

Feature Selection

Jain, A.K.; Duin, P.W.; Jianchang Mao, “**Statistical pattern recognition: a review**”, IEEE Transactions on Pattern Analysis and Machine Intelligence, vol. 22, no. 1, pp. 4-37, Jan. 2000.

Section 4.2

Genetic Algorithms

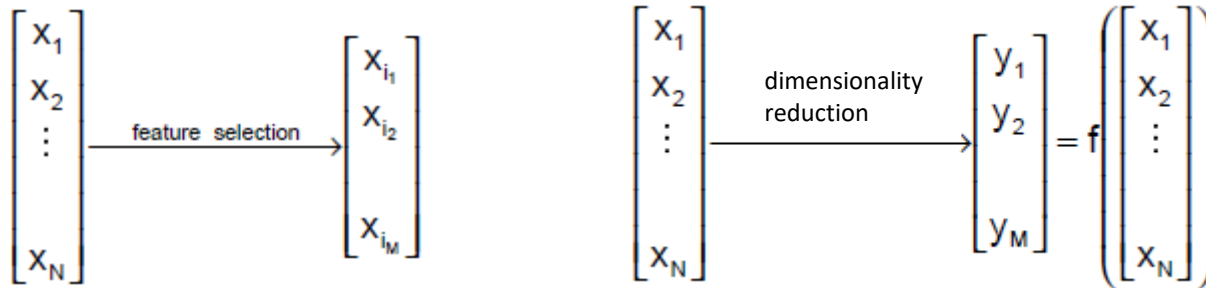
Chapter 7 (Duda et al.) – Section 7.5

CS479/679 Pattern Recognition

Dr. George Bebis

Feature Selection

- Given a set of **n** features, the goal of **feature selection** is to select a subset of **d** features (**d** < **n**) in order to minimize the classification error.



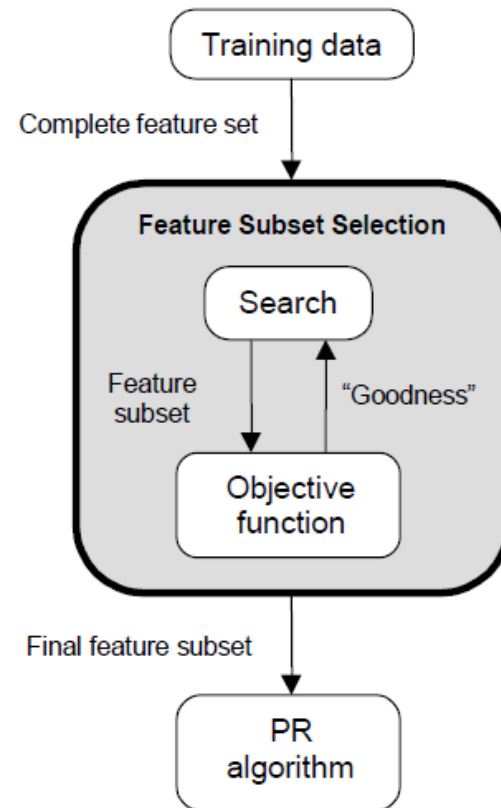
- Fundamentally different from dimensionality reduction (e.g., PCA or LDA) based on feature combinations (i.e., **feature extraction**).

Feature Selection vs Dimensionality Reduction

- Dimensionality Reduction
 - When classifying novel patterns, **all** features need to be computed.
 - The measurement units (length, weight, etc.) of the features are **lost**.
- Feature Selection
 - When classifying novel patterns, only a **small** number of features need to be computed (i.e., faster classification).
 - The measurement units (length, weight, etc.) of the features are **preserved**.

Feature Selection Steps

- Feature selection is an **optimization** problem.
 - **Step 1:** Search the space of possible feature subsets.
 - **Step 2:** Pick the subset that is optimal or near-optimal with respect to some objective function.



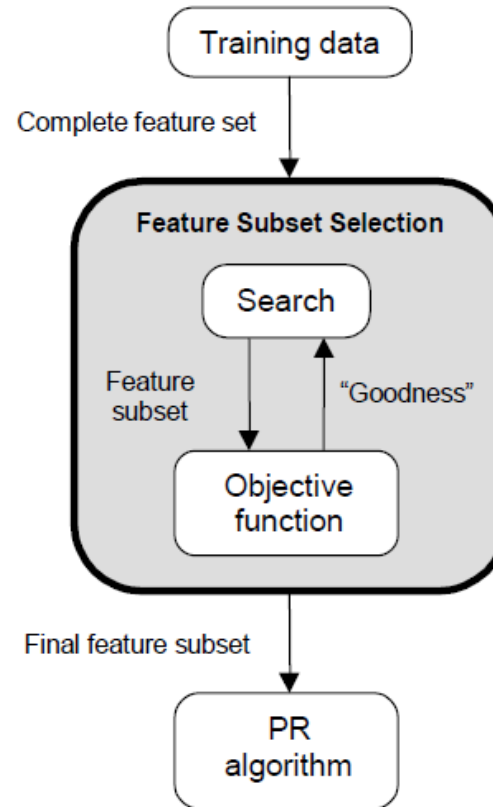
Feature Selection Steps (cont'd)

Search strategies

- Optimum
- Heuristic
- Randomized

Evaluation strategies

- Filter methods
- Wrapper methods



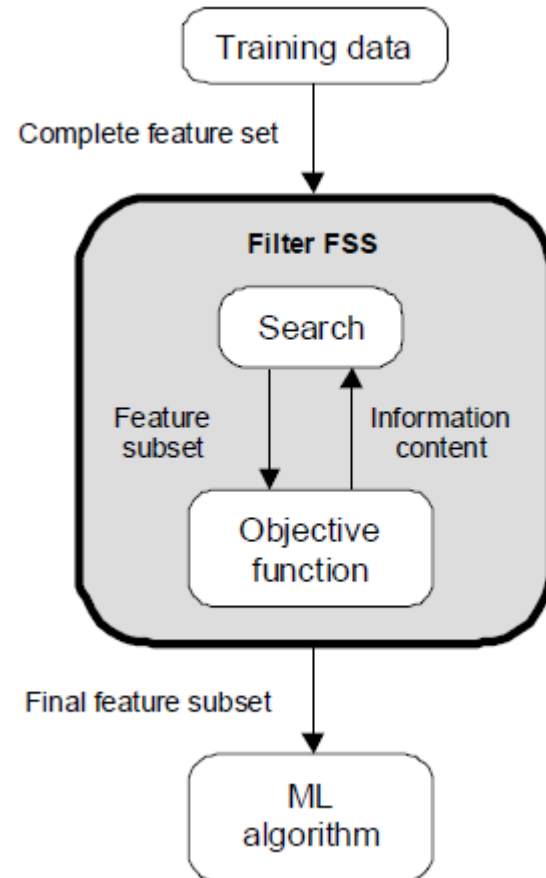
Search Strategies

- Assuming **n** features, an exhaustive search would require:
 - Examining all $\binom{n}{d}$ possible subsets of size d.
 - Selecting the subset that performs the best according to the criterion function.
- The number of subsets grows **combinatorially**, making exhaustive search impractical.
- In practice, heuristics are used to speed-up search but they **cannot** guarantee optimality.

Evaluation Strategies

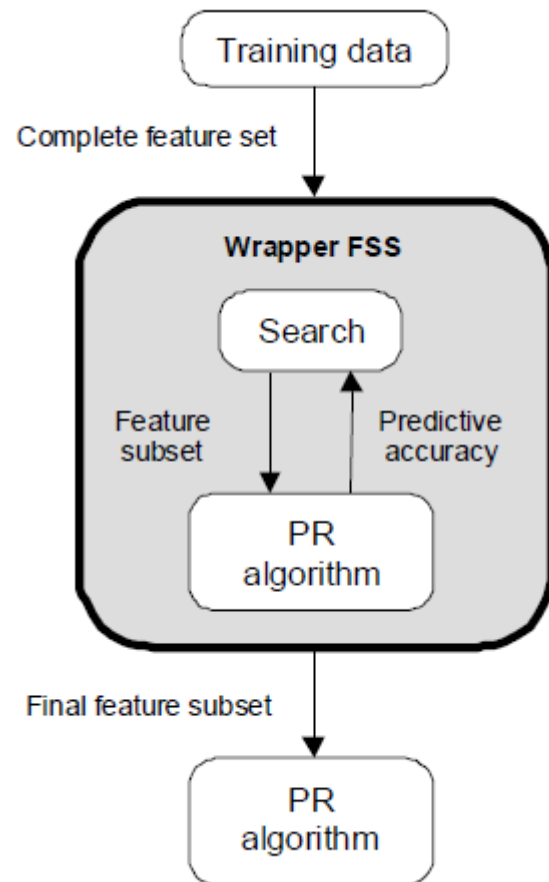
- **Filter Methods**

- Evaluation is **independent** of the classification algorithm.
- The objective function evaluates feature subsets by their information content, typically interclass distance, statistical dependence or information-theoretic measures.



Evaluation Strategies

- **Wrapper Methods**
 - Evaluation uses criteria **related** to the classification algorithm.
 - The objective function is a pattern classifier, which evaluates feature subsets by their predictive accuracy (recognition rate on test data) by statistical resampling or cross-validation.



Filter vs Wrapper Approaches

■ Wrappers

- Advantages
 - **Accuracy:** wrappers generally achieve better recognition rates than filters since they are tuned to the specific interactions between the classifier and the dataset
 - **Ability to generalize:** wrappers have a mechanism to avoid overfitting, since they typically use cross-validation measures of predictive accuracy
- Disadvantages
 - **Slow execution:** since the wrapper must train a classifier for each feature subset (or several classifiers if cross-validation is used), the method can become unfeasible for computationally intensive methods
 - **Lack of generality:** the solution lacks generality since it is tied to the bias of the classifier used in the evaluation function. The “optimal” feature subset will be specific to the classifier under consideration

Filter vs Wrapper Approaches (cont'd)

■ Filters

- Advantages

- **Fast execution:** Filters generally involve a non-iterative computation on the dataset, which can execute much faster than a classifier training session
- **Generality:** Since filters evaluate the intrinsic properties of the data, rather than their interactions with a particular classifier, their results exhibit more generality: the solution will be “good” for a larger family of classifiers

- Disadvantages

- **Tendency to select large subsets:** Since the filter objective functions are generally monotonic, the filter tends to select the full feature set as the optimal solution. This forces the user to select an arbitrary cutoff on the number of features to be selected

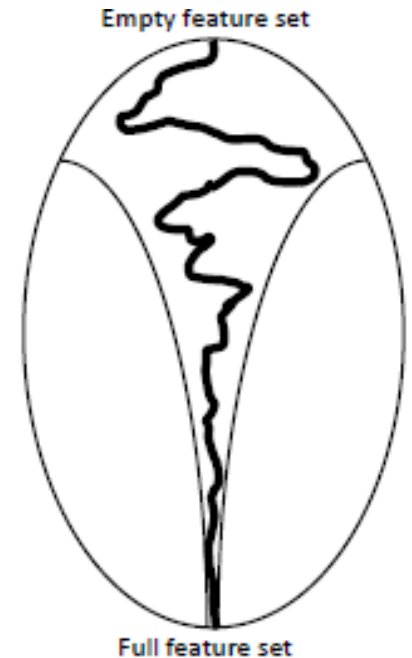
Naïve Search

- Sort the given n features in order of their probability of correct recognition.
- Select the top d features from this sorted list.
- Disadvantage
 - Correlation among features is not considered.
 - The best pair of features may not even contain the best individual feature.

Sequential forward selection (SFS)

(heuristic search)

- First, the best **single** feature is selected (i.e., using some criterion function).
- Then, **pairs** of features are formed using one of the remaining features and this best feature, and the best pair is selected.
- Next, **triplets** of features are formed using one of the remaining features and these two best features, and the best triplet is selected.
- This procedure continues until a predefined number of features are selected.

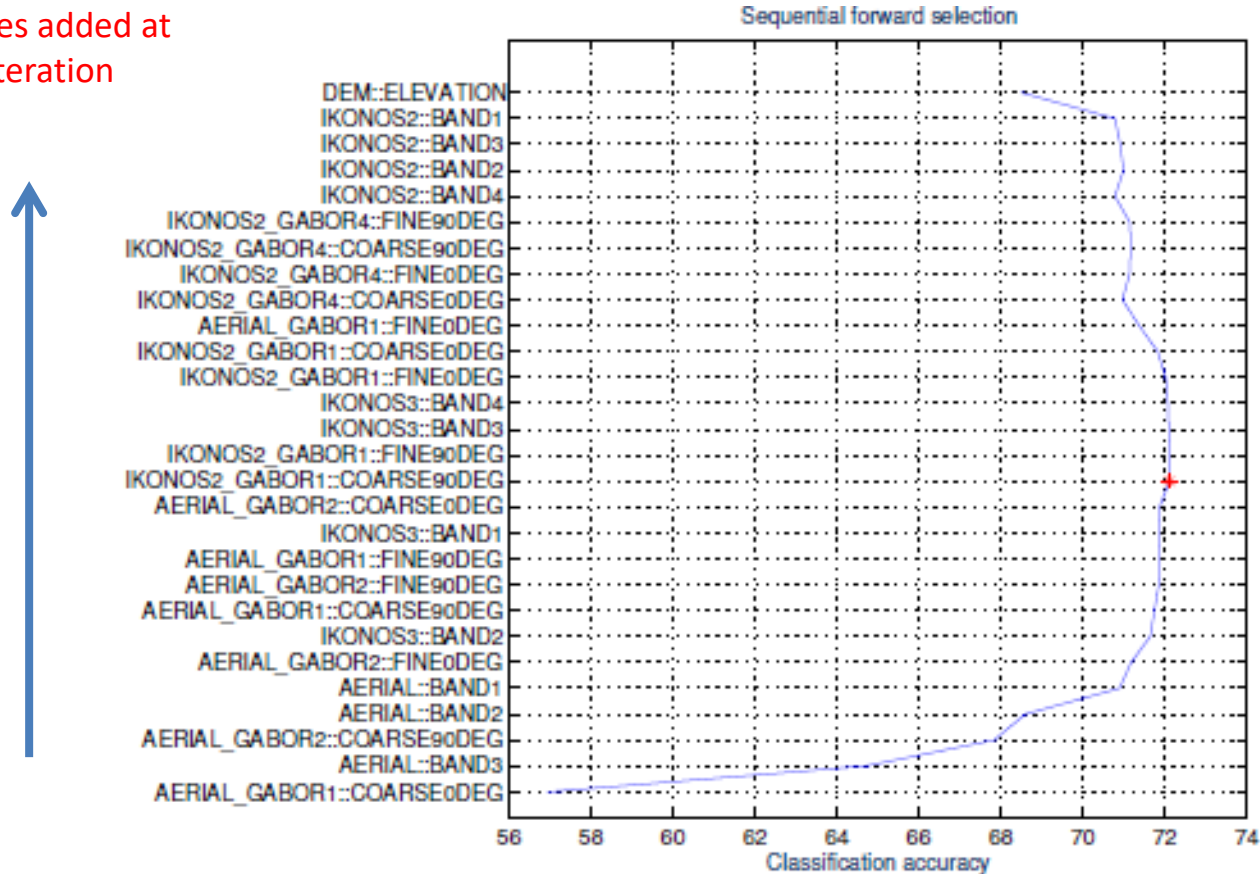


1. Start with the empty set $Y_0 = \{\emptyset\}$
2. Select the next best feature $x^+ = \arg \max_{x \notin Y_k} J(Y_k + x)$
3. Update $Y_{k+1} = Y_k + x^+$; $k = k + 1$
4. Go to 2

SFS performs best when the optimal subset is **small**.

Example

features added at
each iteration



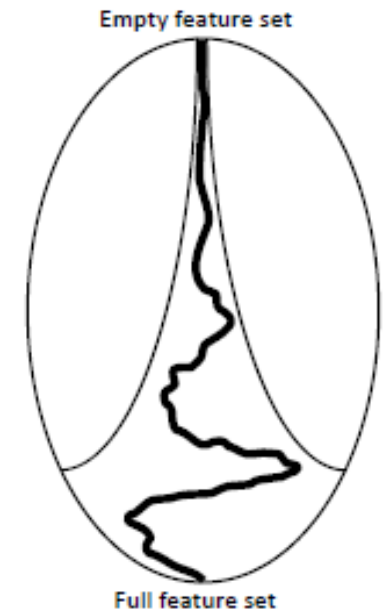
Results of **sequential forward** feature selection for classification of a satellite image using 28 features. x-axis shows the classification accuracy (%) and y-axis shows the features added at each iteration (the first iteration is at the bottom). The highest accuracy value is shown with a star.

Sequential backward selection (SBS)

(heuristic search)

- First, the criterion function is computed for all n features.
- Then, each feature is **deleted** one at a time, the criterion function is computed for all subsets with $n-1$ features, and the worst feature is discarded.
- Next, each feature among the remaining $n-1$ is deleted one at a time, and the worst feature is discarded to form a subset with $n-2$ features.
- This procedure continues until a predefined number of features are left.

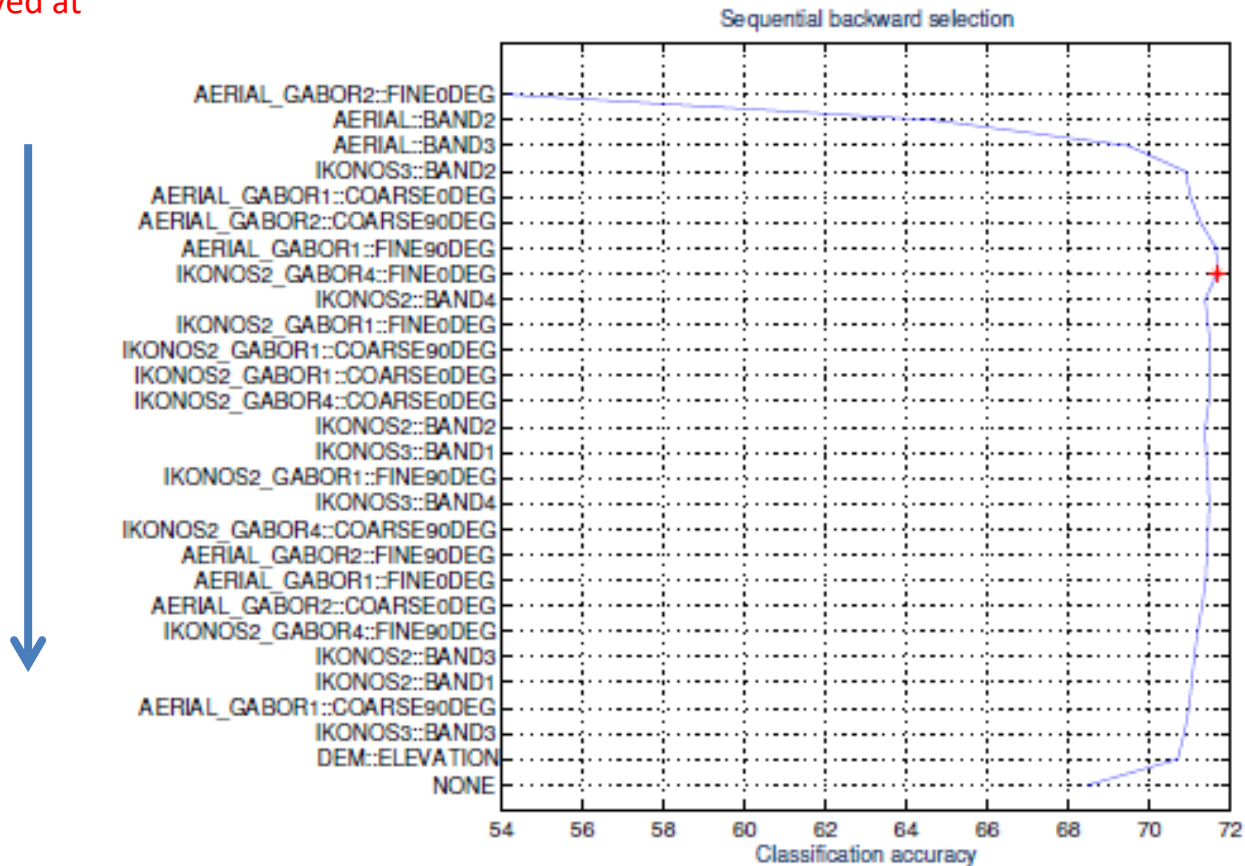
1. Start with the full set $Y_0 = X$
2. Remove the worst feature $x^- = \arg \max_{x \in Y_k} J(Y_k - x)$
3. Update $Y_{k+1} = Y_k - x^-$; $k = k + 1$
4. Go to 2



SBS performs best when the optimal subset is **large**.

Example

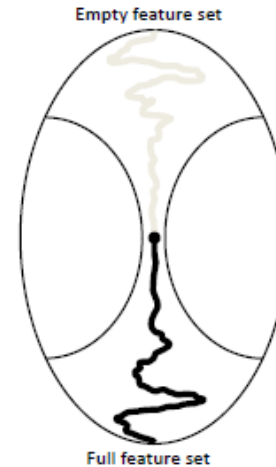
features removed at
each iteration



Results of **sequential backward** feature selection for classification of a satellite image using 28 features. x-axis shows the classification accuracy (%) and y-axis shows the features removed at each iteration (the first iteration is at the top). The highest accuracy value is shown with a star.

Bidirectional Search (BDS)

- BDS applies SFS and SBS simultaneously:
 - SFS is performed from the empty set.
 - SBS is performed from the full set.
- To guarantee that SFS and SBS converge to the same solution:
 - Features already selected by SFS are not removed by SBS.
 - Features already removed by SBS are not added by SFS.



1. Start SFS with $Y_F = \{\emptyset\}$
2. Start SBS with $Y_B = X$
3. Select the best feature

$$x^+ = \arg \max_{\substack{x \notin Y_{F_k} \\ x \in F_{B_k}}} J(Y_{F_k} + x)$$

$$Y_{F_{k+1}} = Y_{F_k} + x^+$$
4. Remove the worst feature

$$x^- = \arg \max_{\substack{x \in Y_{B_k} \\ x \notin Y_{F_{k+1}}}} J(Y_{B_k} - x)$$

$$Y_{B_{k+1}} = Y_{B_k} - x^-; k = k + 1$$
5. Go to 2

Limitations of SFS and SBS

- The main limitation of SFS is that it is unable to **remove** features that become non useful after the addition of other features.
- The main limitation of SBS is its inability to **reevaluate** the usefulness of a feature after it has been discarded.
- We will examine some generalizations of SFS and SBS:
 - Plus-L, minus-R” selection (LRS)
 - Sequential floating forward/backward selection (SFFS and SFBS)

“Plus-L, minus-R” selection (LRS)

- A generalization of SFS and SBS
 - If $L > R$, LRS starts from the **empty** set and:
 - Repeatedly add L features
 - Repeatedly remove R features
 - If $L < R$, LRS starts from the **full** set and:
 - Repeatedly removes R features
 - Repeatedly add L features

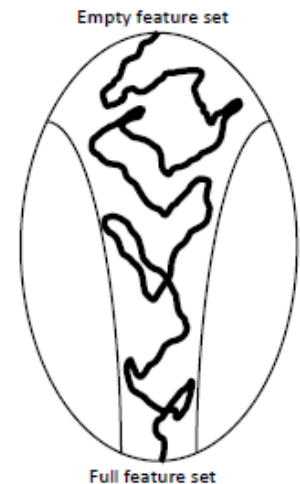
```
1. If  $L > R$  then  $Y_0 = \{\emptyset\}$   
   else  $Y_0 = X$ ; go to step 3  
2. Repeat L times  
    $x^+ = \arg \max_{x \notin Y_k} J(Y_k + x)$   
    $Y_{k+1} = Y_k + x^+; k = k + 1$   
3. Repeat R times  
    $x^- = \arg \max_{x \in Y_k} J(Y_k - x)$   
    $Y_{k+1} = Y_k - x^-; k = k + 1$   
4. Go to 2
```



Its main limitation is the lack of a theory to help choose the optimal values of L and R.

Sequential floating forward/backward selection (SFFS and SFBS)

- An extension to LRS:
 - Rather than fixing the values of L and R, floating methods determine these values from the data.
 - The dimensionality of the subset during the search can be thought to be “floating” up and down
- Two floating methods:
 - Sequential floating forward selection (SFFS)
 - Sequential floating backward selection (SFBS)



P. Pudil, J. Novovicova, J. Kittler, Floating search methods in feature selection, Pattern Recognition Lett. 15 (1994) 1119–1125.

Sequential floating forward selection (SFFS)

- Sequential floating forward selection (SFFS) starts from the empty set.
- After each forward step, SFFS performs backward steps as long as the objective function increases.

```
1.  $Y = \{\emptyset\}$ 
2. Select the best feature
    $x^+ = \arg \max_{x \notin Y_k} J(Y_k + x)$ 
    $Y_k = Y_k + x^+; k = k + 1$ 
3. Select the worst feature*
    $x^- = \arg \max_{x \in Y_k} J(Y_k - x)$ 
4. If  $J(Y_k - x^-) > J(Y_k)$  then
    $Y_{k+1} = Y_k - x^-; k = k + 1$ 
   Go to step 3
Else
   Go to step 2
```

*Notice that you'll need to do book-keeping to avoid infinite loops

Sequential floating backward selection (SFBS)

- Sequential floating backward selection (SFBS) starts from the full set.
- After each backward step, SFBS performs forward steps as long as the objective function increases.

Genetic Algorithms (GAs)

(randomized search)

- What are GAs?
 - A **global** optimization technique for **searching** very large spaces.
 - Inspired by the biological mechanisms of **natural selection** and **reproduction**.
- Main characteristics of GAs
 - Searches **probabilistically** using a **population** of possible solutions.
 - Each solution is properly **encoded** as a string of symbols.
 - Uses an **objective** (or **fitness**) function to evaluate the “goodness” of each solution.
 - Does not require using derivatives.

Population of
encoded solutions

10010110...

01100011...

01100011...

10100100...

Encoding

- Each solution in the search space is represented as a finite length string (**chromosome**) over some finite set of symbols.

e.g., using binary encoding

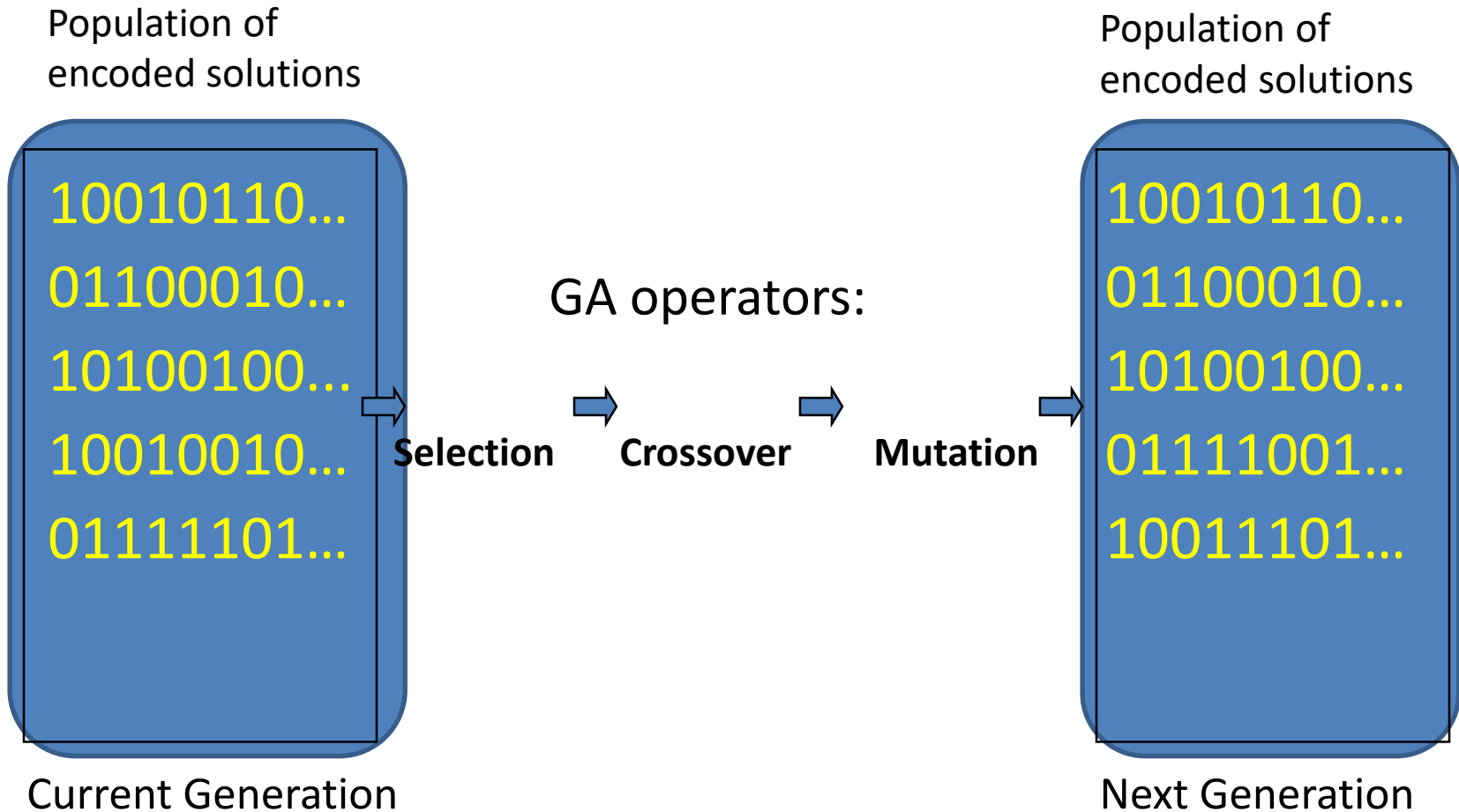
$(11, 6, 9) \rightarrow (1011_0110_1001) \rightarrow (101101101001)$

Fitness Evaluation

- A fitness function is used to evaluate the goodness of each solution.
- The fitness function is “problem specific”.

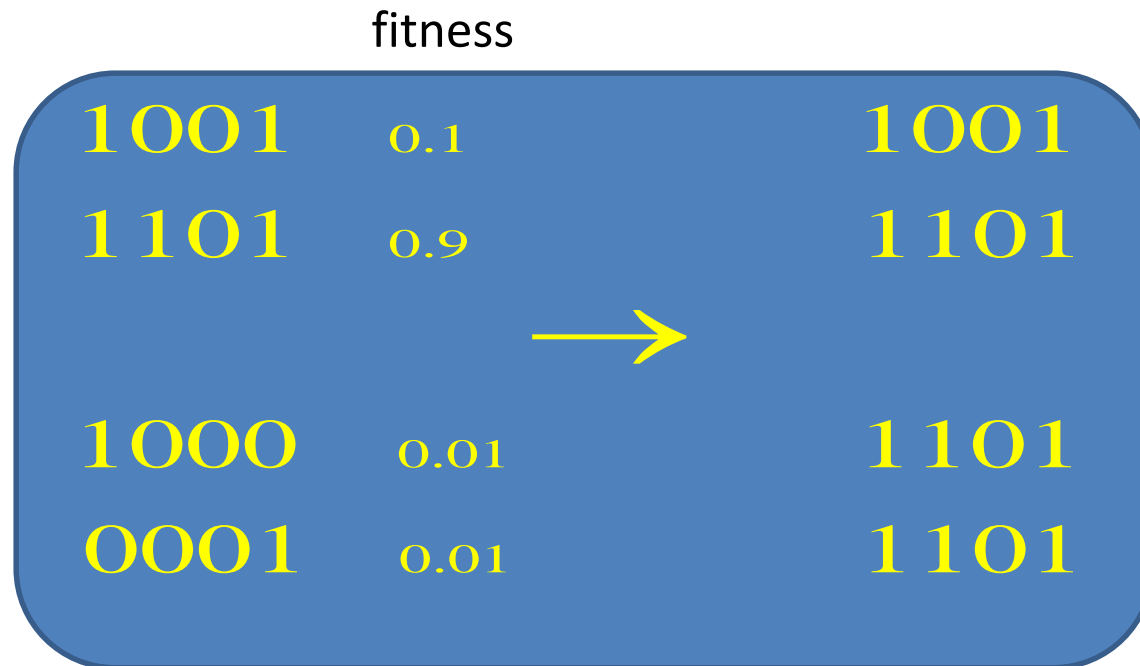
$$Fitness = f(\text{decode}(\text{chromosome}))$$

Searching



Selection

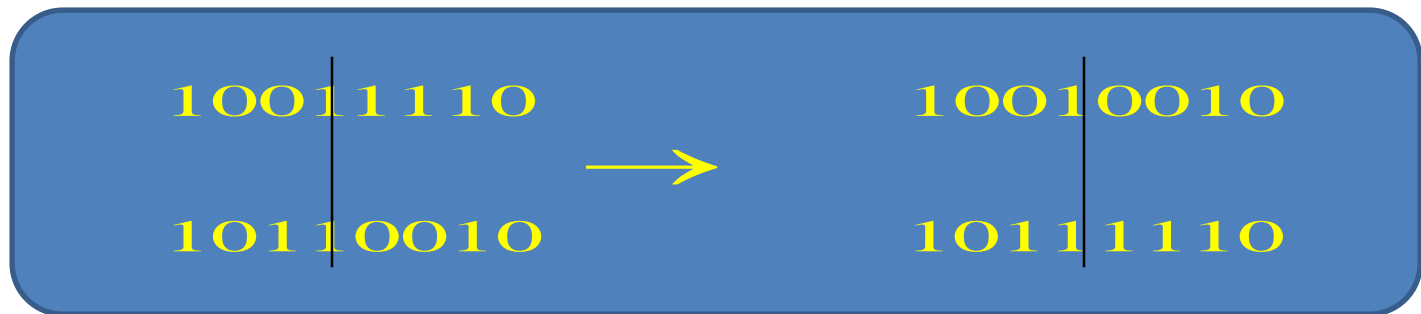
- Probabilistically filters out solutions that perform poorly, choosing high performance solutions to **exploit**.



Crossover

Explore new solutions:

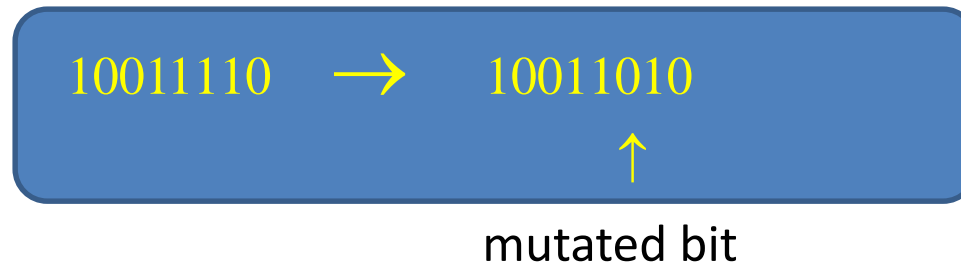
- **Crossover:** information **exchange** between strings.
 - Generate new chromosomes that, hopefully, will retain good features from the previous generation.
 - It is applied to randomly selected pairs of chromosomes with a probability equal to a given *crossover rate*.



Mutation

Explore new solutions:

- **Mutation:** restore lost genetic material.
 - It protects GAs against irrecoverable loss of good solution features.
 - It changes a character of some chromosome with a probability equal to a very low given *mutation rate*.



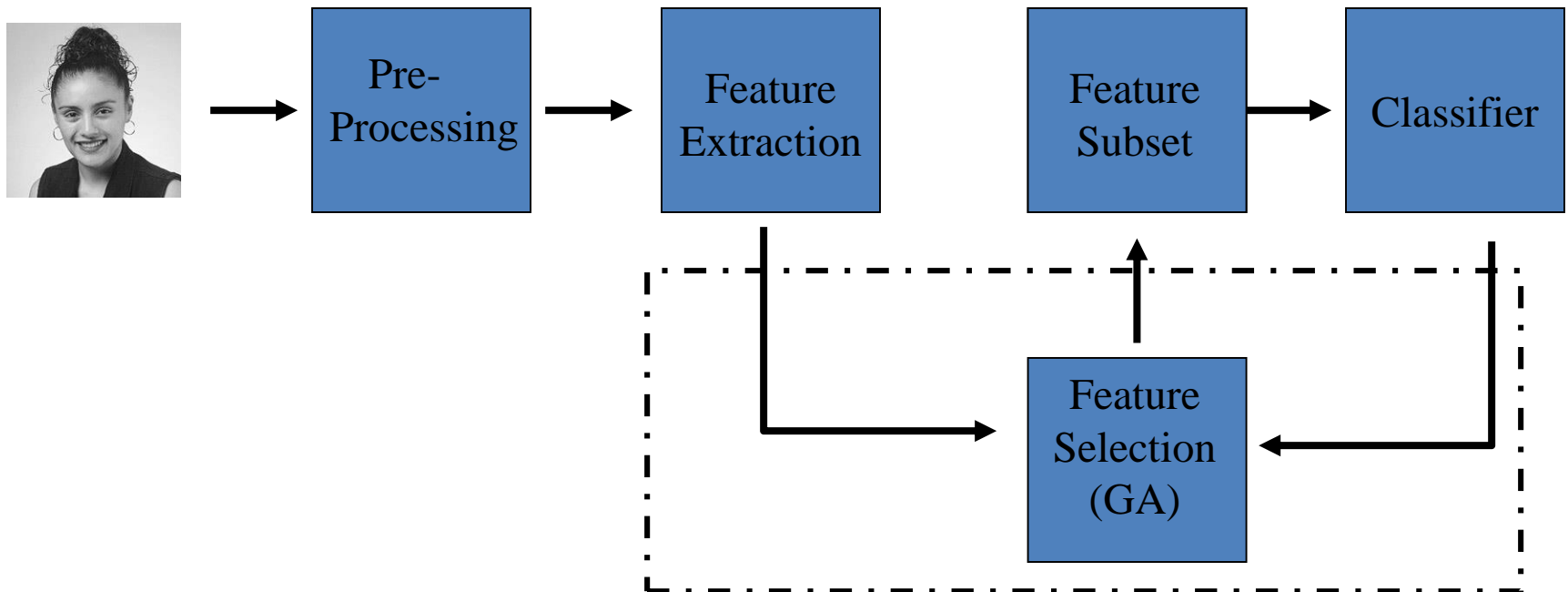
Steps

1. Initialize a population with randomly generated individuals.
2. Evaluate the fitness of each individual.
3. Reproduce high fitness chromosomes in the new population, remove poor fitness chromosomes (*selection*).
4. Construct new chromosomes (*crossover*).
5. Recover lost features (*mutation*).
6. Repeat steps 2-6 until a stopping criterion is met.

Feature Selection using GAs

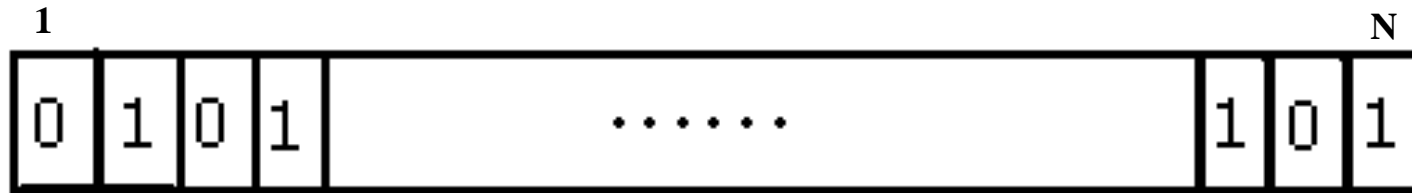
(randomized search)

- GAs provide a simple, general, and powerful framework for feature selection.

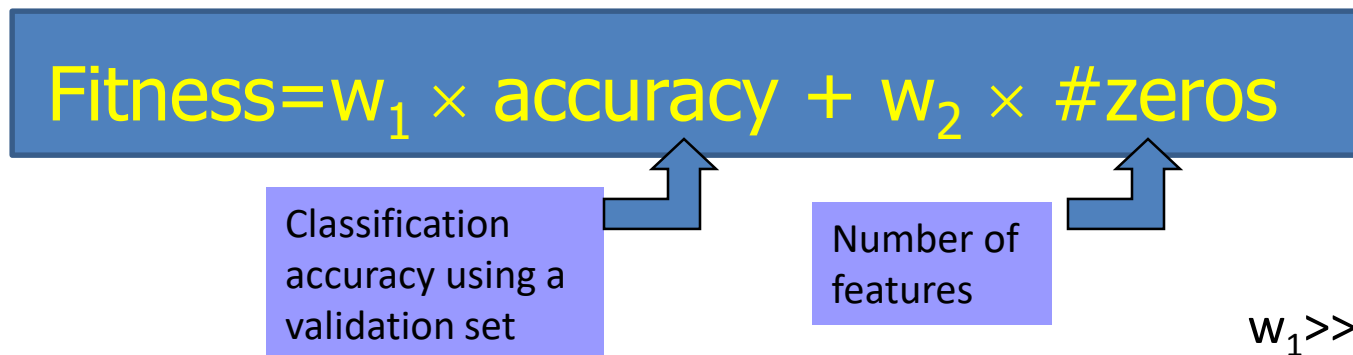


Feature Selection Using GAs (cont'd)

- **Binary encoding:** **1** means “choose feature” and **0** means “do not choose” feature.

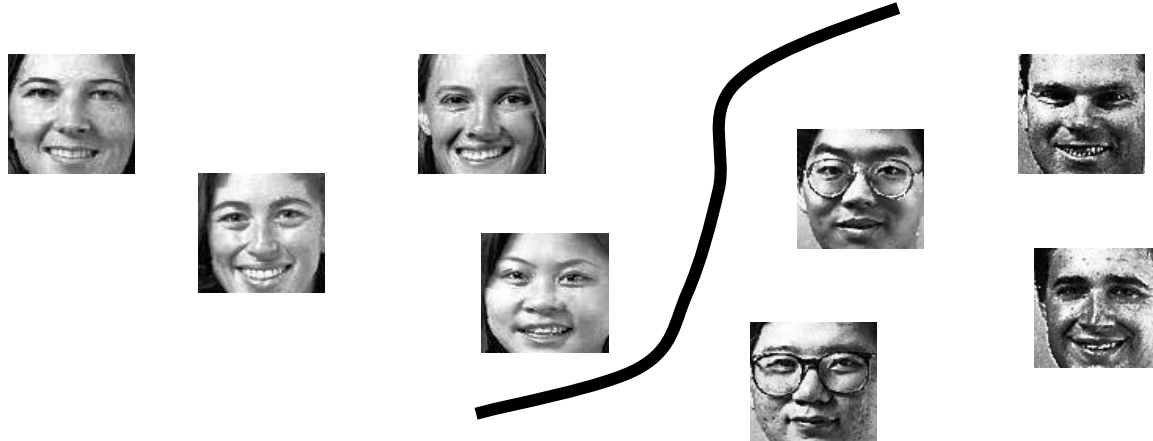


- **Fitness evaluation** (to be maximized)



Case Study 1: Gender Classification

- Determine the gender of a subject from facial images.
 - Challenges: race, age, facial expression, hair style, etc.



Z. Sun, G. Bebis, X. Yuan, and S. Louis, "Genetic Feature Subset Selection for Gender Classification: A Comparison Study", **IEEE Workshop on Applications of Computer Vision**, pp. 165-170, Orlando, December 2002.

Feature Extraction Using PCA

- Use PCA to represent faces in terms of the “best” eigenvectors:


$$\text{Im} = a_1 \text{EV}_1 + a_2 \text{EV}_2 + \dots + a_N \text{EV}_N$$

Which eigenvectors encode mostly gender information?



EV#1



EV#2



EV#3



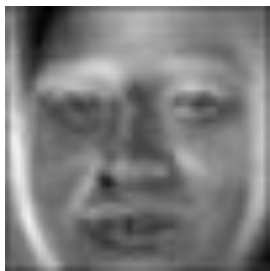
EV#4



EV#5



EV#6



EV#8



EV#10



EV#12



EV#14



EV#19



EV#20

IDEA: Use GAs to search the space of eigenvectors!

Dataset

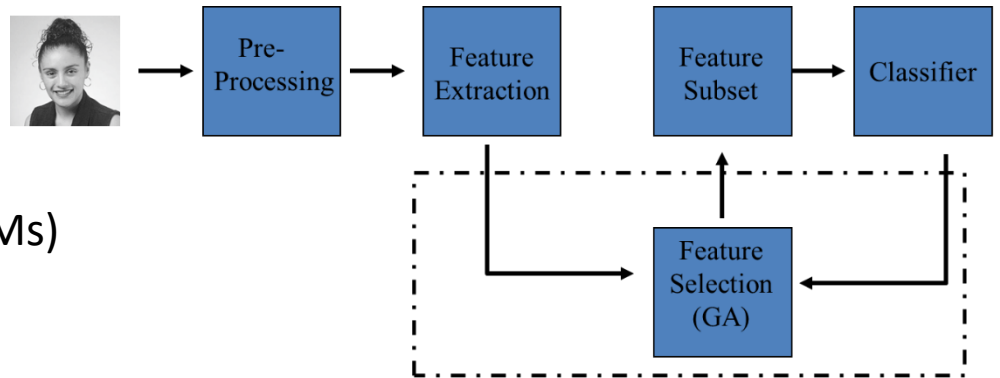
- 400 frontal images from 400 different people
 - 200 male, 200 female
 - Different races, lighting conditions, and facial expressions
- Images were registered and normalized
 - No hair information
 - Normalized to account for different lighting conditions



Experiments

- Tested different classifiers:

- LDA
- Bayes classifier
- Neural Networks (NNs)
- Support Vector Machines (SVMs)

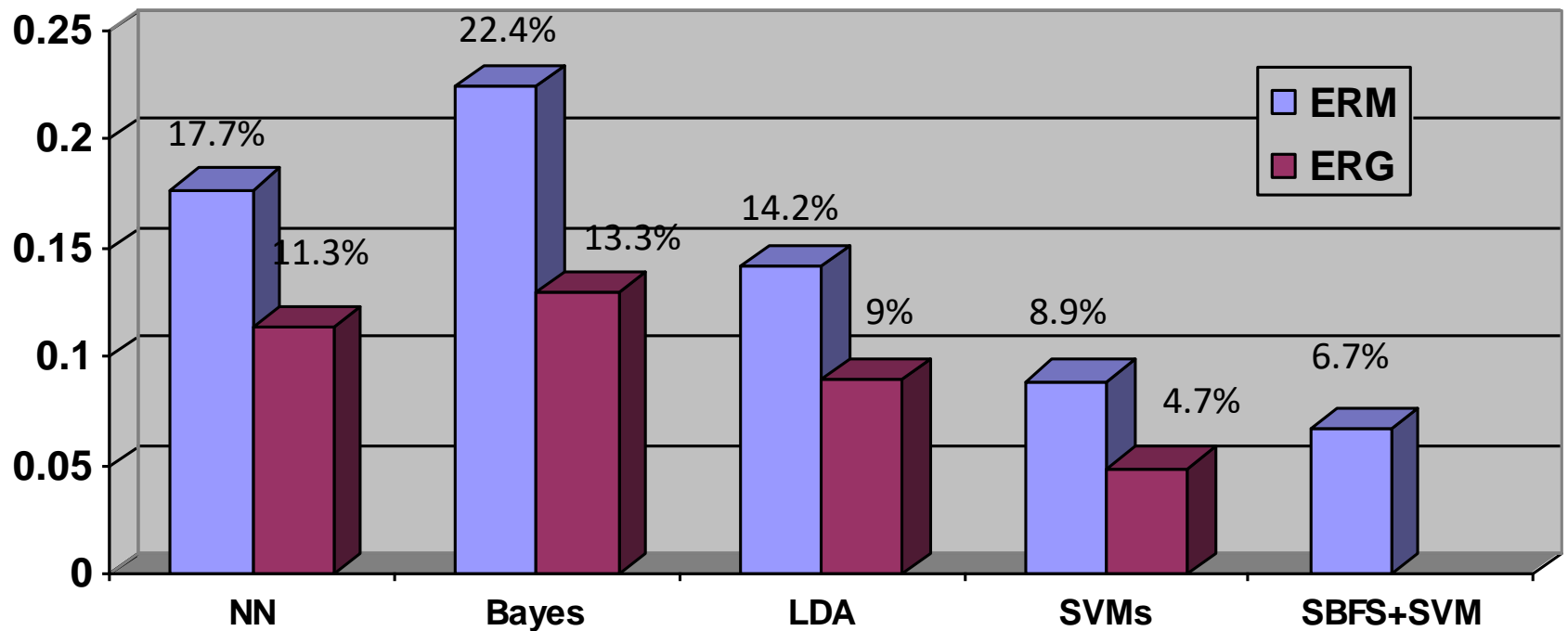


- Used three-fold cross validation

- Training set: 75% of the data
- Validation set: 12.5% of the data
- Test set: 12.5% of the data

- Compared GAs with SFBS

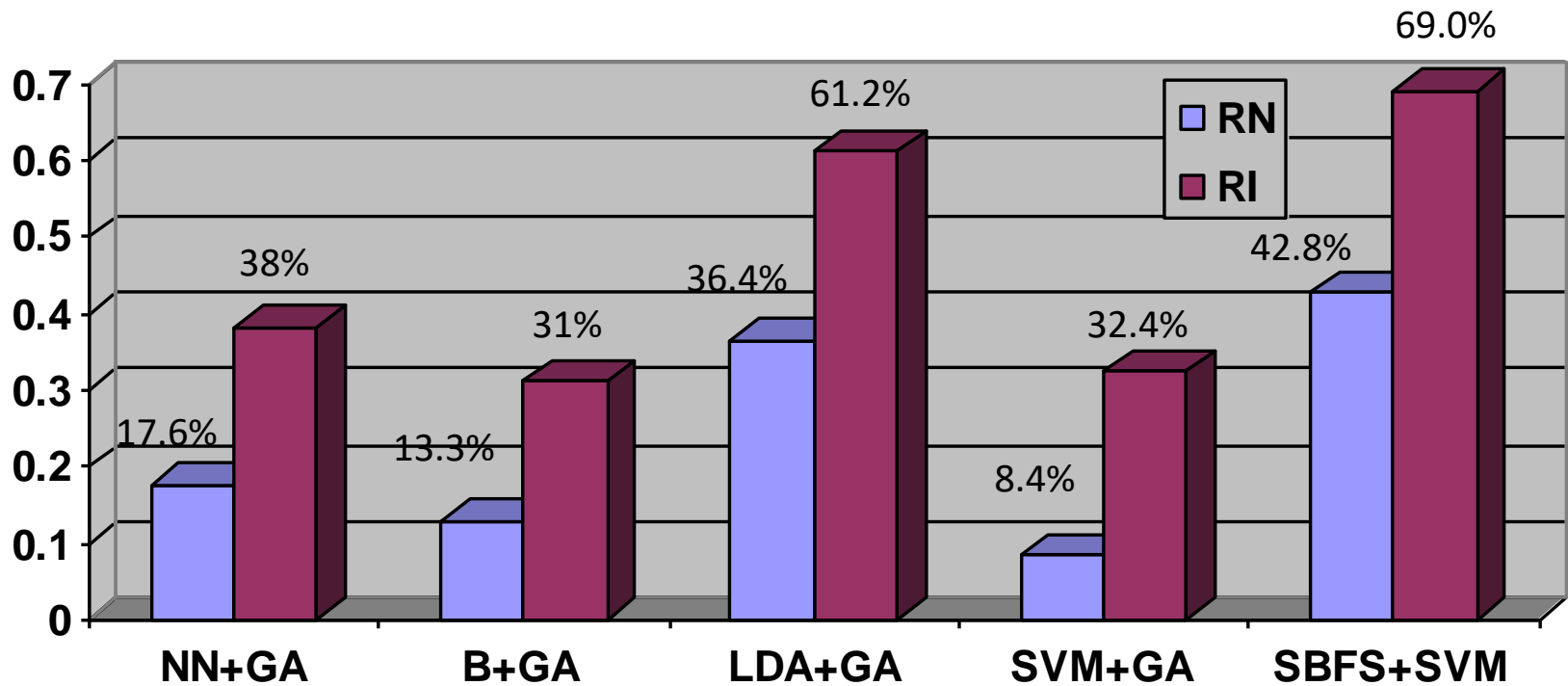
Error Rates



ERM: error rate using top eigenvectors

ERG: error rate using GA selected eigenvectors

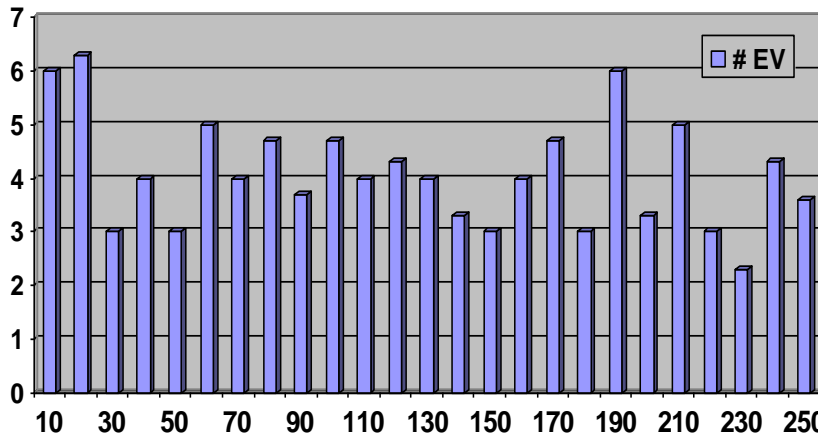
Ratio of Features - Information Kept



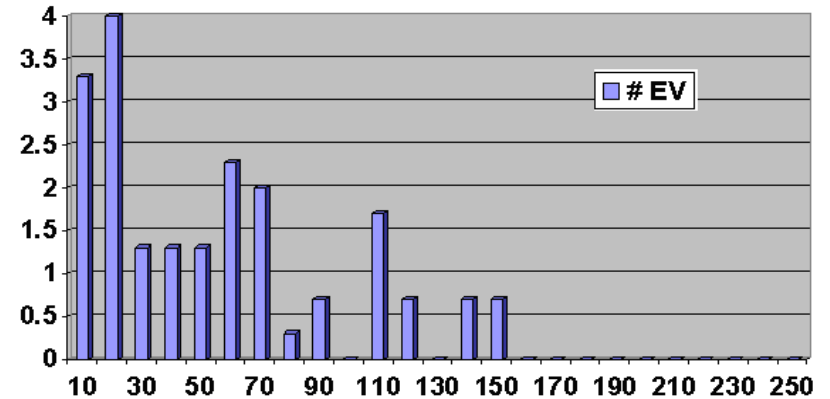
RN: percentage of eigenvectors selected.

RI: percentage of **information** contained in the eigenvector subset selected.

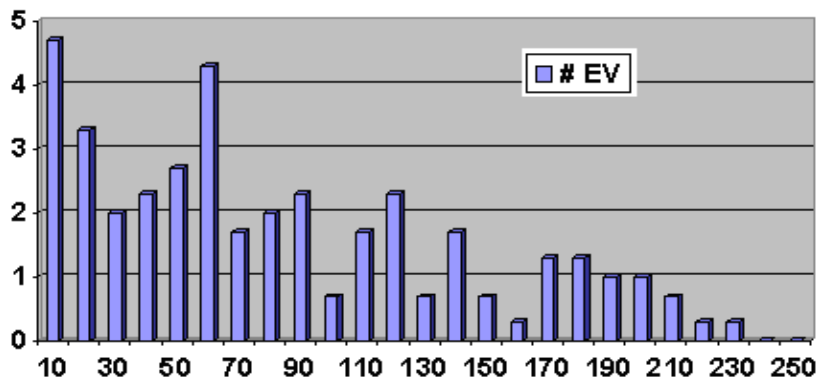
Histograms of Selected Eigenvectors



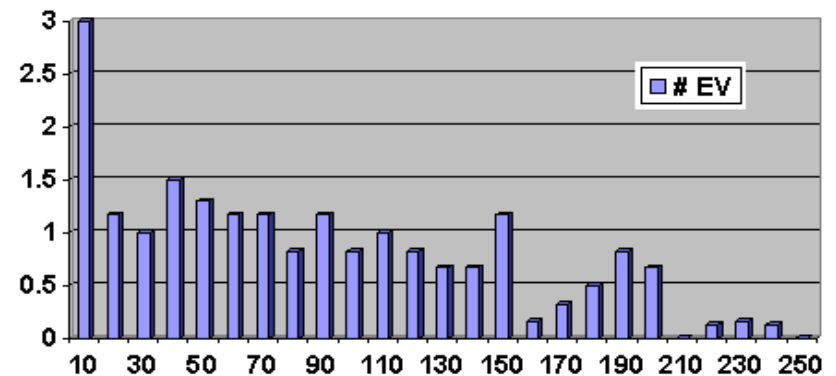
(a) LDA



(b) Bayes



(c) NN



(d) SVMs

Reconstructed Images

Original images



Using top 30 EVs



Using EVs selected by NN+GA



Using EVs selected by SVM+GA



Reconstructed faces using GA-selected EVs do not contain information about identity but **disclose** strong gender information!

Comparison with SFBS

Original images



Top 30 EVs



EVs selected
by SVM+GA



EVs selected
by SVM+SFBS



Case Study 2: Vehicle Detection

Z. Sun, G. Bebis, and R. Miller, "Object Detection Using Feature Subset Selection", **Pattern Recognition**, vol. 37, pp. 2165-2176, 2004.



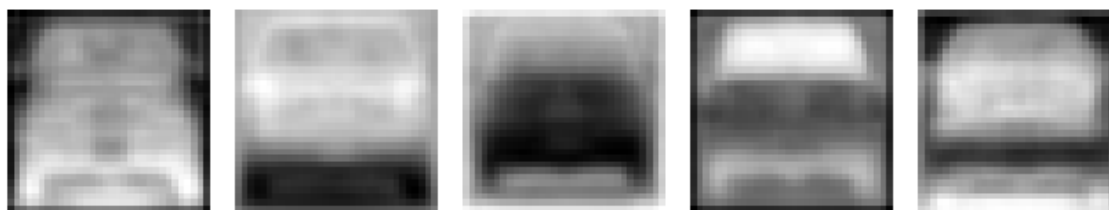
Ford Motor Company

rear views

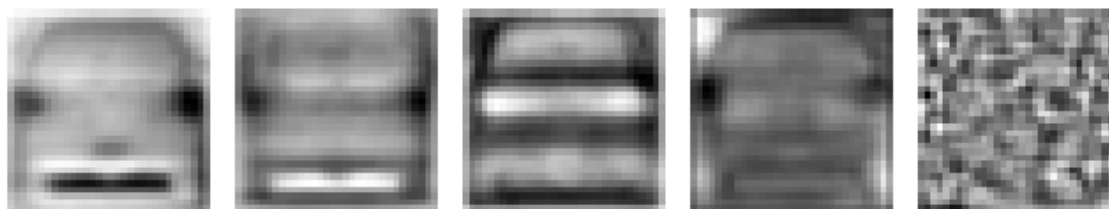


Non-vehicles class much larger than vehicle class.

Which eigenvectors encode the most important vehicle features?



EV#1 EV#2 EV#3 EV#4 EV#5



EV#8 EV#10 EV#12 EV#15 EV#150

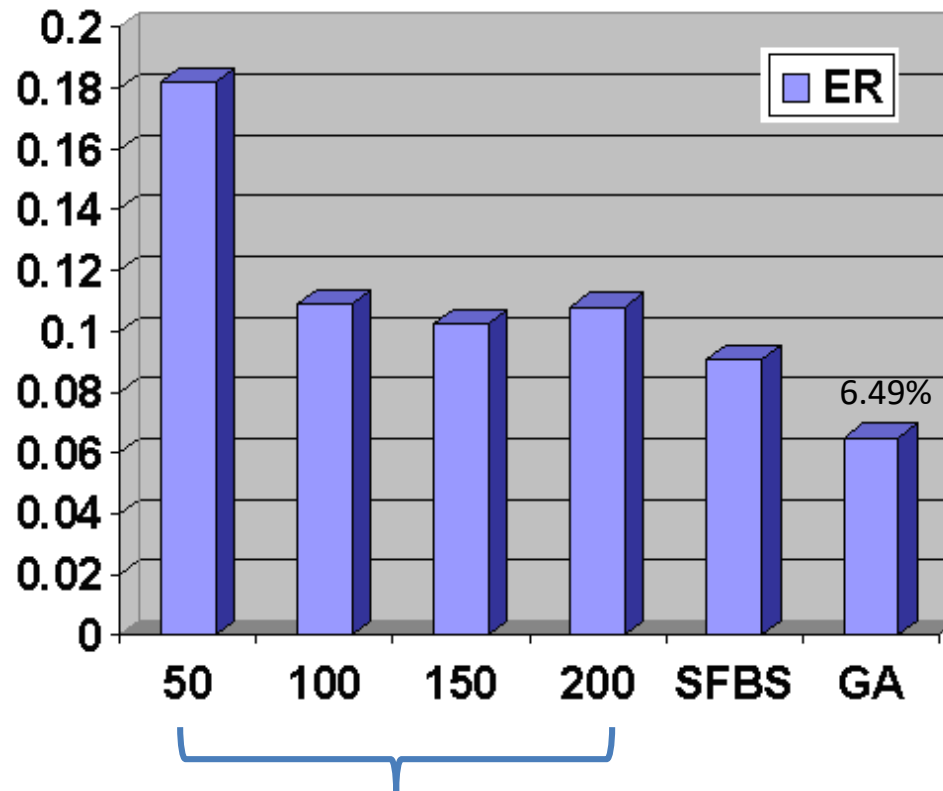
Experiments

- Training data set (collected in **Fall 2001**)
 - 2102 images (1051 vehicles and 1051 non-vehicles)
- Test data sets (collected in **Summer 2001**)
 - 231 images (vehicles and non-vehicles)



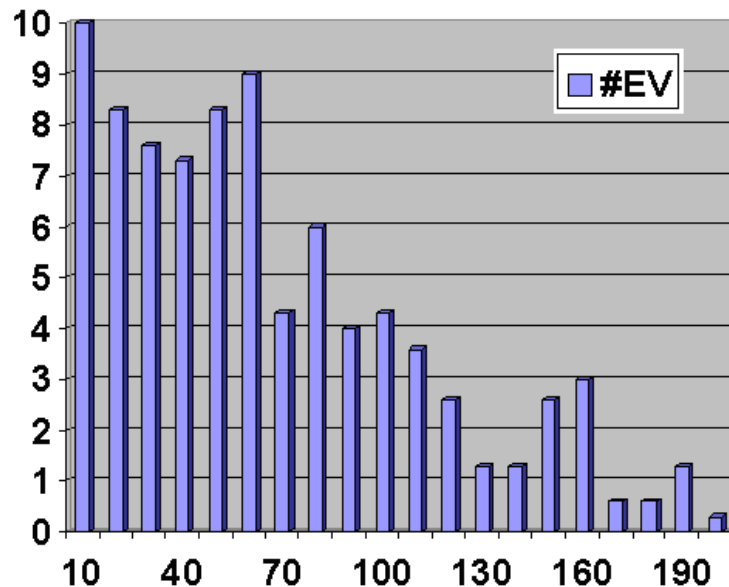
- SVM for classification
- Three-fold cross-validation
- Comparison with SFBS

Error Rate



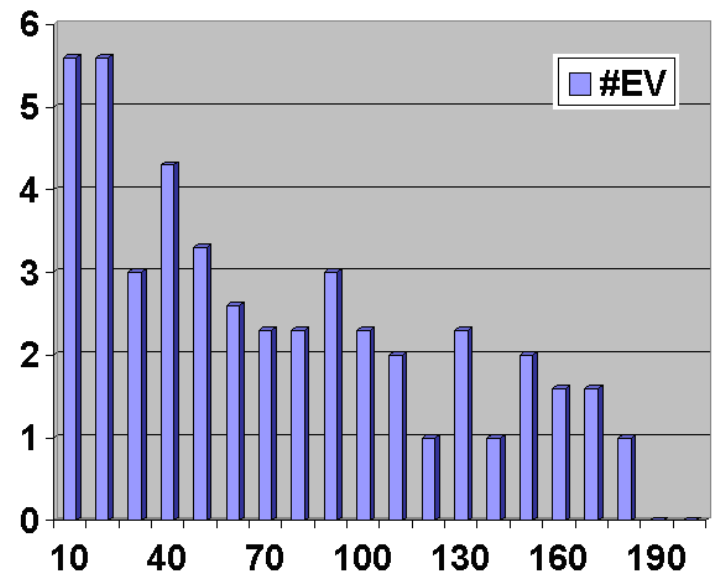
Using "top" eigenvectors

Histograms of Selected Eigenvectors



SFBS-SVM

Number of eigenvectors
selected by SBFS: **87**
(43.5% information)



GA-SVM

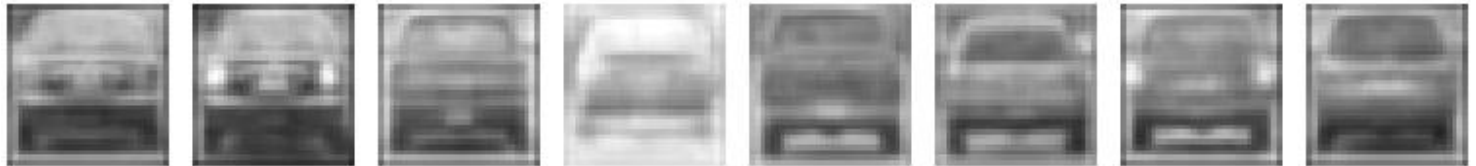
Number of eigenvectors
selected by GA: **46**
(23% information)

Vehicle Detection

Original



Top 50 EVs



EVs selected
by SFBS



EVs selected
by GAs



Reconstructed images using the selected feature subsets.

- Lighting differences have been disregarded by the GA approach.

Case Study 3:

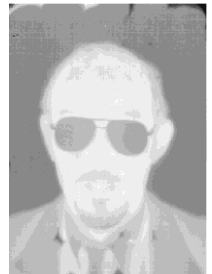
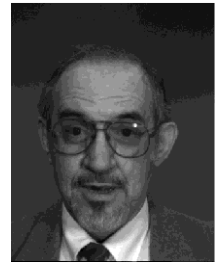
Fusion of Visible-Thermal IR Imagery for Face Recognition

- Improve face recognition performance by integrating information both from the **visible** and **infrared** spectrum.

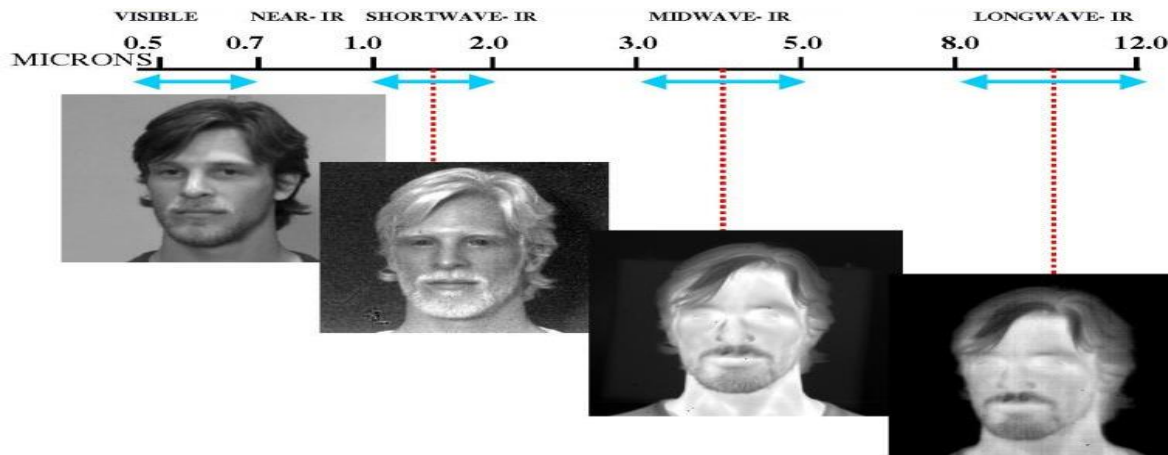
G. Bebis, A. Gyaourova, S. Singh, and I. Pavlidis, "Face Recognition by Fusing Thermal Infrared and Visible Imagery", **Image and Vision Computing**, vol. 24, no. 7, pp. 727-742, 2006.

Visible vs Thermal IR

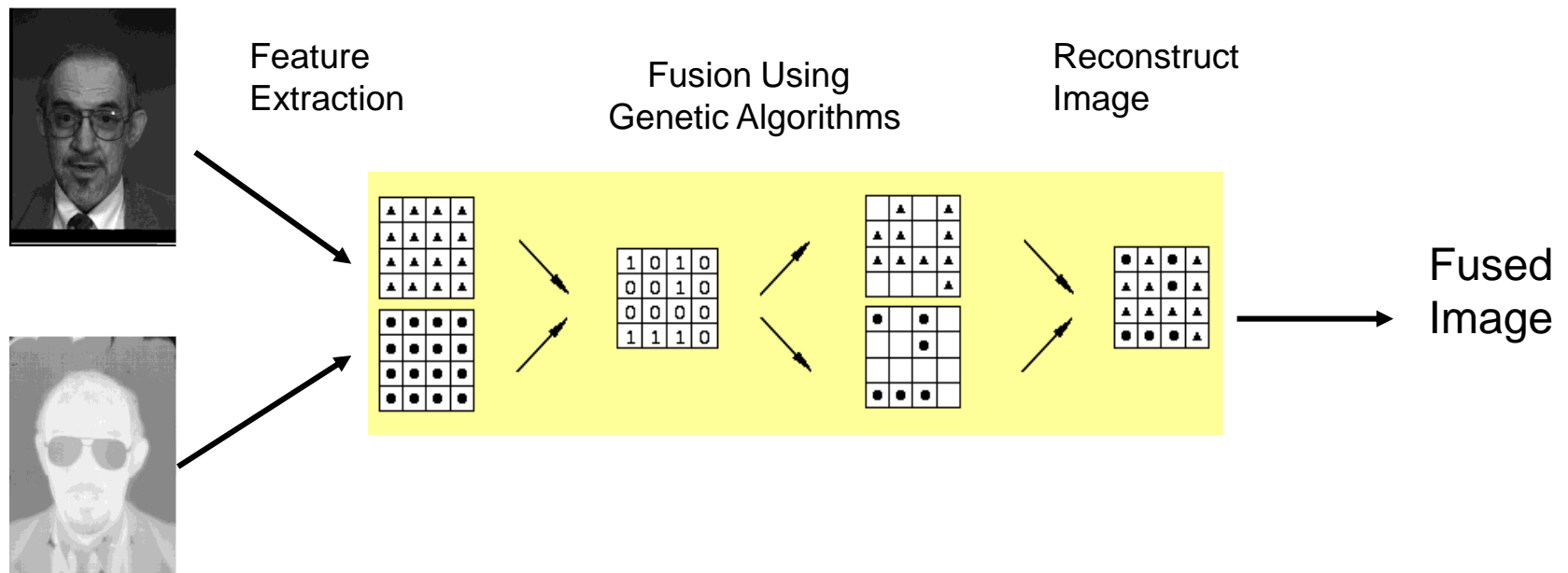
- **Visible** spectrum
 - High resolution, sensitive to changes in illumination and facial expression.
 - Less sensitive to the presence of eyeglasses.
- **Thermal IR** spectrum
 - Robust to illumination changes and facial expressions.
 - Low resolution, face heat patterns, aging, and the presence of eyeglasses.



glass is
opaque to
thermal IR



How should we fuse information from visible and thermal IR?



Tested two feature extraction methods:
- PCA and Wavelets

Dataset



Equinox database

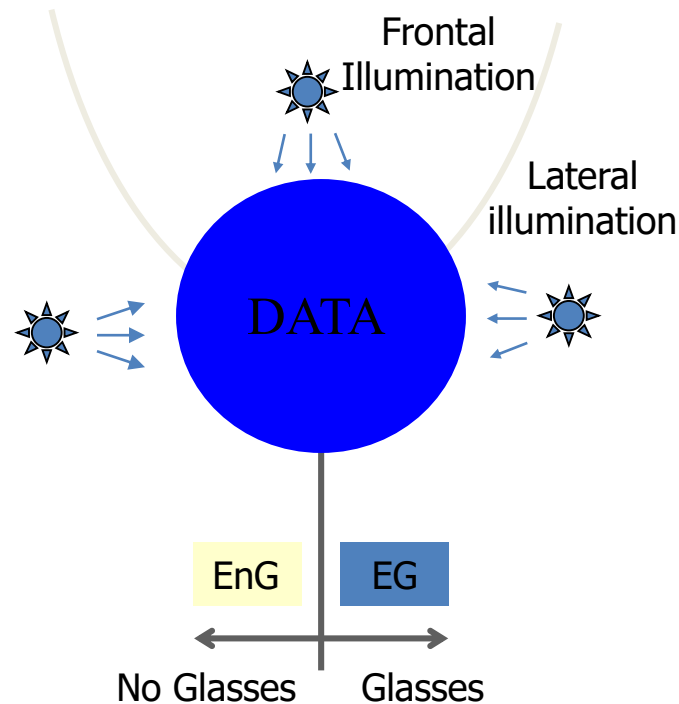
Co-registered thermal infrared and visible images.







Variations among images:

- Illumination (front, left, right)
- Facial expression (smile, frown, surprise and vowels)
- Eyeglasses

Test 1: Eyeglasses/Illumination



Experiments

		test	glasses			no glasses		
training		↓	EG	ELG	EFG	EnG	ELnG	EFnG
	glasses	EG						
		ELG						
		EFG						
	no glasses	EnG						
		ELnG						
		EFnG						



Eyeglasses Tests

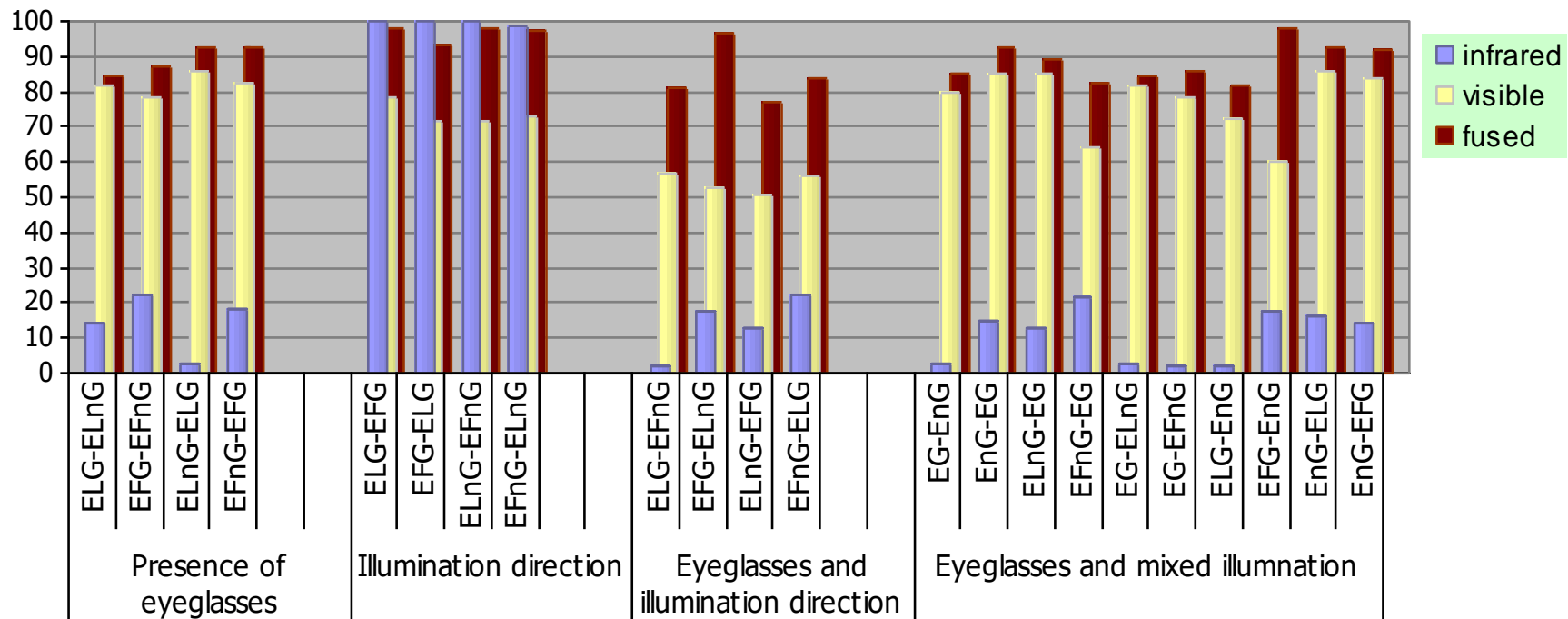


Eyeglasses & Illumination Tests

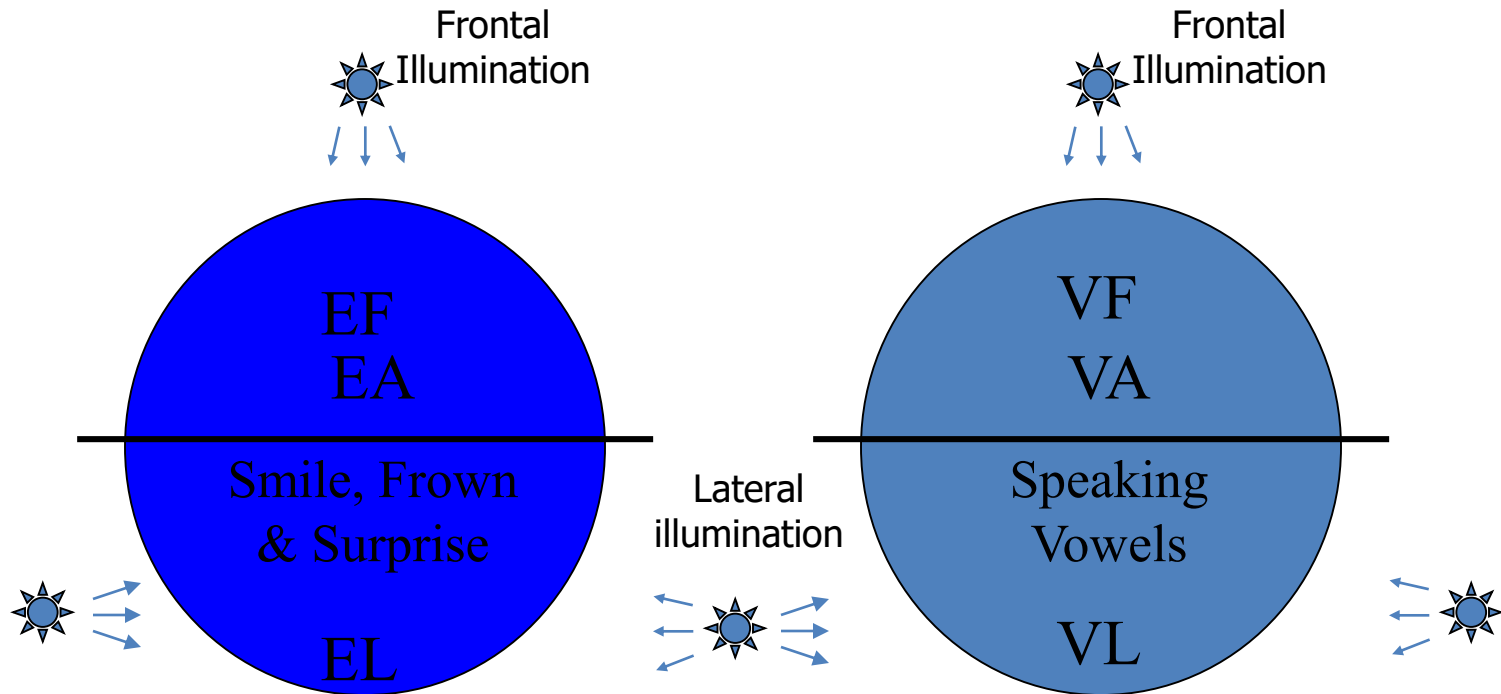
$$EG = ELG \cup EFG, \quad EnG = ELnG \cup EFnG \quad \text{and}$$

$$EG \cap EnG = \emptyset$$

Results



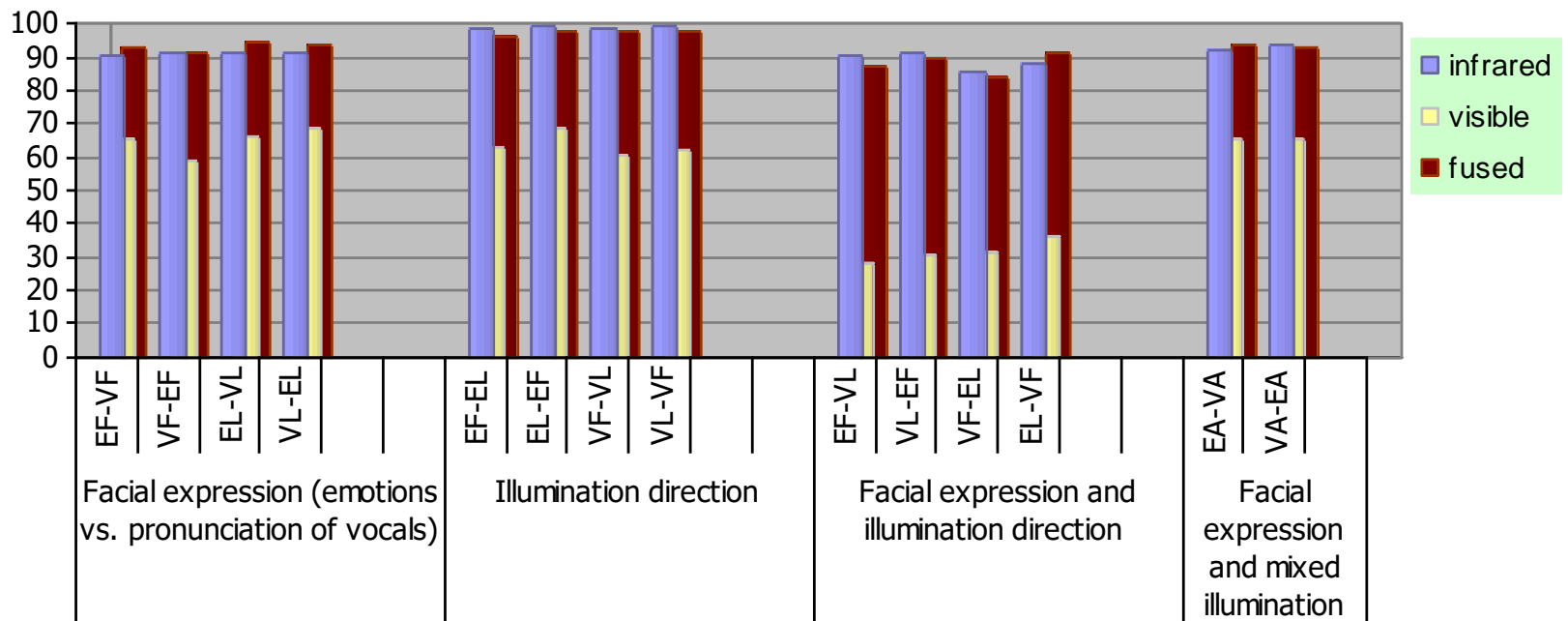
Test 2: Facial Expression



$$EA = EL \cup EF, \quad VA = VL \cup VF \quad \text{and}$$

$$VA \cap EA = \emptyset$$

Results



Overall Accuracy - Eyeglasses

Overall accuracy – eyeglasses

Visible: mean=**66%**, SD=**16%**.

LWIR: mean=**53%**, SD=**42%**.

Fused (wavelet domain): mean=**91%**, SD=**6%**.

Fused (eigenspace domain): mean=**83%**, SD=**12%**.

Overall Accuracy – Facial Expressions

Overall accuracy – facial expressions

Visible: mean=**54.6%**, SD=**15.7%**.

LWIR: mean=**93%**, SD=**4.3%**.

Fused (wavelet domain): mean=**93%**, SD=**4%**.

Fused (eigenspace domain): mean=**92.8%**, SD=**4.6%**.