Feature Selection

- In many applications, we often encounter a very large number of potential features that can be used
- Which subset of features should be used for the best classification?
- Need for a small number of discriminative features
  - To avid “curse of dimensionality”
  - To reduce feature measurement cost
  - To reduce computational burden
- Given an nxd pattern matrix (n patterns in d-dimensional feature space), generate an nxm pattern matrix, where m << d

Feature Selection vs. Extraction

- Both are collectively known as dimensionality reduction
- Selection: choose a best subset of size m from the available d features
- Extraction: given d features (set *Y*), extract m new features (set *X*) by linear or non-linear combination of all the d features
  - Linear feature extraction: $X = TY$, where T is a mxd matrix
  - Non-linear feature extraction: X = f(Y)
- New features by extraction may not have physical interpretation/meaning
- Examples of linear feature extraction
  - Unsupervised: PCA; Supervised: LDA/MDA
- Criteria for selection/extraction: either improve or maintain the classification accuracy, simplify classifier complexity

Feature Selection

- How to find the best subset of size m?
- Recall, best means classifier based on these m features has the lowest probability of error of all such classifiers
- Simplest approach is to do an exhaustive search; computationally prohibitive
  - For d=24 and m=12, there are about 2.7 million possible feature subsets! Cover & Van Campenhout (IEEE SMC, 1977) showed that to guarantee the best subset of size m from the available set of size d, one must examine all possible subsets of size m
- Heuristics have been used to avoid exhaustive search
- How to evaluate the subsets?
  - Error rate; but then which classifier should be used?
  - Distance measure; Mahalanobis, divergence,…
- Feature selection is an optimization problem

Feature Selection: Evaluation, Application, and Small Sample Performance (Jain & Zongker, IEEE Trans. PAMI, Feb 1997)

- Value of feature selection in combining features from different data models
- Potential difficulties feature selection faces in small sample size situation
- Let Y be the original set of features and X is the selected subset
- Feature selection criterion function for the set X is J(X); large values of J indicates better feature subset; problem is to find subset X such that

$$J(X) = \max_{Z \subseteq Y, |Z| = d} J(Z)$$

Taxonomy of Feature Selection Algorithms

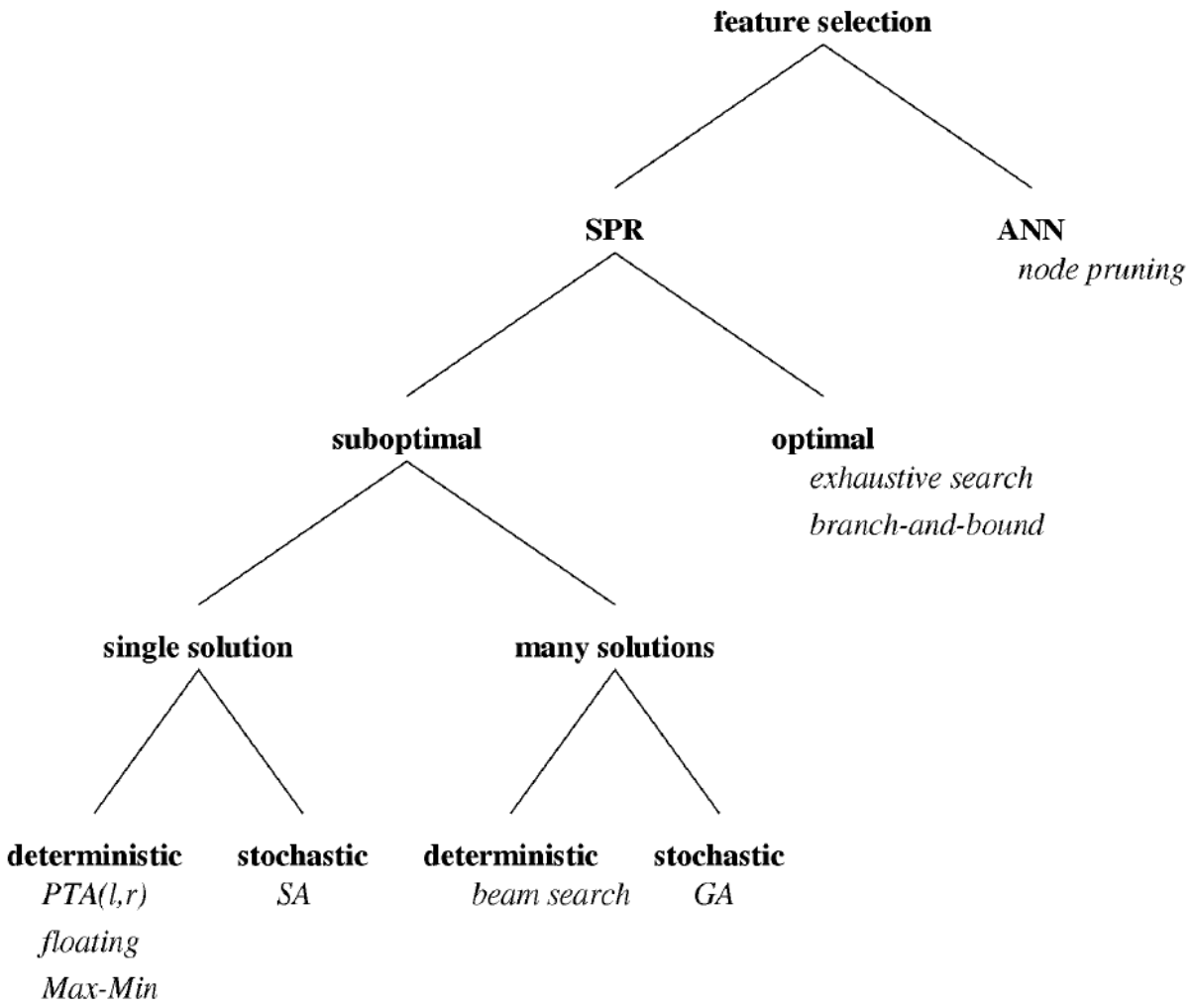


Fig. 1. A taxonomy of feature selection algorithms.

Deterministic Single-Solution Methods

- Begin with a single solution (feature subset) & iteratively add or remove features until some termination criterion is met
- Also known as sequential methods; most popular
  - Bottom up/forward methods: begin with an empty set & add features
  - Top-down/backward methods: begin with a full set & delete features
- Since they do not examine all possible subsets, no guarantee of finding the optimal subset
- Pudil introduced two floating selection methods: SFFS, SFBS
- 15 feature selection methods listed in Table 1 were evaluated

TABLE 1
FEATURE SELECTION ALGORITHMS USED
IN EXPERIMENTAL EVALUATION

SFS	SBS	GSFS(2)	GSBS(2)
GSFS(3)	GSBS(3)	SFFS	SFBS
PTA((1), (2))	PTA((1), (3))	PTA((2), (3)	
BB	MM	GA	NP

Sequential Forward Selection (SFS)

- Start with empty set, *X=0*
- Repeatedly add most significant feature with respect to *X*
- *Disadvantage: Once a feature is retained, it cannot be discarded; nesting problem*

Sequential Backward Selection (SBS)

- Start with full set, $X=Y$
- Repeatedly delete least significant feature in X
- *Disadvantage: SBS requires more computation than SFS; Nesting problem*

Generalized Sequential Forward Selection (GSFS(*m*))

- Start with empty set, $X=0$
- Repeatedly add most significant *m*-subset of ($Y - X$) (found through exhaustive search)

Generalized Sequential Backward Selection (GSBS(*m*))

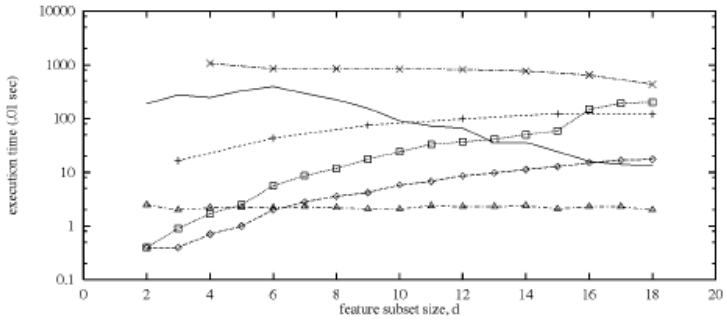
- Start with empty set, $X=Y$
- Repeatedly delete least significant *m*-subset of X (found through exhaustive search)

Sequential Forward Floating Selection (SFFS)

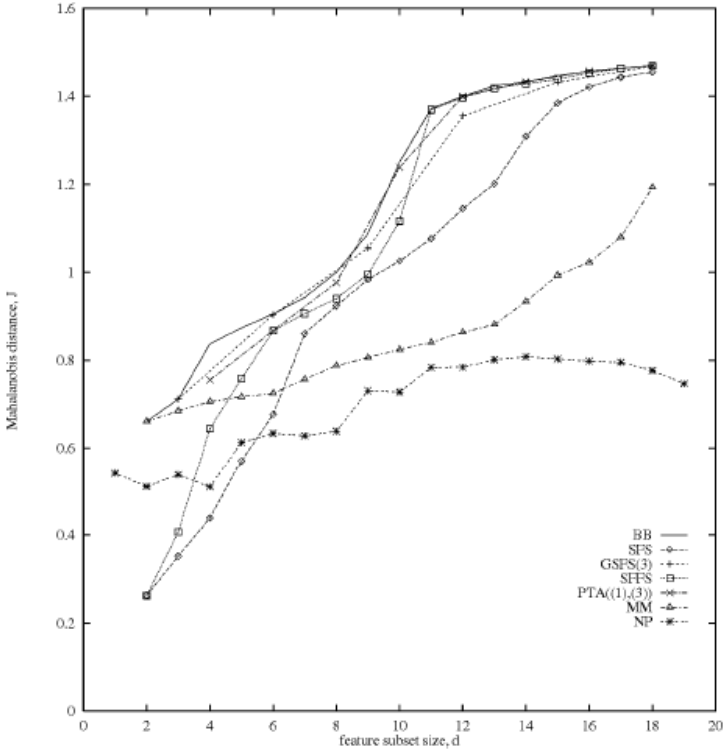
- **Step 1: Inclusion**. Select the most significant feature with respect to *X* and add it to *X*. Continue to step 2.
- **Step 2: Conditional exclusion**. Find the least significant feature *k* in *X*. If it is the feature just added, then keep it and return to step 1. Otherwise, exclude the feature *k*. Note that *X* is now better than it was before step 1. Continue to step 3.
- **Step 3: Continuation of conditional exclusion**. Again find the least significant feature in *X*. If its removal will (a) leave *X* with at least 2 features, and (b) the value of *J(X)* is greater than the criterion value of the best feature subset of that size found so far, then remove it and repeat step 3. When these two conditions cease to be satisfied, return to step 1.

Experimental Results

- 20-dimensional 2-class Gaussian data with the same covariance matrix
- Goodness of features is measured by Mahalanobis distance
- Forward search methods are faster than its backward counterpart
- Performance of floating method is comparable to Branch & bound methods, but they are faster



Classification Performance



Executation Time

Fig. 2. Performance and execution times of selected algorithms on synthetic 2-class Gaussian data set.

Selection of Texture Features

- Selection of texture features for classifying Synthetic Aperture Radar (SAR) images
- A total of 18 different features were extracted from 4 different models
- Can classification error be reduced by feature selection
- 22,000 samples (pixels) from 5 classes; equally split for training & test

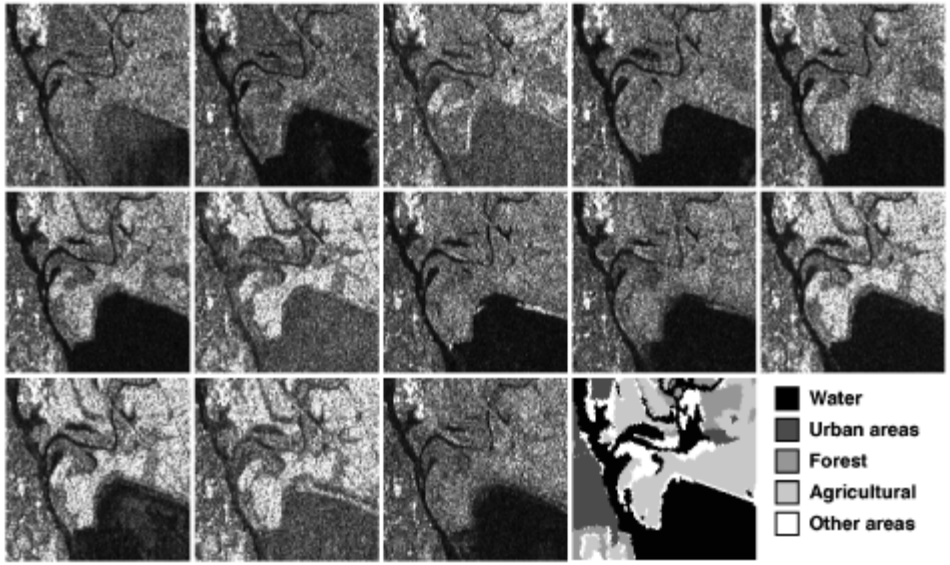


Fig. 3. Sample SAR images, with corresponding ground truth image at lower right.

TABLE 2
SET OF 18 TEXTURE FEATURES FROM FOUR DIFFERENT MODELS

#	feature	model
1	mean	local statistics
2	θ_1	MAR
3	θ_2	MAR
4	θ_3	MAR
5	σ (variance)	MAR
6	mean (logarithmic)	MAR
7	angular second moment	GLCM
8	contrast	GLCM
9	inverse difference moment	GLCM
10	entropy	GLCM
11	inertia	GLCM
12	cluster shade	GLCM
13	power-to-mean ratio	local statistics
14	skewness	local statistics
15	kurtosis	local statistics
16	contrast (from Skriver, 1987)	local statistics
17	lacunarity	fractal
18	dimension	fractal

Performance of SFFS on Texture Features

- Best individual texture model for this data is the MAR model
- Pooling features from different models and then applying feature selection results in an accuracy of 89.3% by 1NN method
- The selected subset has representative feature from every model

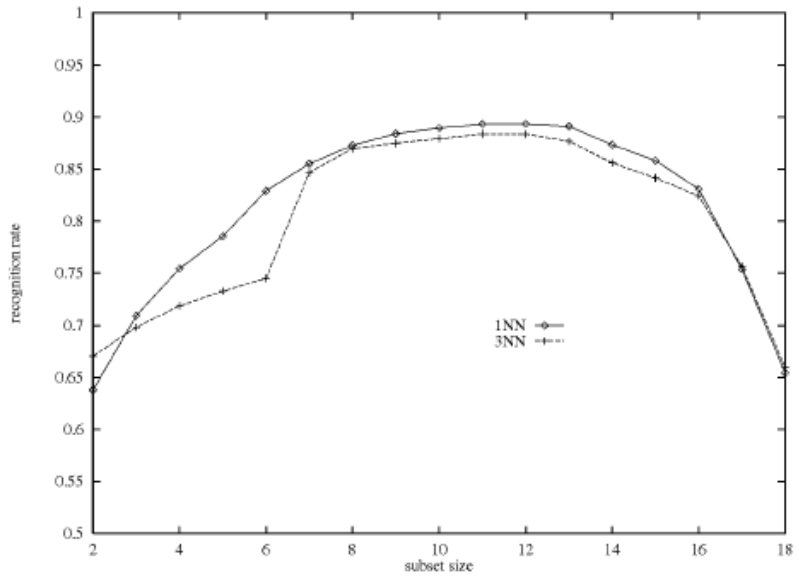


Fig. 4. Recognition rates of SFFS method on texture features.

TABLE 3
BEST CLASSIFICATION ACCURACY ACHIEVED BY THE SFFS FEATURE SELECTION METHOD

Classifier	Recognition rate (%)	Optimal number of features
1NN	89.3	12
3NN	88.4	11

Effect of Training Set Size on Feature Selection

- Suppose the criterion function is Mahalanobis distance; how would the error is estimating the covariance matrix under small sample size will affect the feature selection performance
- Run feature selection on the Trunk data with varying sample size
- 20-dim data from distributions in (2) and (3); n varied from 10 to 5,000
- Feature selection quality: no. of common features in the subset selected by SFFS and by the optimal method
- For n=20, B&B selected the subset {1,2,4,7,9,12,13,14,15,18}; optimal subset is {1,2,3,4,5,6,7,8,9,10}

$$p(\mathbf{x}\,|\,\omega_1) \sim N(\mu, I) \quad p(\mathbf{x}\,|\,\omega_2) \sim N(-\mu, I) \tag{2}$$

$$\mu = \left[1 \quad \frac{1}{\sqrt{1}} \quad \frac{1}{\sqrt{2}} \quad \frac{1}{\sqrt{3}} \quad \cdots\right]^t \tag{3}$$

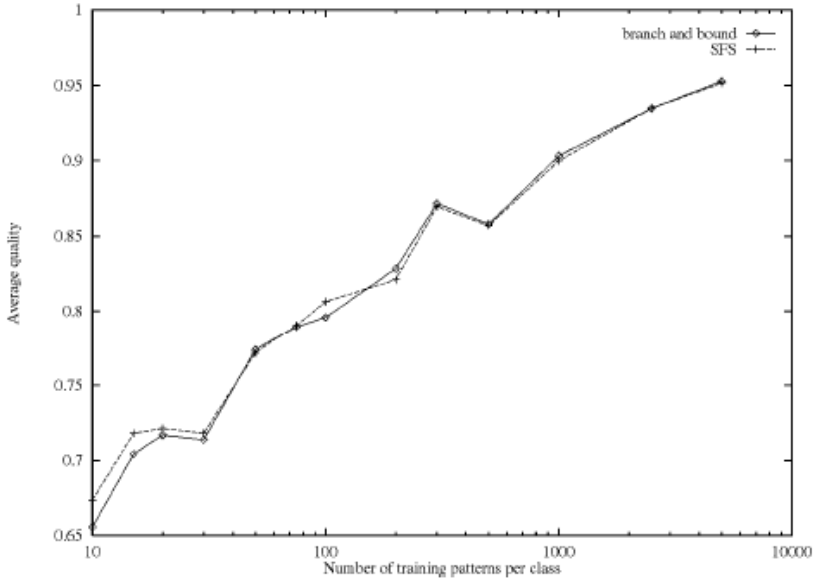


Fig. 5. Quality of selected feature subsets as a function of the size of training data.