

# Operational Feature Selection in Gaussian Mixture Models

## Final internship ENSTA - M2 AIC

A. Lagrange

Supervisor: Mathieu Fauvel (DYNAFOR - INRA/ENSAT), Manuel Grizonnet (CNES)

September 13, 2016

# Outline

## Introduction

- Remote sensing images
- High dimensionality challenge

## Gaussian Mixture Model classifier

- Gaussian Mixture Model
- Assess model quality

## Feature selection

- Sequential Forward Features Selection
- Implementation

## Experimental results

- Hyperspectral dataset
- Heterogeneous features

## Conclusions and perspectives

## Introduction

Remote sensing images

High dimensionality challenge

## Gaussian Mixture Model classifier

Gaussian Mixture Model

Assess model quality

## Feature selection

Sequential Forward Features Selection

Implementation

## Experimental results

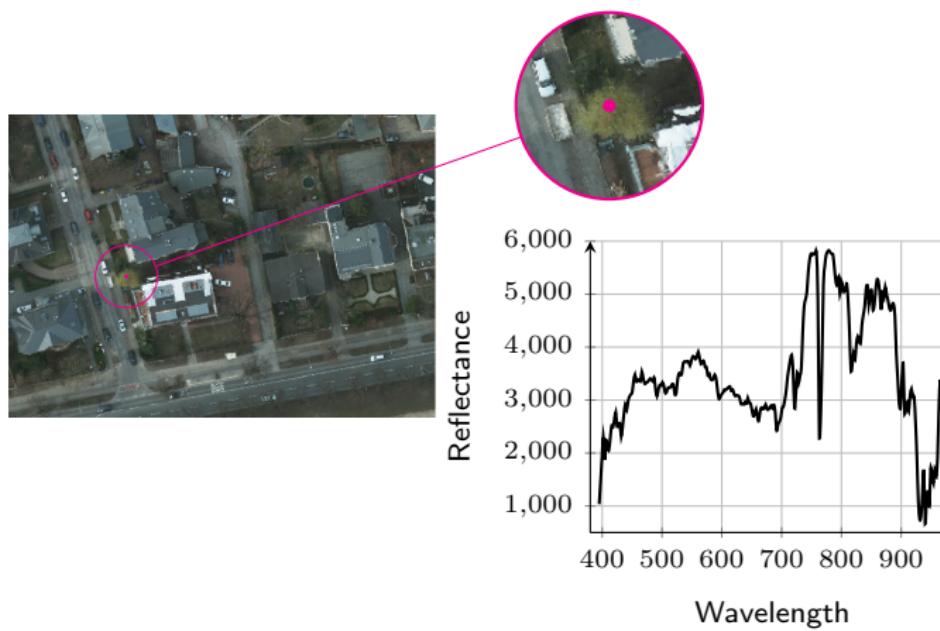
Hyperspectral dataset

Heterogeneous features

## Conclusions and perspectives

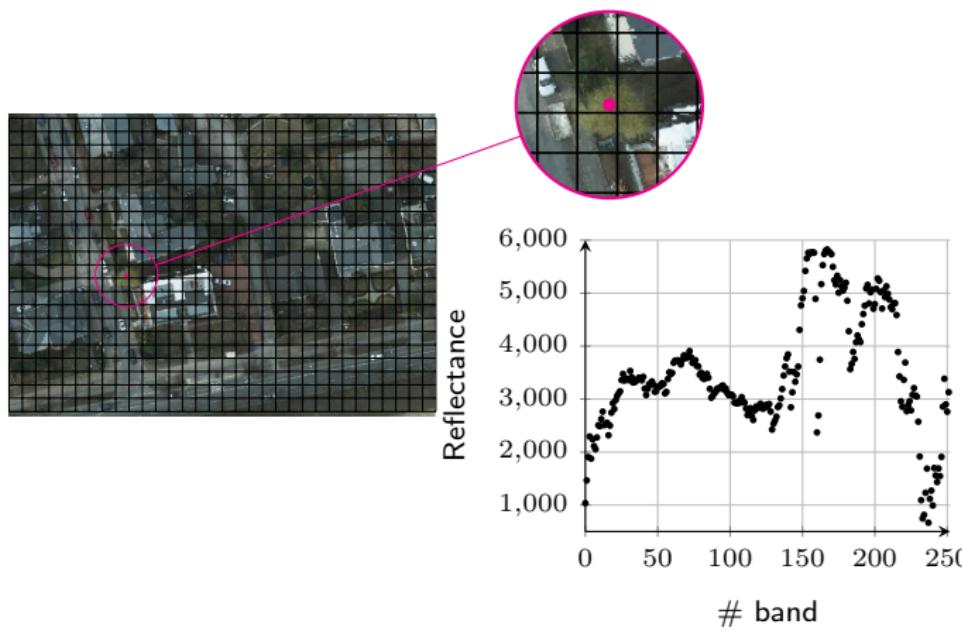
## Remote sensing images

A remote sensing hyperspectral image is a sampling of a spatial, spectral and temporal signal.



## Remote sensing images

A remote sensing hyperspectral image is a sampling of a spatial, spectral and temporal signal.



## Pixelwise image classification in high dimensional space

- Statistical issue (curse of dimensionality): **high number of parameters** to estimate. For Gaussian Mixture Model,  $\sim d^2$  parameters by class.

*Example: for  $d = 250$ , 31625 parameters*

- Computational issue: **huge quantity of data**. Several terabytes of images produced every day.

## Pixelwise image classification in high dimensional space

- Statistical issue (curse of dimensionality): **high number of parameters** to estimate. For Gaussian Mixture Model,  $\sim d^2$  parameters by class.

*Example: for  $d = 250$ , 31625 parameters*

- Computational issue: **huge quantity of data**. Several terabytes of images produced every day.

### Considered approach

- Feature selection to reduce dimension.

## Introduction

Remote sensing images  
High dimensionality challenge

## Gaussian Mixture Model classifier

Gaussian Mixture Model  
Assess model quality

## Feature selection

Sequential Forward Features Selection  
Implementation

## Experimental results

Hyperspectral dataset  
Heterogeneous features

## Conclusions and perspectives

## Gaussian mixture models

- A sample is the realization of a random vector which distribution is a mixture of  $C$  classes conditioned Gaussian distribution:

$$p(\mathbf{x}) = \sum_{c=1}^C \pi_c \frac{1}{(2\pi)^{\frac{d}{2}} |\boldsymbol{\Sigma}_c|^{\frac{1}{2}}} \exp\left(-\frac{1}{2} (\mathbf{x} - \boldsymbol{\mu}_c)^t \boldsymbol{\Sigma}_c^{-1} (\mathbf{x} - \boldsymbol{\mu}_c)\right).$$

- Parameters computed with the conventional unbiased empirical estimator:

$$\hat{\pi}_c = \frac{n_c}{n}, \quad \hat{\boldsymbol{\mu}}_c = \frac{1}{n_c} \sum_{\{i|y_i=c\}} \mathbf{x}_i,$$

$$\hat{\boldsymbol{\Sigma}}_c = \frac{1}{(n_c-1)} \sum_{\{i|y_i=c\}} (\mathbf{x}_i - \hat{\boldsymbol{\mu}}_c)(\mathbf{x}_i - \hat{\boldsymbol{\mu}}_c)^t.$$

- The decision rule uses the Maximum A Posteriori:

$\mathbf{x}$  belongs to  $c \Leftrightarrow c = \arg \max_{c \in C} p(y = c | \mathbf{x}).$

- Quadratic decision rule is rewritten:

$$Q_c(\mathbf{x}) = -(\mathbf{x} - \boldsymbol{\mu}_c)^t \boldsymbol{\Sigma}_c^{-1} (\mathbf{x} - \boldsymbol{\mu}_c) - \log(|\boldsymbol{\Sigma}_c|) + 2 \log(\pi_c) - d \log(2\pi).$$

## Gaussian mixture models

- A sample is the realization of a random vector which distribution is a mixture of  $C$  classes conditioned Gaussian distribution:

$$p(\mathbf{x}) = \sum_{c=1}^C \pi_c \frac{1}{(2\pi)^{\frac{d}{2}} |\boldsymbol{\Sigma}_c|^{\frac{1}{2}}} \exp\left(-\frac{1}{2} (\mathbf{x} - \boldsymbol{\mu}_c)^t \boldsymbol{\Sigma}_c^{-1} (\mathbf{x} - \boldsymbol{\mu}_c)\right).$$

- Parameters computed with the conventional unbiased empirical estimator:

$$\hat{\pi}_c = \frac{n_c}{n}, \quad \hat{\boldsymbol{\mu}}_c = \frac{1}{n_c} \sum_{\{i|y_i=c\}} \mathbf{x}_i,$$

$$\hat{\boldsymbol{\Sigma}}_c = \frac{1}{(n_c-1)} \sum_{\{i|y_i=c\}} (\mathbf{x}_i - \hat{\boldsymbol{\mu}}_c)(\mathbf{x}_i - \hat{\boldsymbol{\mu}}_c)^t.$$

- The decision rule uses the Maximum A Posteriori:

$\mathbf{x}$  belongs to  $c \Leftrightarrow c = \arg \max_{c \in C} p(y = c | \mathbf{x}).$

- Quadratic decision rule is rewritten:

$$Q_c(\mathbf{x}) = -(\mathbf{x} - \boldsymbol{\mu}_c)^t \boldsymbol{\Sigma}_c^{-1} (\mathbf{x} - \boldsymbol{\mu}_c) - \log(|\boldsymbol{\Sigma}_c|) + 2 \log(\pi_c) - d \log(2\pi).$$

Problem:  $\boldsymbol{\Sigma}_c$  is often badly-conditioned.

## Assess model quality: criterion function

### Divergence measure

- Kullback–Leibler divergence:
 
$$\text{SKL}_{ij} = \frac{1}{2} \text{Tr}(\Sigma_{c_i}^{-1} \Sigma_{c_j} + \Sigma_{c_j}^{-1} \Sigma_{c_i}) - d + \frac{1}{2} (\mu_{c_i} - \mu_{c_j})^t (\Sigma_{c_i}^{-1} + \Sigma_{c_j}^{-1}) (\mu_{c_i} - \mu_{c_j})$$
- Bhattacharyya distance:
 
$$B_{ij} = \frac{1}{8} (\mu_i - \mu_j)^t \left( \frac{\Sigma_i + \Sigma_j}{2} \right)^{-1} (\mu_i - \mu_j) + \frac{1}{2} \ln \left( \frac{|\Sigma_i + \Sigma_j|}{\sqrt{|\Sigma_i||\Sigma_j|}} \right)$$
- Jeffries–Matusita distance:
 
$$\text{JM}_{ij} = \sqrt{2\{1 - \exp[-B_{ij}]\}}$$

### Classification rate

- Overall accuracy:

$$OA = \frac{TP}{n}$$

- Cohen's kappa:

$$\text{Kappa} = \frac{OA - \sum_{c=1}^C \frac{TP_c}{FP_c} \frac{TP_c}{FN_c}}{1 - \sum_{c=1}^C \frac{TP_c}{FP_c} \frac{TP_c}{FN_c}}$$

- Mean of F1-score:

$$\text{F1 Mean} = \frac{2TP}{2TP + FN + FP}$$

where  $n$  is the number of samples, TP stands for True Positive, FN False Negative and FP False Positive.

## Assess model quality: criterion function

### Divergence measure

- Kullback–Leibler divergence:

$$\text{SKL}_{ij} = \frac{1}{2} \text{Tr}(\Sigma_{c_i}^{-1} \Sigma_{c_j} + \Sigma_{c_j}^{-1} \Sigma_{c_i}) - d + \frac{1}{2} (\mu_{c_i} - \mu_{c_j})^t (\Sigma_{c_i}^{-1} + \Sigma_{c_j}^{-1}) (\mu_{c_i} - \mu_{c_j})$$

- Bhattacharyya distance:

$$B_{ij} = \frac{1}{8} (\mu_i - \mu_j)^t \left( \frac{\Sigma_i + \Sigma_j}{2} \right)^{-1} (\mu_i - \mu_j) + \frac{1}{2} \ln \left( \frac{|\Sigma_i + \Sigma_j|}{\sqrt{|\Sigma_i||\Sigma_j|}} \right)$$

- Jeffries–Matusita distance:

$$\text{JM}_{ij} = \sqrt{2\{1 - \exp[-B_{ij}]\}}$$

### Classification rate

- Overall accuracy:

$$OA = \frac{TP}{n}$$

- Cohen's kappa:

$$\text{Kappa} = \frac{OA - \sum_{c=1}^C \frac{TP_c}{FP_c} \frac{TP_c}{FN_c}}{1 - \sum_{c=1}^C \frac{TP_c}{FP_c} \frac{TP_c}{FN_c}}$$

- Mean of F1-score:

$$F1 \text{ Mean} = \frac{2TP}{2TP + FN + FP}$$

where  $n$  is the number of samples, TP stands for True Positive, FN False Negative and FP False Positive.

Strongly dependant of Gaussian hypothesis

Need classification for estimation

## Introduction

Remote sensing images  
High dimensionality challenge

## Gaussian Mixture Model classifier

Gaussian Mixture Model  
Assess model quality

## Feature selection

Sequential Forward Features Selection  
Implementation

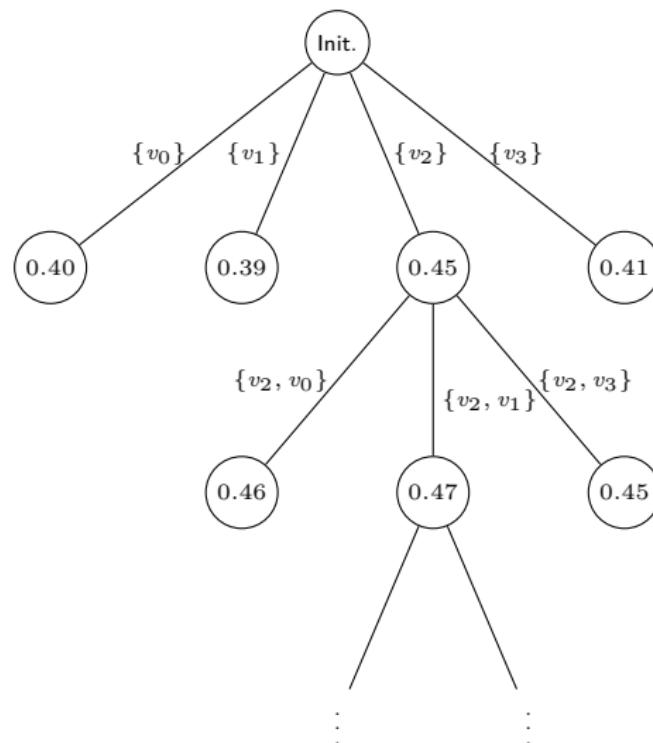
## Experimental results

Hyperspectral dataset  
Heterogeneous features

## Conclusions and perspectives

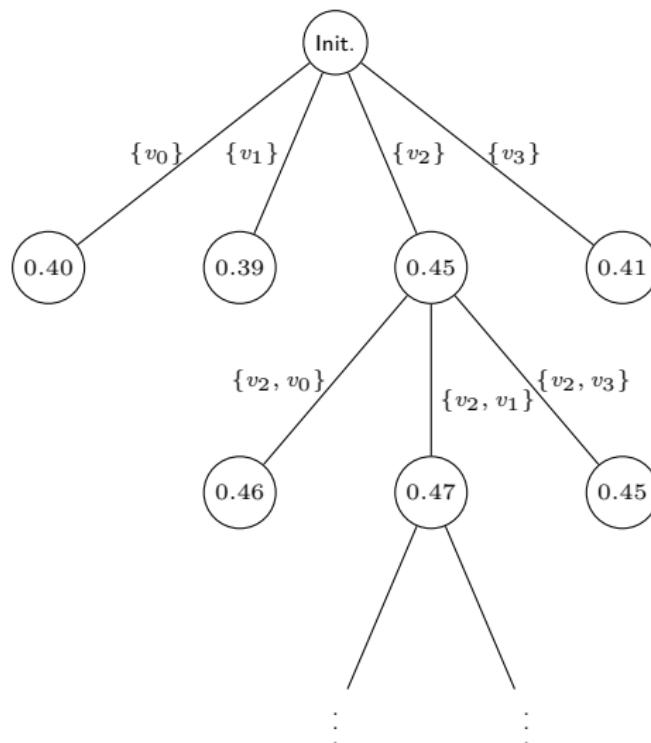
## Sequential Forward Features Selection

- A forward iterative selection method
- Two key elements:
  - ▶ A criterion function to rank subsets of features
  - ▶ A search strategy picking features one by one



## Sequential Forward Features Selection

- A forward iterative selection method
- Two key elements:
  - ▶ A criterion function to rank subsets of features
  - ▶ A search strategy picking features one by one
- Floating forward variation:  
**addition of backward steps**  
 after each forward step



## Update rules for criterion computation

## Update rules for criterion computation

### Inverse update for augmented feature set

$$(\Sigma^{(k)})^{-1} = \begin{bmatrix} (\Sigma^{(k-1)})^{-1} + \frac{1}{\alpha} (\Sigma^{(k-1)})^{-1} \mathbf{u} \mathbf{u}^t (\Sigma^{(k-1)})^{-1} & -\frac{1}{\alpha} (\Sigma^{(k-1)})^{-1} \mathbf{u} \\ -\frac{1}{\alpha} \mathbf{u}^t (\Sigma^{(k-1)})^{-1} & \frac{1}{\alpha} \end{bmatrix}$$

where  $\Sigma^{(k)} = \begin{bmatrix} \Sigma^{(k-1)} & \mathbf{u} \\ \mathbf{u}^t & \sigma_{kk} \end{bmatrix}$  and  $\alpha = \sigma_{kk} - \mathbf{u}^t (\Sigma^{(k-1)})^{-1} \mathbf{u}$ .

## Update rules for criterion computation

### Inverse update for augmented feature set

$$(\Sigma^{(k)})^{-1} = \begin{bmatrix} (\Sigma^{(k-1)})^{-1} + \frac{1}{\alpha} (\Sigma^{(k-1)})^{-1} \mathbf{u} \mathbf{u}^t (\Sigma^{(k-1)})^{-1} & -\frac{1}{\alpha} (\Sigma^{(k-1)})^{-1} \mathbf{u} \\ -\frac{1}{\alpha} \mathbf{u}^t (\Sigma^{(k-1)})^{-1} & \frac{1}{\alpha} \end{bmatrix}$$

where  $\Sigma^{(k)} = \begin{bmatrix} \Sigma^{(k-1)} & \mathbf{u} \\ \mathbf{u}^t & \sigma_{kk} \end{bmatrix}$  and  $\alpha = \sigma_{kk} - \mathbf{u}^t (\Sigma^{(k-1)})^{-1} \mathbf{u}$ .

### Quadratical term update for augmented feature set

$$(\mathbf{x}^{(k)})^t (\Sigma^{(k)})^{-1} \mathbf{x}^{(k)} = (\mathbf{x}^{(k-1)})^t (\Sigma^{(k-1)})^{-1} \mathbf{x}^{(k-1)} + \alpha \left( \begin{bmatrix} \mathbf{v}^t & \frac{1}{\alpha} \end{bmatrix} \mathbf{x}^{(k)} \right)^2$$

where  $(\Sigma^{(k)})^{-1} = \begin{bmatrix} A & \mathbf{v} \\ \mathbf{v}^t & \frac{1}{\alpha} \end{bmatrix}$

### Determinant update for augmented feature set

$$\log(|\Sigma^{(k)}|) = \log(|\Sigma^{(k-1)}|) + \log \alpha$$

## Example: update rule for decision function

$$\begin{aligned}
 Q_c(\mathbf{x}) = & -(\mathbf{x}^{(k-1)} - \boldsymbol{\mu}_c^{(k-1)})^t (\boldsymbol{\Sigma}_c^{(k-1)})^{-1} (\mathbf{x}^{(k-1)} - \boldsymbol{\mu}_c^{(k-1)}) \\
 & - \log(|\boldsymbol{\Sigma}_c^{(k-1)}|) + 2 \log(\pi_c) + k \log(2\pi) \\
 & - \alpha \left( \begin{bmatrix} \mathbf{v}^t & \frac{1}{\alpha} \end{bmatrix} (\mathbf{x}^{(k)} - \boldsymbol{\mu}_c^{(k)}) \right)^2 - \log \alpha
 \end{aligned}
 \quad \left. \begin{array}{l} \text{Computed once for all} \\ \text{Computed for each set} \end{array} \right\}$$

# Production

- A Python module:

[https://github.com/Laadqr/FFFS.](https://github.com/Laadqr/FFFS)

- A C++ module: a remote module for Orfeo Toolbox v5.4 (CNES)

[https://github.com/Laadqr/otbExternalFastFeaturesSelection.](https://github.com/Laadqr/otbExternalFastFeaturesSelection)

The screenshot shows a GitHub repository page for a Python module named 'FFFS'. The repository has 69 commits, 2 branches, and 0 releases. The master branch is 49 commits ahead of orfeotoolbox/master. The latest commit was pushed 19 seconds ago. The commit history includes several changes to app, demo, include, and test files, along with updates to README.org and LICENSE. A file named CMakelists.txt is also present. The repository has 3 contributors. Below the repository details, there is a section titled 'Operational Feature Selection in Gaussian Mixture Models' with a 'General' subsection. This section contains a brief description of the module's purpose and its implementation based on papers from arXiv and PFGDA.

Branch: master • New pull request

69 commits • 2 branches • 0 releases • 3 contributors

This branch is 49 commits ahead of orfeotoolbox/master.

Latest commit: 19 seconds ago

Pull request • Compare

Laadr committed on GitHub Update and rename README to README.org

app add rand seed to app 21 days ago

demo add demo + improve computeClassRate a month ago

include change name of model save for selection 17 days ago

src add demo + improve computeClassRate a month ago

test fix some warnings + change some for to use assertor 28 days ago

.gitignore Adding initial structure for ITKv4 External Modules 5 years ago

CMakelists.txt bugs fixed for selection crossvalidation - change various variables n... 2 months ago

LICENSE Adding initial structure for ITKv4 External Modules 5 years ago

README.org Update and rename README to README.org 13 seconds ago

otb-module.create green training application ok 2 months ago

README.org

### Operational Feature Selection in Gaussian Mixture Models

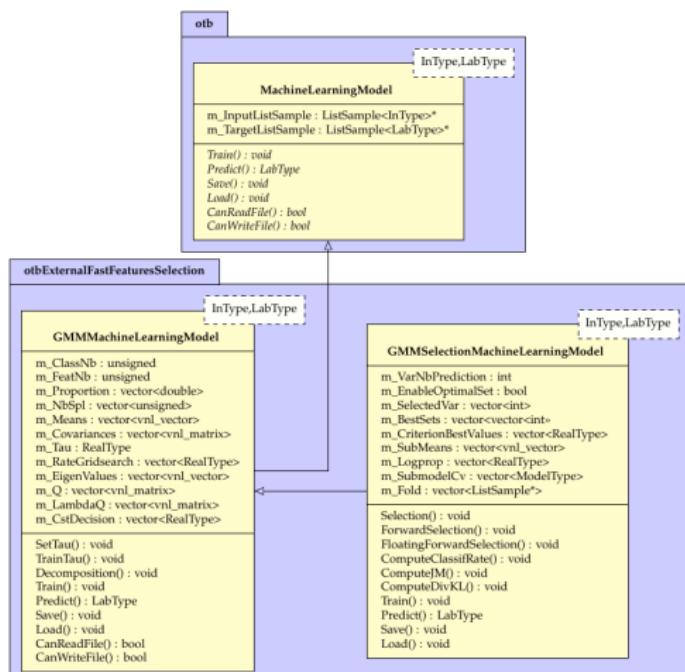
#### General

This is a remote module for the ORFEOToolbox (<https://www.orfeo-toolbox.org>). It is designed to work with OTBv5 modular system and to be places in OTB/Module/Remote.

This module implements a method to perform a fast forward feature selection using a Gaussian Mixture Model. The algorithm is based on the following papers <http://arxiv.org/abs/1501.00857> and Nonlinear parsimonious feature selection for the classification of hyperspectral images ([http://auvel.mathieu.free.fr/pdfs/pgda\\_grsf.pdf](http://auvel.mathieu.free.fr/pdfs/pgda_grsf.pdf)).

## Module OTB: code

- Fork from otb external module template.
- Inherit from MachineLearningModel.



## Module OTB: applications

- **otbcli\_TrainGMMApp**: train a GMM classifier and optionally perform a ridge regularization with an embedded selection of the regularization parameter;
- **otbcli\_TrainGMMSelectionApp**: train a GMM classifier and perform a feature selection algorithm (SFS, SFFS) with various possible criterion (J-M dist., KL div., Cohen's kappa, overall accuracy, mean F1-score);
- **otbcli\_PredictGMMApp**: perform a classification of the input image using a GMM model.

## Module OTB: possible improvement

- Modify training applications to use the new sampling module;
- Parallelize code for training and selection:
  - ▶ model parameters estimation,
  - ▶ cross-validation,
  - ▶ criterion function estimation.

## Introduction

Remote sensing images  
High dimensionality challenge

## Gaussian Mixture Model classifier

Gaussian Mixture Model  
Assess model quality

## Feature selection

Sequential Forward Features Selection  
Implementation

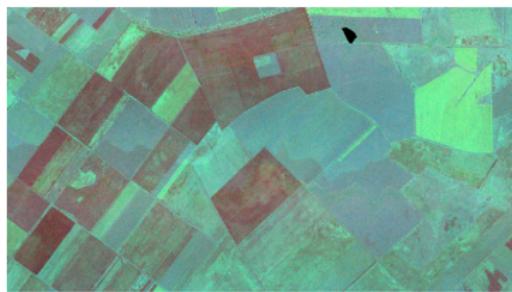
## Experimental results

Hyperspectral dataset  
Heterogeneous features

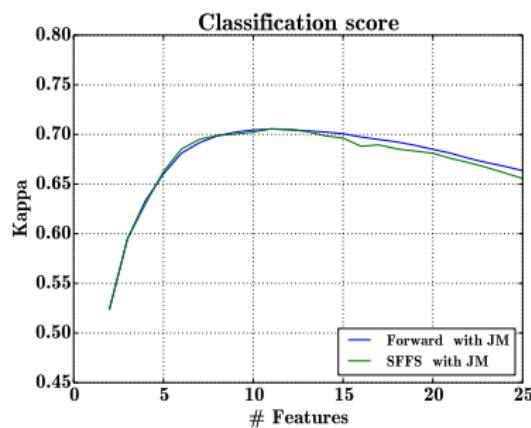
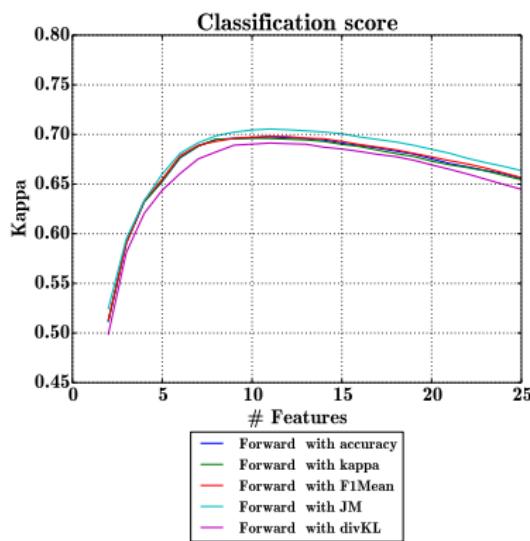
## Conclusions and perspectives

## Aisa dataset

- Hyperspectral image acquired by the AISA Eagle sensor during a flight campaign over Heves, Hungary.
- 252 bands ranging from 395 to 975 nm.
- 16 classes with 361,971 referenced pixels.
- Classification with spatial stratification (separate polygons for training and validation).



## Selection: method variation tested on Aisa



No advantage with floating forward variation.

Best results with Jeffries-Matusita distance.

## Aisa dataset: results

# samples by class	Cohen's kappa			Training time (s)			Classification time (s)		
	250	500	1000	250	500	1000	250	500	1000
GMM SFS kappa	<b>0.684</b>	<b>0.698</b>	<b>0.711</b>	257	496	955	<b>7.7</b>	<b>8.6</b>	<b>8.7</b>
GMM SFS JM	<b>0.680</b>	<b>0.700</b>	<b>0.710</b>	<b>8.6</b>	<b>8.9</b>	<b>9.1</b>	<b>9.7</b>	<b>9.8</b>	<b>9.6</b>
GMM SFFS JM	<b>0.680</b>	<b>0.700</b>	<b>0.710</b>	<b>8.8</b>	<b>9.0</b>	<b>9.3</b>	<b>9.7</b>	<b>9.8</b>	<b>9.8</b>
GMM Ridge	0.611	0.620	0.642	71.7	105	167	530	530	530
KNN	0.551	0.563	0.574	<b>8.9</b>	19.6	59.7	387	639	887
Random Forest	0.645	0.673	<b>0.693</b>	24.5	49.3	105	33.0	41.7	45.9

## Potsdam dataset

- Aerial images of Potsdam urban area distributed by the International Society for Photogrammetry and Remote Sensing (ISPRS).
- Patches of 6000x6000 pixels with a 5cm-resolution and 4 channels (Red, Blue, Green and Infrared) + Digital Surface Model (DSM).
- 6 classes: Low vegetation, High vegetation, Impervious surfaces, Buildings, Cars, Clutter.



## Potsdam dataset: extracted features

- Radiometric indexes: NDVI, TNDVI, RVI, SAVI, TSAVI, MSAVI, MSAVI2, GEMI, IPVI, NDWI2, NDTI, RI, CI, BI, BI2 (15 features);
- Morphological profile with reconstruction of each band with a disk of radius 5, 9, 13, 17, 21, 25, 29, 33, 37 and 41 (80 features);
- Attribute profile of each band with area as attribute and 1000, 2000, 5000, 10000 and 15000 as thresholds (40 features);
- Attribute profile of each band with diagonal of bounding box as attribute and 100, 200, 500, 1000 and 20000 as thresholds (40 features);
- Textural features with neighborhood of 19x19 pixels: mean, standard deviation, range and entropy (16 features);
- DSM and raw image (5 features)

## Potsdam dataset: extracted features

- Radiometric indexes: NDVI, TNDVI, RVI, SAVI, TSAVI, MSAVI, MSAVI2, GEMI, IPVI, NDWI2, NDTI, RI, CI, BI, BI2 (15 features);
- Morphological profile with reconstruction of each band with a disk of radius 5, 9, 13, 17, 21, 25, 29, 33, 37 and 41 (80 features);
- Attribute profile of each band with area as attribute and 1000, 2000, 5000, 10000 and 15000 as thresholds (40 features);
- Attribute profile of each band with diagonal of bounding box as attribute and 100, 200, 500, 1000 and 20000 as thresholds (40 features);
- Textural features with neighborhood of 19x19 pixels: mean, standard deviation, range and entropy (16 features);
- DSM and raw image (5 features)

196 features stacked to create new images with 196 bands.

## Potsdam dataset: results

- Training set: 1000 samples by class

	Cohen's kappa of 5_11 (train)	Cohen's kappa of 5_12 (test)	Training time (s)	Classification time (s)	# features used to classify
GMM SFS kappa	<b>0.698</b> (0.008)	<b>0.679</b> (0.007)	100	300	13.8
GMM SFS JM	0.460 (0.221)	0.453 (0.226)	1	400	29.4
GMM SFFS JM	0.460 (0.221)	0.453 (0.226)	1	400	29.4
GMM ridge	0.521 (0.010)	0.478 (0.014)	10	2000	all
KNN	0.536 (0.005)	0.468 (0.002)	1	1000	all
Random Forest	<b>0.734</b> (0.003)	<b>0.680</b> (0.005)	20	800	all

- Training set: 50000 samples by class

	Cohen's kappa of 5_11 (train)	Cohen's kappa of 5_12 (test)	Training time (s)	Classification time (s)	# features used to classify
GMM SFS kappa	<b>0.710</b> (0.007)	<b>0.681</b> (0.007)	5000	400	24.5
GMM SFS JM	0.461 (0.084)	0.442 (0.088)	5	400	30
GMM SFFS JM	0.476 (0.111)	0.468 (0.108)	5	400	27.6
GMM ridge	0.508 (0.000)	0.467 (0.001)	500	2000	all
Random Forest	<b>0.854</b> (0.001)	<b>0.720</b> (0.001)	2000	2000	all

## Introduction

Remote sensing images  
High dimensionality challenge

## Gaussian Mixture Model classifier

Gaussian Mixture Model  
Assess model quality

## Feature selection

Sequential Forward Features Selection  
Implementation

## Experimental results

Hyperspectral dataset  
Heterogeneous features

## Conclusions and perspectives

## Conclusions and perspectives

- An algorithm for the classification of high dimension data was designed.
  - ▶ GMM model to select iteratively the most relevant features.
  - ▶ Few or no advantages with floating forward variation.
- An efficient implementation available as a remote module of the Orfeo toolbox.
  - ▶ Good performances compared to standard classifiers implemented in Orfeo toolbox.
- Need to investigate robustness to Gaussian hypothesis (Jeffries-Matusita vs kappa).
- Stability of selected bands could be improve by selecting continuous intervals.