly 58 unique uniform binary patterns. A histogram of 59 elements is created to store the frequency of different binary patterns. The first 58 elements correspond to unique uniform patterns while the last element corresponds to all non-unique patterns.

The traditional way to construct feature vectors from local binary patterns is to first subdivide the image into subregions called patches. Once the image is divided, a histogram is computed for each of the patches. These histograms are then concatenated together to form a feature vector for the whole image. The feature vectors of two images can be compared using any convenient distance metric.

Genetic and Evolutionary Computations

Genetic and evolutionary computations (GECs) are optimization algorithms that simulate the evolutionary processes found in nature (Eiben and Smith, 2003). The GEC used in this study is an Estimation of Distribution algorithm (EDA). This technique (Larranaga and Lozano 2002) generates new generations of candidate solutions from the previous generation. In our research, a candidate solution is a feature extractor. The EDA technique creates a population distribution function from the feature extractors in one generation. The next generation of feature extractors is created by sampling this distribution. A certain number of the top-performing feature extractors, known as elites, are copied into the new generation. This process, as it applies in this study, is described in more detail in Section 2.4 below.

GEFE$_{ML}$

In GEFE$_{ML}$, a feature extractor is represented by a set of patches. Each patch is comprised of four components. The first component is the patch's location (x, y). The second component is the patch's size (height x width). The third component is a masking bit which is used to determine if the patch will contribute to the resulting biometric template and the fourth component is a fitness value representing the quality of that feature extractor.

The fitness, f_i, is calculated using the percentage of the image being processed and the number of incorrect matches that occur when evaluating a training dataset, D. To calculate the number of errors within the training dataset, D is subdivided into a probe and gallery set. The probe set consists of one image per subject and the gallery set consists of two images per subject. Feature extraction is then performed on each image resulting in a set of probe templates and gallery templates. A simulation of the biometric system is then performed by comparing each probe template to each gallery template using a user defined distance metric. The smallest distance between a probe and all gallery elements is considered a match. An error occurs when the template of an individual in the probe set is incorrectly matched with the template of a different individual in the gallery set. The fitness, shown in Equation 3, is the number of errors multiplied by 10 plus the percentage of image being processed.

$$f_i = 10\varepsilon(D) + \gamma(fe_i) \qquad (3)$$

Cross-validation is used to prevent overfitting the training data. This cross-validation process works by keeping track of the feature extractors that generalize well to a dataset of unseen subjects. While offspring in the evolutionary process are being applied to the training dataset, they are also applied to a mutually exclusive validation dataset which does not affect the fitness value. The offspring that perform best on the validation dataset are recorded even if they do not perform well on the training set.

Darwinian-based Feature Extraction

Creating Darwinian feature extractors is a two stage process: (a) creating hyper feature extractors and (b) creating and sampling the pixel frequency matrix.

Hyper Feature Extractors

We can determine the most discriminatory pixels of an image based on which regions are extracted by feature extractors. Examples of feature extractors evolved by GEFE$_{ML}$ are shown in Figure 1. We take a set of feature extractors from GEFE$_{ML}$ and determine the frequency in which each pixel was processed. We do this by overlaying all feature extractors in the set to create what we refer to as a hyper feature extractor. The hyper feature extractor is used to create a pixel frequency matrix. A pixel frequency matrix is a two dimensional matrix containing a count of the number of time a particular pixel location has been processed by each feature extractor from GEFE$_{ML}$. If a pixel's location falls within a patch of a feature extractor, it is said to have been processed. If patches overlap, then the pixels in the overlap were considered to be processed for the number of times that overlap occurs. Once the pixel frequency matrix is created, it is sampled to form the Darwinian feature extractor.

Sampling the Pixel Frequency Matrix

To create the Darwinian feature extractor, a user specified number of clusters and a total number of pixels to be selected for extraction is decided. In addition, a tournament selection pressure, which determines how many pixels are selected to compete for extraction, is also chosen. The pixels that will be chosen to be extracted are based on a k-tournament selection technique. The user specified number of pixels are chosen to compete, and the pixel that has been processed the most gets chosen to be extracted. Once a pixel has been selected via tournament selection, it will not be selected again as a winner.

In (Shelton et al. 2012b), after all pixels have been selected using tournament selection, centroid positions are randomly assigned and the closest pixels to each centroid are assigned to it. This process is referred to as random clustering. After clusters have been created, the pixels within each cluster will be processed using the LBP feature extraction technique and used to form a histogram. After all selected pixels have been processed; histograms are concatenated to form a feature vector.

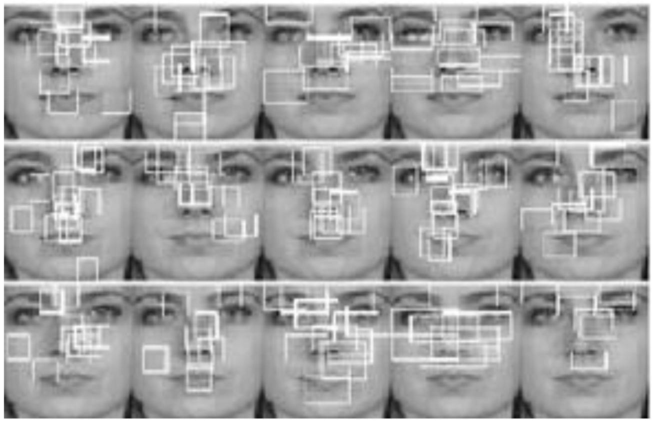

Figure 1: Set of feature extractors.

Techniques for Grouping Pixels to Form Better Feature Extractors

DFE$_C$ (Darwinian Feature Extraction – Clustering) is the process of grouping the pixels selected by DFE. The first step in DFE$_C$ is to form the pixel frequency matrix from the results of GEFE$_{ML}$. A hyper feature extractor is then created from the pixel frequency matrix. A user specified number of centers are randomly placed in the hyper feature extractor and each pixel is assigned to its nearest center. The pixels of the hyper feature extractor are then clustered using one, or a combination, of the following clustering methods.

K-Means Clustering

The process of k-means clustering (Kanungo et al. 2002) the pixels of a Darwinian feature extractor is as follows. Each pixel is assigned to its nearest center. Once this enrollment process is complete, the average location for each enrolled pixel (for a single center) is computed. This average location becomes the new location for that specific center. This process is repeated for each of the user specified centers. Once each of the centers have been relocated, the distance between their old location and their new location is computed. Once the centers no longer move or a maximum of 1,000 iterations have completed, k-means clustering is considered completed.

Kohonen Clustering

The process of clustering the pixels of a Darwinian feature extractor using the Kohonen method (Kohonen 1990) is as follows. Given a user specified learning rate, the Kohonen clustering method iterates though each of the pixels in the Darwinian feature extractor. At each pixel, the nearest center is pulled towards that pixel. The distance which the center is moved is based on the user specified learning rate. With a learning rate of 0.25, the magnitude which the center is moved would be 25% of the distance between the current pixel and the nearest center.

After iterating through all of the pixels, the distance between each centers starting position and ending position is calculated. Kohonen clustering is stopped once the centers no longer move or if one thousand iterations have been completed.

Experiments

To build the hyper feature extractor, we use the 30 best feature extractors created from GEFE$_{ML}$ in previous experiments. Once the hyper feature extractor is created, we use an EDA to evolve the feature extractors based on research (Shelton et al. 2012a) suggesting that EDA is a superior GEC to use. Once we have all of the feature extractors, we built a hyper feature extractor by counting the number of times pixels on an image were processed by each feature extractor. A feature extractor could create up to 24 patches, and we took overlap into account, meaning one feature extractor could process a pixel up to 24 times. We then build a pixel frequency matrix from the hyper feature extractor and sample the pixel frequency matrix to create the new feature extractor.

To evaluate these feature extractors, we apply them to 309 subjects, 3 images per subject, from the Facial Recognition Grand Challenge (FRGC) dataset (Phillips et al. 2005). To test how well each feature extractor generalizes, we divide the FRGC dataset into three subsets. The first subset, FRGC-100_{trn}, is used as our training set and consists of 100 subjects. The second subset, FRGC-109, consists of 109 subjects and is used as our validation set. The remaining 100 subjects are used to form our test set, FRGC-100_{tst}. The recognition accuracy of each feature extractor was determined using the same method to detect errors in Section 2.3, one-to-many matching with cross validation.

For this experiment, we use 12 clusters, 90% of the total number of pixels processed in the pixel frequency matrix, and used a 10% selection pressure, meaning 10% of 504 pixels were selected for tournament selection. We use a constant of 504 due to this being the average number of pixels in a patch within the set of feature extractors. We use this set-up because this was the best performing setup in (Shelton et al 2012b). After each of the feature extractors have been created, we perform four different clustering methods. The first method, DFE_{km}, uses k-means clustering only. The second method, DFE_{k}, uses Kohonen clustering only. The third method, DFE_{km+k}, first uses K-means clustering. The result of K-means clustering is then clustered using Kohonen clustering. The last method, DFE_{k+km}, uses Kohonen clustering first. K-means clustering is then used to cluster the results of Kohonen clustering. Each of these DFE_{c} methods are then applied to FRGC-100_{tst} to evaluate their performance.

Results

The results of DFE_{c} are not deterministic. Each method was tested 30 times; each test utilized 100 different images from FRGC-100tst as described in section 4. Table 1 show our results, including comparison with the best DFE Random Cluster result (Shelton et al., 2012b).

- Column 1 shows the method used.
  - The best DFE Random Cluster used 12 clusters, sampling 0.9 of the pixels, with 0.1 selection pressure (Shelton et al., 2012b).
  - DFE_{km} uses k-means only
  - DFE_{k} uses Kohonen only
  - DFE_{km+k} uses k-means first, then uses these clusters as the starting point for Kohonen
  - DFE_{k+km} uses Kohonen first, then uses these clusters as the starting point for k-means
- Column 2 shows the mean accuracy: the average of the 30 trials of measured accuracies.
- Column 3 shows the standard deviation of the 30 accuracies.

The results of the DFE_{c} instances, when compared to DFE using random clustering, vary in terms of performance. DFE_{km} and DFE_{k+km} have average accuracies that are similar to random clustering. Both DFE_{k} and DFE_{km+k} perform, on average, worse than random clustering. Based on these results, DFE_{km} and DFE_{k+km} seem to be equivalent to DFE. To test this, we performed statistical analysis using a t-test with a 95% confidence interval.

A 2-sample t-test (assuming similar standard deviations) shows that the results of DFE_{k+km} is statistically significant to each of the others ($p < 0.05$), including the best-performing DFE random-clustering method. DFE_{km} is not distinguishable from the random clustering. DFE_{k} and DFE_{km+k} are significantly worse than random clustering.

Given these results, it appears that using Kohonen clustering either as the only technique or using it last in a hybrid clustering technique results in poor performance. However, the hybrid clustering technique proved to be the best, meaning that the order of clustering techniques is relevant.

TABLE 1: RESULTS OF DFE_{C}

	n=30 trials	
Method	**Mean Accuracy**	**Standard Deviation**
<12,0.9,0.1>	99.34%	0.8840866
DFE_{km}	99.43%	0.6260623
DFE_{k}	92.23%	3.1588227
DFE_{km+k}	92.23%	3.1588227
DFE_{k+km}	**99.77%**	**0.4301831**

Conclusion and Future Work

As you can see from the results, DFE_{k+km} provided a higher average accuracy than any of the other methods including DFE without clustering. These results are statistically significant meaning DFE_{k+km} is a better feature extraction method.

None of the other methods were able to achieve an average accuracy greater than or equal to DFE. This gives an indication that the way pixels are grouped together in the feature extraction process significantly affects the accuracy of the biometric system.

The result of Kohonen clustering from a random starting point is fairly stable. Although it theoretically can converge to different solutions from different starting points, as a practical matter different random starting points often converge to the same answer. Thus there is no reason to believe that starting the Kohonen algorithm from the output of k-means would give a better result than the usual random start. Our data bears this out: it seems there was no difference between DFE_{k} and DFE_{km+k}.

The results also show that the two methods (DFE_{km} & DFE_{k+km}) where k-means computes the final clusters have a) the lowest standard deviations and b) the best results. This is consistent with k-means being fairly stable---usually finding same or similar clusters in the same data. It also seems to show that picking a good set of starting clusters, in this case by using Kohonen first, does indeed improve average K-means performance.

Our future work will include creating a hybrid of genetic and k-means clustering. Future work will also consist of using a lower percentage of top pixels from the hyper feature extractor to see how much the computational complexity can be reduced before the recognition accuracy is affected.

References

T. Ahonen, A. Hadid, M. Pietikinen, "Face Description with Local Binary Patterns: Application to Face Recognition," IEEE Transactions on Pattern Analysis and Machine Intelligence, vol. 28, no. 12, pp. 2037-2041, Dec. 2006.

K. Alsabti, S. Ranka, V. Singh. “An Efficient K-Means Clustering Algorithm.” http:://www.cise.ufl.edu/~ranka/, 1997

E. Roy. Davies. “Machine Vision: Theory, Algorithms, Practicalities” (3rd ed.). Amsterdam, Boston, 2005.

L. Davis, “Handbook of Genetic Algorithms”, New York: Van Nostrand Reinhold, 1991.

R.C. Dubes and A.K. Jain, “Algorithms for Clustering Data”. Prentice Hall, 1988

A. E. Eiben and J. E. Smith, “Introduction to evolutionary computing”, Springer, 2003.

D.E. Goldberg, “Genetic Algorithms in Search, Optimization & Machine Learning”, Addison-Wesley Publishing Company, Inc., Reading, Massachusetts, 1989.

A. Jain, Lin Hong, and Sharath Pankanti. “Biometric identification.” Commun. ACM 43, 90-98, vol. no. 2 February 2000.

T. Kanungo, D. Mount, N. Netanyahu, C. Piatko, R. Silverman, A. Wu, “An Efficient k-Means Clustering Algorithm: Analysis and Implementation,” IEEE Transactions on Pattern analysis and Machine Intelligence, vol 24, no 7 July 2002.

T. Kohonen, “The self-organizing map”, Proceedings IEEE, 78 (9). 1464-1480, (1990).

P. Larranaga, and J. A. Lozano, “Estimation of Distribution Algorithms: A New Tool for Evolutionary Computation”, Kluwer Academic Publishers, 2002.

A. Likas, N. Vlassis, J. Verbeek, “The Global k-Means Clustering Algorithm,” The journal of the Pattern recognition Society, Pattern Recognition vol. no. 36, 451-461, 2001.

T. M. Mitchell, “Machine Learning”, McGraw-Hill Companies, Inc. 1997.

T. Ojala, M. Pietikainen, “Multiresolution Gray-Scale and Rotation Invariant Texture Classification with Local Binary Patterns”, IEEE Trans. Pattern Analysis and Machine Intellegence; 971-987; 2002

P. J. Phillips, P. J. Flynn, T. Scruggs, K. W. Bowyer, J. Chang, K. Hoff, J. Marques, J. Min, and W. Worek, “Overview of face recognition grand challenge,” in IEEE Conference on Computer Vision and Pattern Recognition, 2005.

J. Shelton, A. Alford, T. Abagez, L. Small, D. Leflore, J. Williams, J. Adams, G. Dozier, K. Bryant, “Genetic & Evolutionary Biometrics: Feature Extraction from a Machine Learning Perspective”, in submission to IEEE SoutheastCon 2012a.

J. Shelton, M. Venable, S. Neal, J. Adams, A. Alford, G. Dozier; “Pixel Consistency, K-Tournament Selection, and Darwinian-Based Feature Extraction”. Submitted to the Midwest Artificial Intelligence and Cognitive Science Conference (MAICS), 2012b.

M. Su, C. Chou, “A modified version of the K-Means Algorithm with a Distance Based on Cluster Symmetry”, IEEE Transactions on Pattern Analysis and Machine Intelligence, vol.23, no.6, Jun 2001.

K. Wagstaff, C. Cardie, S. Rogers, S. Schroedl, “Constrained K-Means Clustering with Background Knowledge,” Proceedings of the Eighteenth International Conference on Machine Learning, p. 577-584, 2001.