Automatically Searching for Optimal Parameter Settings Using a Genetic Algorithm*

David S. Bolme[1], J. Ross Beveridge[1], Bruce A. Draper[1], P. Jonathon Phillips[2], and Yui Man Lui[1]

[1] Colorado State University, Fort Collins, CO, USA
[2] NIST, Gaithersburg, MD, USA

Abstract. Modern vision systems are often a heterogeneous collection of image processing, machine learning, and pattern recognition techniques. One problem with these systems is finding their optimal parameter settings, since these systems often have many interacting parameters. This paper proposes the use of a Genetic Algorithm (GA) to automatically search parameter space. The technique is tested on a publicly available face recognition algorithm and dataset. In the work presented, the GA takes the role of a person configuring the algorithm by repeatedly observing performance on a tuning-subset of the final evaluation test data. In this context, the GA is shown to do a better job of configuring the algorithm than was achieved by the authors who originally constructed and released the LRPCA baseline. In addition, the data generated during the search is used to construct statistical models of the fitness landscape which provides insight into the significance from, and relations among, algorithm parameters.

1 Introduction

Recent years have seen significant improvements in computer vision, as demonstrated by measurable progress of standard data sets in areas such as face recognition, object recognition, and action recognition. Much of this improvement comes from combining algorithms within single systems. Therefore, many modern vision systems contain image processing, machine learning, and pattern recognition techniques that work together to solve a specific problem. Unfortunately, tuning these multi-part algorithms is difficult, particularly when changing a parameter in one part of a system may have unforeseen effects on another.

Common practice is to use expert judgment and trial-and-error to search for optimal tunings of parameters. All too often, researchers choose a set of parameters, train the system, and evaluate it on the test data. They then alter a

* This work was funded in part by the Technical Support Working Group (TSWG) under Task SC-AS-3181C. Jonathon Phillips was supported by the Department of Homeland Security, Director of National Intelligence, Federal Bureau of Investigation and National Institute of Justice. The identification of any commercial product or trade name does not imply endorsement or recommendation by Colorado State University or the National Institute of Standards and Technology.

parameter and repeat the process until they "crack" the data set. This process has several problems. Most significantly, the test data becomes an implicit part of the training data. In addition, optimal parameters may still be missed because tests were run on individual components instead of the whole system, or because interactions among parameters were misunderstood. Intuition can also be misleading, with the result being that some good parameters are never tested.

This paper presents a technique that replaces parameter tuning by a human experimenter with a Genetic Algorithm (GA). This has many advantages. The GA can tirelessly evaluate thousands of algorithm configurations, improving the likelihood that the best configurations in the search space will be explored. All parameters are optimized simultaneously, allowing the GA to seek out superior configurations in the presence of complex parameter interactions. The configuration is based on a "fitness function" that evaluates the system as a whole, rather than the performance of subcomponents. The GA is not subject to errors of human judgement that may exclude lucrative regions of the search space.

This research uses a GA to find an optimal configuration for the Local Region Principal Components Analysis (LRPCA) face recognition baseline algorithm, as applied to the the Good, Bad, and Ugly (GBU) challenge problem [11]. This baseline improves on Principal Components Analysis (PCA) by adding pre-processing and post-processing steps as well as multiple subspaces for 14 regions of the face. The results show that the GA configuration outperforms the best known manual configuration and highlights the importance of parameter configurations where performance on the tuning subset varies from 5% to 35%, a factor of 7 change in accuracy simply by tuning parameters[3].

The performance results with the GA parameters must be taken with a grain of salt. These results represent the GA's ability to "crack" the data set, since the test data was evaluated as part of the fitness function. Nonetheless, this is similar to the process followed by many human researchers. Moreover, it enables the second contribution of this paper: the use of a Generalized Linear Model (GLM) to analyze system parameters, revealing which parameters are critical and which sets of parameters are most strongly inter-related. The GA evaluates a system's performance over thousands of parameter value combinations. This creates a treasure trove of test data, which can be mined to determine how system performance is affected by each parameter. In this paper we fit a GLM to the performance data to model how the parameters impact LRPCA performance on the GBU data sets. For the LRPCA algorithm, the results indirectly indicate which regions of the face are most important for the algorithm and have the potential to produce improvements in future versions of the system.

2 Related Work

Parameter tuning for a complex algorithm is a well known problem. Support vector machines (SVMs) are one example of a complex technique with a large

[3] Performance numbers are for the Correct Verification Rate at a 0.001 False Accept Rate.

number of domain-specific parameters, and as a result there are several papers that search for optimal parameters, e.g. [2,10]. Of particular interest is a report by Hsu and Lin [7] in which experts were asked to hand-tune parameters of an SVM, and the results were compared to parameters learned by an automatic grid search of parameter space. In all three cases, the learned parameters outperformed those sets turned by the human experts.

In computer vision where algorithms are built that embody models, parameter estimation is often approached as statistical model fitting. For example, Felzenszwalb's object detector models objects as mixtures of multi-scale deformable parts [4], and much of the technique involves fitting parameters to data. These are "strong" techniques that exploit top-down constraints to guide parameter selection and have proven to be effective when they can be applied.

Unfortunately, the best face recognition algorithms are multi-step systems with interacting components, and the implications of their parameters are often poorly understood. A recent paper by Cox and Pinto [3] uniformly samples parameter space (using many processors) for a face recognition algorithm and shows that the resulting parameters improve performance over hand-tuned configurations. It should be noted that finding optimal parameters differs from the methods of Karurgaru [8] who used a GA to find optimal positions and scales for templates within the face matching process.

Earlier, Givens et al [5] used a generalized linear mixed-effects model (GLMM) to analyze the effects of parameters on an LDA+PCA algorithm [17] but not to search for optimal parameter values. In our approach a GLM is used to model parameter space in a manor similar to Givens et al [5], thereby extracting configuration information about the underlying algorithm. Harzallah et al [6] used a rank-based *Friedman* Test for a similar purpose, however the GLM's model can be better related to the fitness landscape.

3 Searching for Optimal Configurations

3.1 Training, Tuning, and Test Datasets

The Face Recognition Vendor Test 2006 showed that face recognition technology could verify a person's identity with 99% accuracy in high quality images taken under controlled conditions [13]. However, face recognition in uncontrolled conditions is much more difficult. The GBU challenge problem [11] contains three partitions of face images of varying difficulty from uncontrolled environments. The Good partition contains images that are easy to match, while the Ugly partition is extremely difficult, and the Bad partition is somewhere in between. The purpose of GBU is to improve performance on the Bad and Ugly partitions without sacrificing performance on the Good.

The GBU Challenge Problem has a clearly stated protocol for presenting performance results. It requires training be done on an independent set of images that contain no images of the people present in the GBU test data. For this purpose, a set of 673 images from the MBGC Still Image problem [12] that are

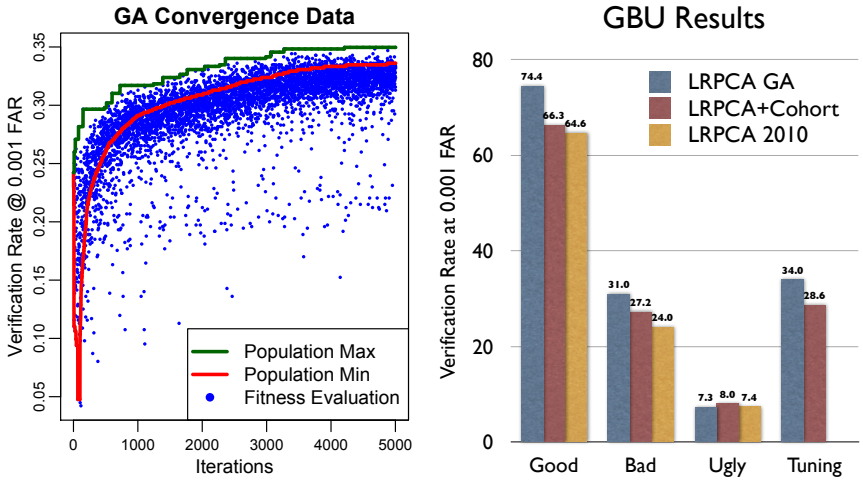

Fig. 1. The left plot shows the convergence of the GA where each blue dot is one fitness function evaluation. The right plot compares the GA tuned algorithm to the manual tuned equivalent (LRPCA+Cohort) and the standard configuration (LRPCA 2010).

disjoint from the people included in GBU is used as a training set. These images are used to train the algorithms basis vectors.

A distinction is drawn between training and tuning. Tuning is the process typically carried out by an algorithm developer where parameters are repeatedly modified and then performance is tested on the challenge imagery. Here, when the GA evaluates the fitness of a particular tuning, it considers the verification rate on a tuning-subset of the actual GBU test data.

If one views the entire GA as a machine learning tool for constructing a better algorithm, the use of the tuning-subset of the test data is a violation of the GBU protocol which requires a separate dataset for training. However, this paper views the role of the GA as a surrogate for what researchers do when tuning algorithms. A goal of this paper is to better understand how tuning-parameters effects performance on a benchmark problem, which requires that the tuning-subset drawn from the test data itself. The tuning-subset used here is composed of approximately 1/6 of the GBU testing images. In future work, the GA will be tested as a method to improve generalized performance, where the GA only has access to training data.

3.2 The LRPCA Baseline Algorithm

The experiments presented use an open source face recognition baseline algorithm called Local Region Principal Components Analysis (LRPCA) [11][4]. LRPCA is based on the well known eigenfaces algorithm [9,14] but includes improvements to the way faces are preprocessed, analyzed, and compared to produce higher accuracy than a simple PCA based approach.

The input to the algorithm is an image containing a face and the coordinates of both eyes. The eye coordinates are used to geometrically normalize the face and the image is then split into 14 smaller images that represent local regions of the face focused on the eyebrows, eyes, nose, mouth, etc. Each region is preprocessed using the Self Quotient Image (SQI) [15] which reduces the effect of lighting, and the pixel values are then normalized to have a mean of 0.0 and a standard deviation of 1.0.

PCA is run on each region to produce a set of basis vectors. A configurable number of eigenvectors can be dropped corresponding to both the largest (PCA Min) and smallest (PCA Max) eigenvalues to further reduce the effect of illumination and noise. This dimension reduction allows the algorithm to better generalize. LRPCA optionally whitens the basis vectors such that, when projected, the training data has a variance of 1.0 in all dimensions.

A weight is also computed for each basis vector which is the between-class variance divided by the within-class variance (σ_b^2/σ_w^2). Vectors with the largest weight are kept where the total number of vectors is a configurable parameter (Total Dimensions). This weight is also used to emphasize vectors that better discriminate among people.

During testing, new faces are normalized and projected onto the basis and the similarity between the faces is measured using correlation. LRPCA was extended with cohort normalization [1] which offers a slight improvement to the verification rates shown in Figure 1. This is done by computing the similarity between each testing image and faces in the training set. The non-match distribution can then be normalized using the following equation:

$$s'(i,j) = \frac{s(i,j) - \frac{1}{2}(\mu_i + \mu_j)}{\frac{1}{2}(\sigma_i + \sigma_j)} \tag{1}$$

where $s(i,j)$ is correlation, and μ_i and σ_i are the mean and standard deviation of non-match scores for test images i and j estimated from the cohort set.

3.3 Genetic Algorithm and Configuration Space

The parameter space was optimized by a rank-based genetic algorithm similar to GENATOR [16] available as part of the PyVision library[5]. Genetic algorithms are stochastic optimization techniques inspired by evolution and natural

[4] `http://www.cs.colostate.edu/facerec/algorithms/lrpca2010.php`
[5] `http://pyvision.sourceforge.net`

Table 1. This table shows the parameters tuned by the GA along with their optimal values.

Parameter	Type	Range	Manual Value	GA Value
Region 0: Full Face	Float	0.50 - 1.00	1.00	0.927
Region 1: Left Eye	Float	0.10 - 0.50	0.33	0.433
Region 2: Right Eye	Float	0.10 - 0.50	0.33	0.342
Region 3: Far Left Brow	Float	0.10 - 0.36	0.33	0.360
Region 4: Center Left Brow	Float	0.10 - 0.36	0.33	0.285
Region 5: Center Right Brow	Float	0.10 - 0.36	0.33	0.286
Region 6: Far Right Brow	Float	0.10 - 0.36	0.33	0.360
Region 7: Nose Bridge	Float	0.10 - 0.66	0.33	0.395
Region 8: Nose Tip	Float	0.10 - 0.66	0.33	0.100
Region 9: Left Nose	Float	0.10 - 0.70	0.33	0.211
Region 10: Right Nose	Float	0.10 - 0.70	0.33	0.259
Region 11: Left Mouth	Float	0.10 - 0.20	0.20	0.167
Region 12: Center Mouth	Float	0.10 - 0.20	0.20	0.200
Region 13: Right Mouth	Float	0.10 - 0.20	0.20	0.154
SQI Blurring Radius	Float	0.5 - 20.0	3.0	19.43
PCA Min Dimension	Int	0 - 20	2	19
PCA Max Dimension	Int	100 - 400	250	169
PCA Whitening Enabled	Bool	True/False	True	True
Final Basis Dimensions	Int	100 - 4000	3500	880

selection. Algorithm configurations are represented as individuals in a simulated population, where more fit individuals are selected for survival and breeding. In this experiment, the population contains 100 randomly generated individuals and each iteration follows these steps:

1. Two individuals of the population are selected randomly.
2. Those individuals are combined to produce a new individual where configuration parameters are selected randomly from the parents.
3. Small perturbations are made to the new individual to simulate mutation.
4. The new individual is evaluated using the fitness function.
5. If the new individual scores higher than the previously lowest rank individual in the population, that lowest ranked individual is replaced.

The fitness function evaluates each individual by completing the full training and testing process. The algorithm was trained using the configuration in the genetic code and then was evaluated on the tuning-subset at a false accept rate of 0.001. The GA was run on a quad-core Intel i7 with 8 worker processes which completed forty evaluations per hour resulting in 5004 total evaluations. Table 1 summarizes the 19 parameters that were tuned by the GA and also gives the manually selected default parameters as well as the best configuration produced by the GA.

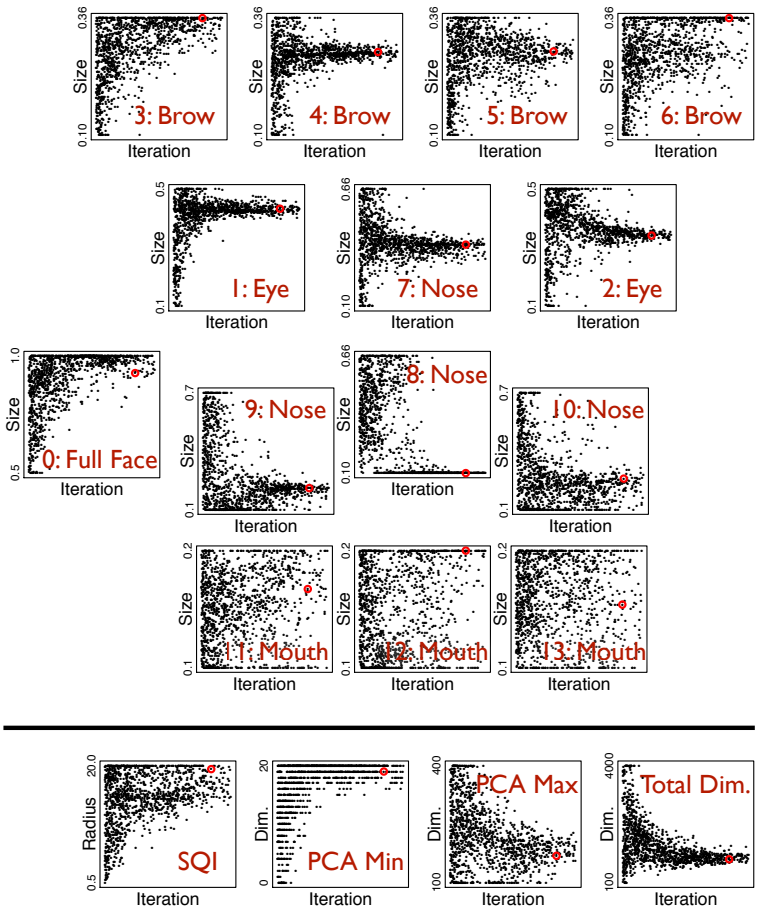

Fig. 2. This figure shows that the configuration parameters converge as optimization progresses. The top shows local region sizes where plots are arranged relative to their locations on the face. The bottom shows the convergence of the radius of the Gausian filter used for the SQI normalization, the minimum and maximum PCA cutoffs, and the total number of basis vectors included in the final configuration. Only configurations added to the population are shown and the best configuration is circled in red (Iteration 4209).

3.4 The Optimal Configuration

Figure 1 shows the GA convergence. Each iteration corresponds to one fitness function evaluation where the fitness scores are shown as blue dots. The green line represents the best known configuration at each iteration and the red line represents the worst configuration in the population. Figure 2 shows how each

of the parameters converge throughout the GA run. There are a few parameters where there is no clear preference for any particular value. This suggests those parameters have little effect on performance. Regions 3, 6, and 8 show interesting behavior where the best values are at the boundary of the configuration space which suggest the range for those parameters could be expanded.

The best configuration was evaluated on the full GBU challenge problem in Figure 1. This illustrates the benefit of using the GA to search for optimal configurations. Good and Bad performance improved significantly, while the performance on the Ugly partition dropped by a small amount. This suggests that tuning real world systems using GAs may offer important performance increases.

4 Data Mining the Search Space

A more interesting aspect of this work is what the optimization process tells us about the shape of the parameter space. Each fitness evaluation relates a point in that space to a score for the algorithm. During the course of the run the space is sampled thousands of times, with higher density near the optimal solutions.

To understand the configuration space, a GLM was fit to the search results. The response variable $\hat{Y}$ is the score produced by the fitness function and the X_i correspond to the values and squared values of the algorithm parameters:

$$\hat{Y} = \alpha + \beta_1 X_1 + \beta_2 X_1^2 + \beta_3 X_2 + \beta_4 X_2^2 + \cdots \tag{2}$$

The α (intercept) and β_i variables are fit to the dataset to minimize the sum of squared error in the model, which is a second order approximation to the configuration landscape and is used to estimate the importance of each parameter.

Figure 3 illustrates the GLM approximation to the fitness surface. In this case only points that were added to the population are shown. Additionally, whitening always resulted in a better score; therefore, the model was fit to configurations where whitening was turned on. The basic shape of the landscape can be inferred by the top row but the shape is more obvious when the parameters are controlled for by the GLM as shown on the bottom.

This analysis reveals interesting shapes in the fitness landscape. Region 3 suggests a linear response and the range searched by the GA could be extended. Region 7 shows a nice second order response where the best values selected by the GA correspond nicely to the best values suggested by the GLM. Region 8 shows a response curve suggesting the best values may be larger or smaller than what was searched by the GA. Also, a hill climbing approach would not properly optimize this region. The GA, however, maintains multiple configurations in its population and therefore focuses the search on both ends of the range. Total Dimensions are also an interesting case where the model does not appear to fit the data well and suggests a higher order GLM may be necessary.

While the model presented in the previous section is used to understand the effects of the parameters, it is often important to understand which parameters are effecting each other. Again a GLM is the analysis tool, but the new model

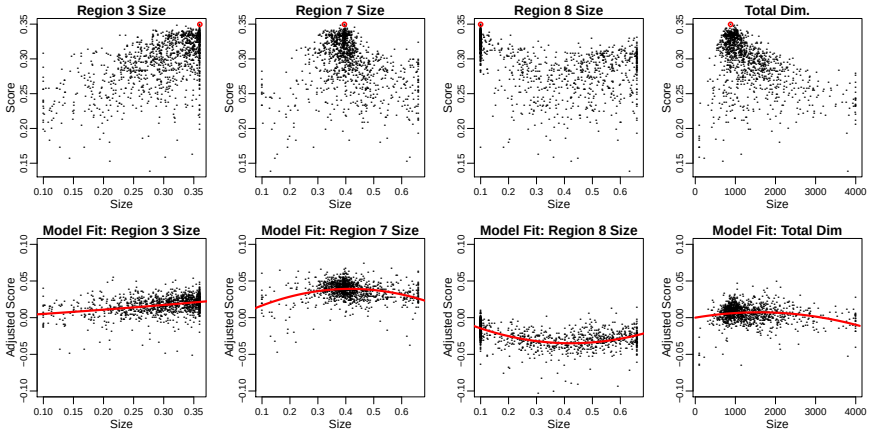

Fig. 3. These figures illustrate the fitness landscape for some of the configuration parameters. The top figures show the raw fitness scores relative to the parameter value. The bottom figures show the adjusted score taking all other parameters into account. The model is shown as a red line.

will add interaction terms. If a and b are parameters, the original model had the terms for a, a^2, b, and b^2, while the new model adds an interaction term $a * b$.

Initially 153 interaction terms were added to the GLM; however, most terms did not contribute predictive information. A greedy local search reduced the model parameters to the minimum needed to accurately represent the data, as measured by Akaike Information Criterion (AIC). This reduced the model to 78 interactions. This number was further reduced by computing the significance of dropping each remaining term from the model which used an F-test. This resulted in 20 terms that were highly significant ($P < 0.001$).

A few parameters were found to repeatedly participate in the most significant interactions. Region 1 participated four times, Region 2 three times, Region 7 four times, Region 8 three times, PCA Min Dimensions three times, PCA Max Dimensions six times, and Total Dimensions six times. The regions participating in these interactions correspond to the eyes, nose bridge, and nose tip which are thought to be the best areas of the face for biometric matching.

5 Conclusions

This paper used a GA to find the optimal parameter settings for the LRPCA algorithm, producing a better configuration than manual tuning. The GA simultaneously optimizes 19 parameters in the context of the complete system, which takes the fitness landscape and parameter interactions into account.

A GLM-based analysis provides additional knowledge of the algorithm by modeling the fitness landscape. This shows when parameters have been set cor-

rectly or when additional tuning may be necessary. The analysis identifies which parameters are most significant and which parameters have the strongest interactions. This insight into the parameter space may lead to better performance in future versions of the system.

References

1. G. Aggarwal, N.K. Ratha, R.M. Bolle, and R. Chellappa. Multi-biometric cohort analysis for biometric fusion. In *ASSP*, 2008.
2. V. Cherkassky and Y. Ma. Practical selection of SVM parameters and noise estimation for SVM regression. *Neural Networks*, 17(1):113–126, 2004.
3. D.D. Cox and N. Pinto. Beyond simple features: A large-scale feature search approach to unconstrained face recognition. In *Face and Gesture*, 2011.
4. P. Felzenszwalb, R. Girschick, D. McAllester, and D. Ramanan. Object detection with discriminatively trained part based models. *T-PAMI*, 2009.
5. G.H. Givens, J.R. Beveridge, B.A. Draper, and D.S. Bolme. Using a generalized linear mixed model to study the configuration space of pca+lda human face recognition algorithm. *LNCS : Articulated Motion and Deformable Objects*, 2004.
6. H. Harzallah, F. Jurie, and C. Schmid. Combining efficient object localization and image classification. In *International Conference Computer Vision*, 2009.
7. C. W. Hsu, C. C. Chang, and C. J. Lin. A practical guide to support vector classification. From LibSVM, December 2007.
8. S. Karungaru, M. Fukumi, and N. Akamatsu. Face recognition using genetic algorithm based template matching. In *Communications and Information Technology*, volume 2, pages 1252–1257. IEEE, 2004.
9. M. Kirby and L. Sirovich. Application of the karhunen-loeve procedure for the characterization of human faces. *T-PAMI*, 12(1), 1990.
10. A. Lorena and A. de Carvalho. Evolutionary tuning of svm parameter values in multi-class problems. *Neurocomputing*, 71(16 – 18):3326–3334, 2008.
11. P.J. Phillips, J.R. Beveridge, B.A. Draper, G.H. Givens, A.J. O'Toole, D.S. Bolme, J. Dunlop, Y.M. Lui, H. Sahibzada, and S. Weimer. An introduction to the good, the bad, & the ugly face recognition challenge problem. In *Face and Gesture*, 2011.
12. P.J. Phillips, R. Beveridge, G. Givens, B. Draper, D. Bolme, Y.M. Lui, N. Teli, T. Scruggs, G.E. Cho, K. Bowyer, P. Flynn, and A. O'Toole. Overview of the multiple biometric grand challenge results of version 2. Presentation at Multiple Biometric Grand Challenge 3rd Workshop, December 2009.
13. P.J. Phillips, W.T. Scruggs, A.J. O'Toole, P.J. Flynn, K.W. Bowyer, C.L. Schott, and M. Sharpe. FRVT 2006 and ICE 2006 large-scale results. *National Institute of Standards and Technology*, 2007.
14. M. A. Turk and A. P. Pentland. Face recognition using eigenfaces. In *CVPR*, 1991.
15. H. Wang, S.Z. Li, Y. Wang, and J. Zhang. Self quotient image for face recognition. In *ICIP*, 2004.
16. D. Whitley. The GENITOR algorithm and selection pressure: Why rank-based allocation of reproductive trials is best. In *Int. conf. on genetic algorithms*, 1989.
17. W. Zhao, R. Chellappa, and A. Krishnaswamy. Discriminant analysis of principal components for face recognition. In *Face and Gesture*, 1998.