Introduction to pySPACE workflows (`https://github.com/pyspace`)

a Signal Processing and Classification Environment written in Python

M M Krell, S Straube, A Seeland, H Wöhrle, J Teiwes, J H Metzen, E A Kirchner, F Kirchner

DFKI Bremen & University of Bremen

Robotics Innovation Center

Director: Prof. Dr. Frank Kirchner

www.dfki.de/robotics

robotics@dfki.de

Supported by:

Federal Ministry
of Economics
and Technology

on the basis of a decision
by the German Bundestag

DLR

pySPACE: Computation of Multiple Workflows

- ...with **applications** in robotics and brain-computer interfaces
- ...with **simple configuration** and **automatic processing** of empirical evaluations
- ...on feature vector and time series datasets
- ...where configuration requires no programming (YAML used)
- ...with execution in a distributed manner (embarrassingly parallel)
- ...intuitive **structure**
- ...choosing from more than 100 signal processing and classification algorithms (additionally interfaces to other libraries)

pySPACE: Computation of Multiple Workflows

- . . . with **applications** in robotics and brain-computer interfaces
- . . . with **simple configuration** and **automatic processing** of empirical evaluations
- . . . on feature vector and time series datasets
- . . . where configuration requires no programming (YAML used)
- . . . with execution in a distributed manner (embarrassingly parallel)
- . . . intuitive **structure**
- . . . choosing from more than 100 signal processing and classification algorithms (additionally interfaces to other libraries)

pySPACE: Computation of Multiple Workflows

- . . . with **applications** in robotics and brain-computer interfaces
- . . . with **simple configuration** and **automatic processing** of empirical evaluations
- . . . on feature vector and time series datasets
- . . . where configuration requires no programming (YAML used)
- . . . with execution in a distributed manner (embarrassingly parallel)
- . . . intuitive **structure**
- . . . choosing from more than 100 signal processing and classification algorithms (additionally interfaces to other libraries)

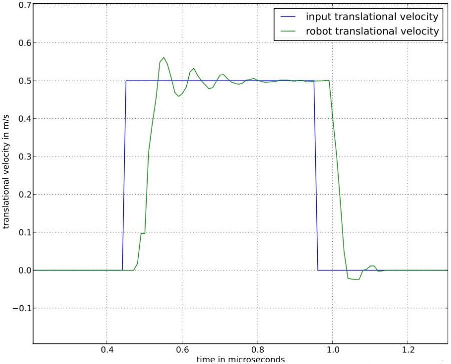
Robotic application in the Project VirGo[4]

Predict sensor values

- e.g. gyro, temperature, battery load

⇒ detect and react to unexpected events

- Methods for predicting upcoming sensor readings are developed



pySPACE is used to:

- process different datasets,
- compare/evaluate different regression algorithms,
- and tune their parameters (e.g. nodes in the hidden layers of a Multilayer perceptron)

other Applications

- evaluation and comparison of
  - ...sensor selection algorithms (on EEG data) [2]
  - ...dimensionality reduction algorithms (ICA, PCA, xDAWN, PiSF, CSP) [3, 4, 5, 13]
  - ...classifiers (BRMM, online classifiers, ...) [6, 11, 14, 15]
- Brain-Computer Interfaces (movement prediction, interaction error detection, detection of warning perception) [1, 7, 8, 9, 10, 16, 17]
- soil detection
- parallelization of robot simulations
- classify iterative closest point (ICP) matches for good and bad *localization*

other Applications

- evaluation and comparison of
  - . . . sensor selection algorithms (on EEG data) [2]
  - . . . dimensionality reduction algorithms (ICA, PCA, xDAWN, PiSF, CSP) [3, 4, 5, 13]
  - . . . classifiers (BRMM, online classifiers, . . .) [6, 11, 14, 15]
- Brain-Computer Interfaces (movement prediction, interaction error detection, detection of warning perception) [1, 7, 8, 9, 10, 16, 17]
- soil detection
- parallelization of robot simulations
- classify iterative closest point (ICP) matches for good and bad *localization*

other Applications

- evaluation and comparison of
  - ...sensor selection algorithms (on EEG data) [2]
  - ...dimensionality reduction algorithms (ICA, PCA, xDAWN, PiSF, CSP) [3, 4, 5, 13]
  - ...classifiers (BRMM, online classifiers, ...) [6, 11, 14, 15]
- Brain-Computer Interfaces (movement prediction, interaction error detection, detection of warning perception) [1, 7, 8, 9, 10, 16, 17]
- soil detection
- parallelization of robot simulations
- classify iterative closest point (ICP) matches for good and bad *localization*

Short Facts

- medium sized framework ($>$ 40000 lines of code)
- developed and tested on Mac OS X and Linux
- 5 years old (open source since August 2013)
- core developer team of 3-5 people and approx. 10 in total
- open source software (GPL, `https://github.com/pyspace`)
- extensive documentation: `http://pyspace.github.io/pyspace/`
- paper about pySPACE published yesterday:

  Mario Michael Krell, Sirko Straube, Anett Seeland, Hendrik Wöhrle, Johannes Teiwes, Jan Hendrik Metzen, Elsa Andrea Kirchner, and Frank Kirchner. pySPACE - A Signal Processing and Classification Environment in Python. *Frontiers in Neuroinformatics*, 7(40), 2013

Short Facts

- medium sized framework ($>$ 40000 lines of code)
- developed and tested on Mac OS X and Linux
- 5 years old (open source since August 2013)
- core developer team of 3-5 people and approx. 10 in total
- open source software (GPL, `https://github.com/pyspace`)
- extensive documentation: `http://pyspace.github.io/pyspace/`
- paper about pySPACE published yesterday:

  Mario Michael Krell, Sirko Straube, Anett Seeland, Hendrik Wöhrle, Johannes Teiwes, Jan Hendrik Metzen, Elsa Andrea Kirchner, and Frank Kirchner. pySPACE - A Signal Processing and Classification Environment in Python. *Frontiers in Neuroinformatics*, 7(40), 2013

How to install and use pySPACE

1. installation (very simple, see tutorial)
2. prepare your data for pySPACE
3. decide and define the processing file
4. potentially modify your config file
5. start software

prepare your data: Input Formats

feature vector:	csv, arff
time series segments:	csv
time series stream:	csv, EDF2
	.set (EEGLAB), .eeg (BrainProducts GmbH)

1. installation
2. **prepare your data**
   dataset description of banana dataset (metadata.yaml)

```
storage_format: [csvUnnamed, real]
type: FEATURE_VECTOR
file_name: banana_data.csv
label_column: 1
...
```

3. decide and define the processing file
4. potentially modify your config file
5. start software

1. installation
2. prepare your data
3. **decide and define the processing file** (examples/bench.yaml)

```
type: node_chain
input_path: "example_summary"
runs : 3
node_chain:
  -   node: FeatureVectorSourceNode
  -   node: TrainTestSplitter
      parameters :
          train_ratio: 0.4
  -   node: __Normalization__
  -   node : 2SVM
      parameters :
          complexity : __C__
  -   node: PerformanceSinkNode
parameter_ranges :
    __C__ : [0.01,0.1,1]
    __Normalization__ : [GaussianFeatureNormalization,
                        EuclideanFeatureNormalization]
```

4. potentially modify your config file

1. installation
2. prepare your data for pySPACE
3. decide and define the processing file (bench.yaml)
4. **potentially modify your config file** (config.yaml)

```
storage: ~/pySPACEcenter/storage
spec_dir: ~/pySPACEcenter/specs
console_log_level : logging.WARNING
file_log_level : logging.INFO
python_path:
        - /home/user/pySPACE/external/libsvm/python/
...
```

5. start software

1. installation
2. prepare your data for pySPACE
3. decide and define the processing file (bench.yaml)
4. potentially modify your config file (config.yaml)
5. **start software**
   go to pySPACEcenter on the command line and type:

```
./launch.py -o examples/bench.yaml --mcore
```

Parallelization

- single-core: — — *serial*
- multi-core: — — *mcore*
- cluster (common storage system needed): — — *loadl*
- possibility to add new modes: — — *cloud*

- online and offline mode
- no interprocess communication (restricted to embarrassingly parallel)
- shared file system required

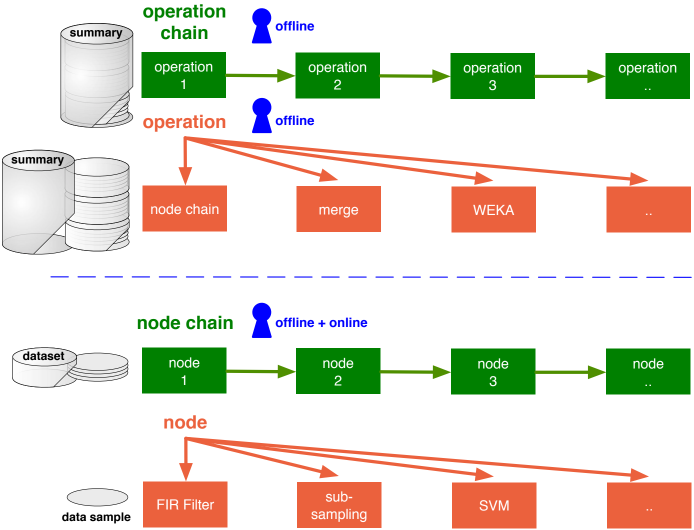
Parallelization

- single-core: – – *serial*
- multi-core: – – *mcore*
- cluster (common storage system needed): – – *loadl*
- possibility to add new modes: – – *cloud*

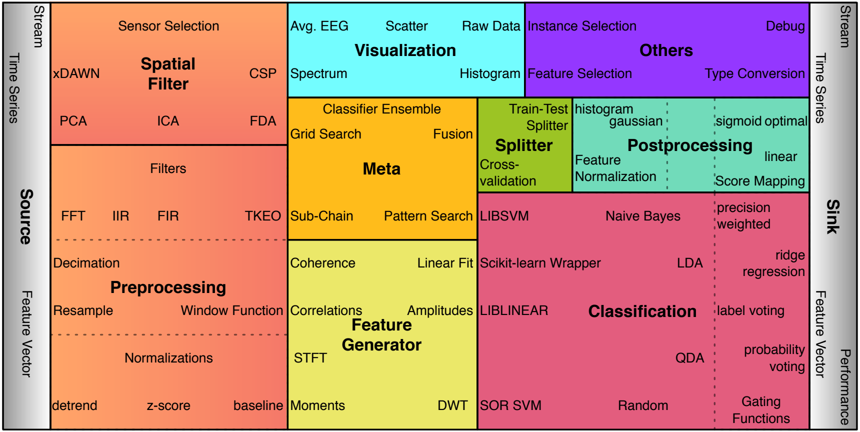
design decisions to enable parallelization:

- online and offline mode
- no interprocess communication (restricted to embarrassingly parallel)
- shared file system required

General Structure Concept



More than 100 algorithms



Stream
Time Series
Source
Feature Vector

Spatial Filter
Sensor Selection
xDAWN
CSP
PCA
ICA
FDA

Preprocessing
Filters
FFT
IIR
FIR
TKEO
Decimation
Resample
Window Function
Normalizations
detrend
z-score
baseline

Visualization
Avg. EEG
Scatter
Raw Data
Spectrum
Histogram

Others
Instance Selection
Debug
Feature Selection
Type Conversion

Meta
Classifier Ensemble
Grid Search
Fusion
Sub-Chain
Pattern Search

Splitter
Train-Test Splitter
Cross-validation

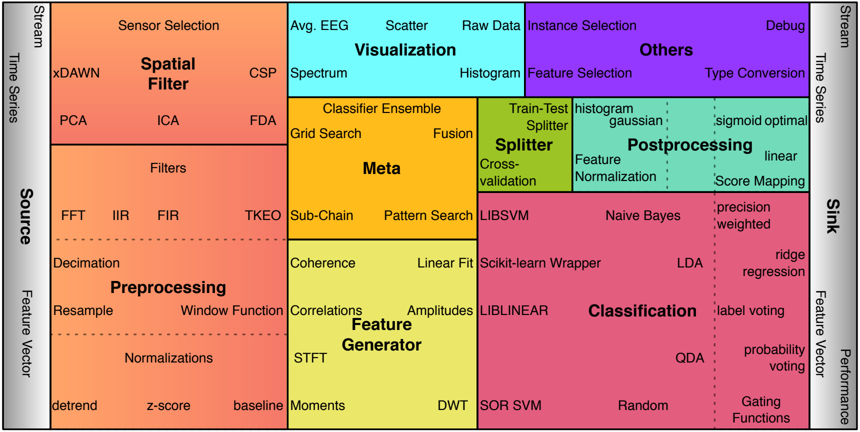
Postprocessing
histogram
gaussian
sigmoid optimal
Feature Normalization
linear
Score Mapping

Feature Generator
Coherence
Linear Fit
Correlations
Amplitudes
STFT
Moments
DWT

Classification
LIBSVM
Naive Bayes
Scikit-learn Wrapper
LDA
LIBLINEAR
QDA
SOR SVM
Random
precision weighted
ridge regression
label voting
probability voting
Gating Functions

Stream
Time Series
Sink
Feature Vector
Performance

More than 100 algorithms



Source: Stream, Time Series, Feature Vector

Spatial Filter: Sensor Selection, xDAWN, CSP, PCA, ICA, FDA

Preprocessing: Filters, FFT, IIR, FIR, TKEO, Decimation, Resample, Window Function, Normalizations, detrend, z-score, baseline

Visualization: Avg. EEG, Scatter, Raw Data, Spectrum, Histogram

Others: Instance Selection, Debug, Feature Selection, Type Conversion

Meta: Classifier Ensemble, Grid Search, Fusion, Sub-Chain, Pattern Search

Splitter: Train-Test Splitter, Cross-validation

Postprocessing: histogram, gaussian, sigmoid optimal, linear, Feature Normalization, Score Mapping

Feature Generator: Coherence, Linear Fit, Correlations, Amplitudes, STFT, Moments, DWT

Classification: LIBSVM, Naive Bayes, Scikit-learn Wrapper, LDA, LIBLINEAR, QDA, SOR SVM, Random, precision weighted, ridge regression, label voting, probability voting, Gating Functions

Sink: Stream, Time Series, Feature Vector, Performance

Modularity concept of node chain based on Modular toolkit for Data Processing (MDP)!

Here new algorithms/libraries can be integrated/interfaced!

Conclusion

- pySPACE automatizes the signal processing and classification workflow.
- automatic parallel execution of other evaluations (WEKA, Reinf. Learning with MMLF `http://mmlf.sourceforge.net/`)
- intuitive configuration without scripting (YAML based)
  ⇒ useable by non-programmers
- possibility to integrate other algorithms/libraries

future steps

- more algorithms and interfaces to other libraries
- more data types (e.g. pictures, videos)
- more applications (e.g. clustering, regression)
- installation suite
- ...

Conclusion

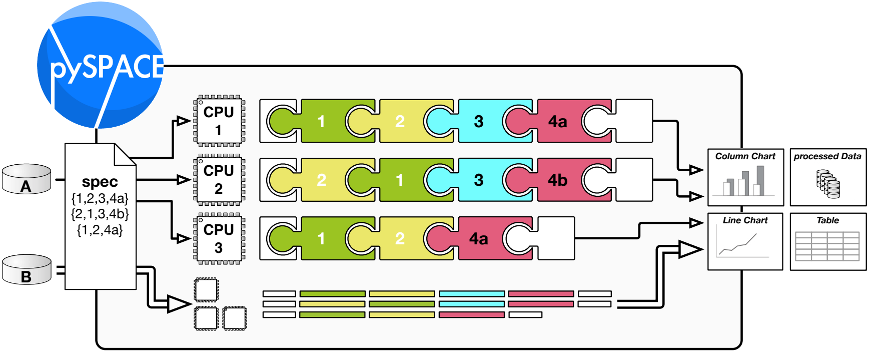
- pySPACE automatizes the signal processing and classification workflow.
- automatic parallel execution of other evaluations (WEKA, Reinf. Learning with MMLF `http://mmlf.sourceforge.net/`)
- intuitive configuration without scripting (YAML based) ⇒ useable by non-programmers
- possibility to integrate other algorithms/libraries

future steps

- more algorithms and interfaces to other libraries
- more data types (e.g. pictures, videos)
- more applications (e.g. clustering, regression)
- installation suite
- ...

Thank you for your attention!

Do you have questions?



funding statement: Supported by the national aeronautics and space research centre of the Federal Republic of Germany (DLR) by grants of the Federal Ministry of Economics and Technology on the basis of a decision by the German Bundestag under the grant numbers 50RA1113 and 50RA1114 in the project VirGo4.

[1] Effects of eye artifact removal methods on single trial P300 detection, a comparative study. *Journal of Neuroscience Methods*, 221(0):41–47, Jan 2014.

[2] David Feess, Mario Michael Krell, and Jan Hendrik Metzen. Comparison of Sensor Selection Mechanisms for an ERP-Based Brain-Computer Interface. *PLoS ONE*, 8(7):e67543, Jul 2013.

[3] Foad Ghaderi. Joint spatial and spectral filter estimation for single trial detection of event related potentials. In *IEEE International Workshop on Machine Learning for Signal Processing, (MLSP)*, 9 2013.

[4] Foad Ghaderi and Elsa Andrea Kirchner. Periodic Spatial Filter for Single Trial Classification of Event Related Brain Activity. In *Proceedings of the 10th IASTED International Conference on Biomedical Engineering (BioMed-2013), February 13-15, Innsbruck, Austria*. ACTA Press, 2 2013.

[5] Foad Ghaderi and Sirko Straube. An adaptive and efficient spatial filter for event-related potentials. In *Proceedings of European Signal Processing Conference, (EUSIPCO)*, 9 2013.

[6] Yohannes Kassahun, Hendrik Wöhrle, Alexander Fabisch, and Marc Tabie. Learning parameters of linear models in compressed parameter space. In Alessandro E. Villa, Włodzisław Duch, Péter Érdi, Francesco Masulli, and Günther Palm, editors, *Artificial Neural Networks and Machine Learning ICANN 2012*, volume 7553 of *Lecture Notes in Computer Science*, pages 108–115. Springer, 2012.

[7] Su Kyoung Kim and Elsa Andrea Kirchner. Classifier transferability in the detection of error related potentials from observation to interaction. In *Proceedings of the IEEE International Conference on Systems, Man, and Cybernetics, SMC-2013, Manchester, UK, October 13-16*, 2013.

[8] Elsa Andrea Kirchner, Su Kyoung Kim, Sirko Straube, Anett Seeland, Hendrik Wöhrle, Mario Michael Krell, Marc Tabie, and Manfred Fahle. On the applicability of brain reading for self-controlled, predictive human-machine interfaces in robotics. *PLoS ONE*, 2013, in press.

[9] Elsa Andrea Kirchner and Marc Tabie. Closing the gap: Combined EEG and EMG analysis for early movement prediction in exoskeleton based rehabilitation. In *Proceedings of the 4th European Conference on Technically Assisted Rehabilitation - TAR 2013*, March 2013.

[10] Elsa Andrea Kirchner, Hendrik Wöhrle, Constantin Bergatt, Su Kyoung Kim, Jan Hendrik Metzen, David Feess, and Frank Kirchner. Towards Operator Monitoring via Brain Reading – An EEG-based Approach for Space Applications. In *Proceedings of the 10th International Symposium on Artificial Intelligence, Robotics and Automation in Space*, pages 448–455, Sapporo, 2010.

[11] Mario Michael Krell, David Feess, and Sirko Straube. Balanced Relative Margin Machine – The Missing Piece Between FDA and SVM Classification. *Pattern Recognition Letters*, Sep 2013, in press.

[12] Mario Michael Krell, Sirko Straube, Anett Seeland, Hendrik Wöhrle, Johannes Teiwes, Jan Hendrik Metzen, Elsa Andrea Kirchner, and Frank Kirchner. pySPACE - A Signal Processing and Classification Environment in Python. *Frontiers in Neuroinformatics*, 7(40), 2013.

[13] Jan Hendrik Metzen, Su Kyoung Kim, Timo Duchrow, Elsa Andrea Kirchner, and Frank Kirchner. On Transferring Spatial Filters in a Brain Reading Scenario. In *Proceedings of the 2011 IEEE Workshop on Statistical Signal Processing*, pages 797–800, June 2011.

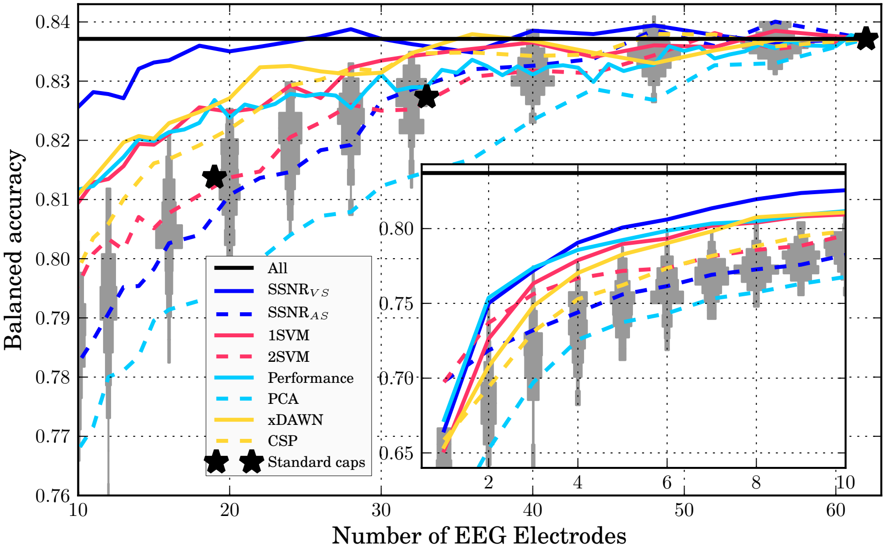
[14] Jan Hendrik Metzen, Su Kyoung Kim, and Elsa Andrea Kirchner. Minimizing Calibration Time for Brain Reading. In Rudolf Mester and Michael Felsberg, editors, *Pattern Recognition*, volume 6835 of *Lecture Notes in Computer Science*, pages 366–375. Springer, August 2011.

[15] Jan Hendrik Metzen and Elsa Andrea Kirchner. Rapid Adaptation of Brain Reading Interfaces based on Threshold Adjustment. In *Proceedings of the 2011 Conference of the German Classification Society, (GfKl-2011)*, page 138, August 2011.

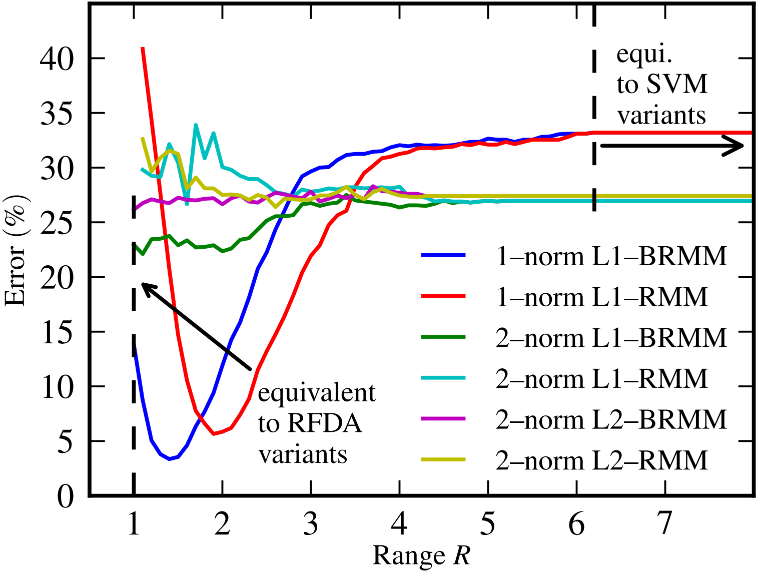
[16] Anett Seeland, Hendrik Wöhrle, Sirko Straube, and Elsa Andrea Kirchner. Online movement prediction in a robotic application scenario. In *6th International IEEE EMBS Conference on Neural Engineering (NER)*, pages 41–44, San Diego, California, Nov 2013.

[17] Hendrik Wöhrle, Johannes Teiwes, Elsa Andrea Kirchner, and Frank Kirchner. A Framework for High Performance Embedded Signal Processing and Classification of Psychophysiological Data. In *APCBEE Procedia. International Conference on Biomedical Engineering and Technology (ICBET-2013), 4th, May 19-20, Kopenhagen, Denmark*. Elsevier, 2013.

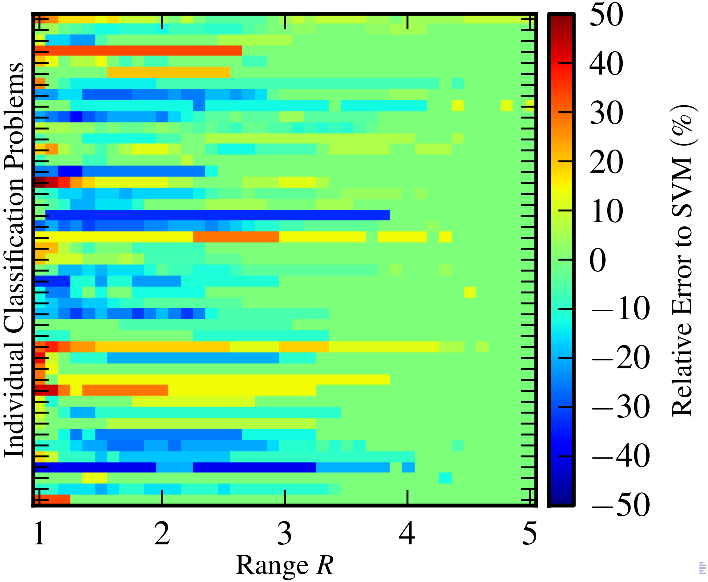
Sensor Selection



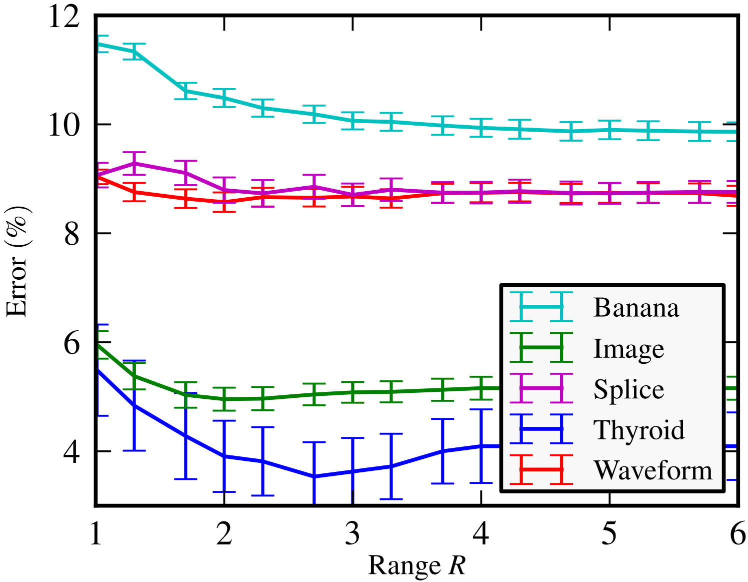
Balanced Relative Margin Machine 1/3



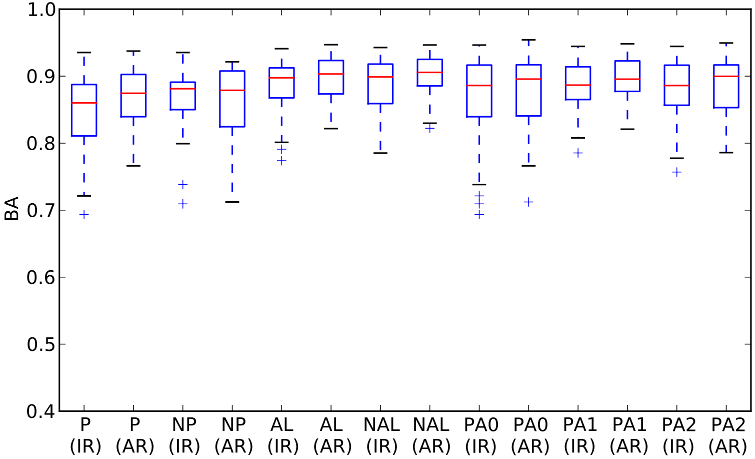
Balanced Relative Margin Machine 2/3



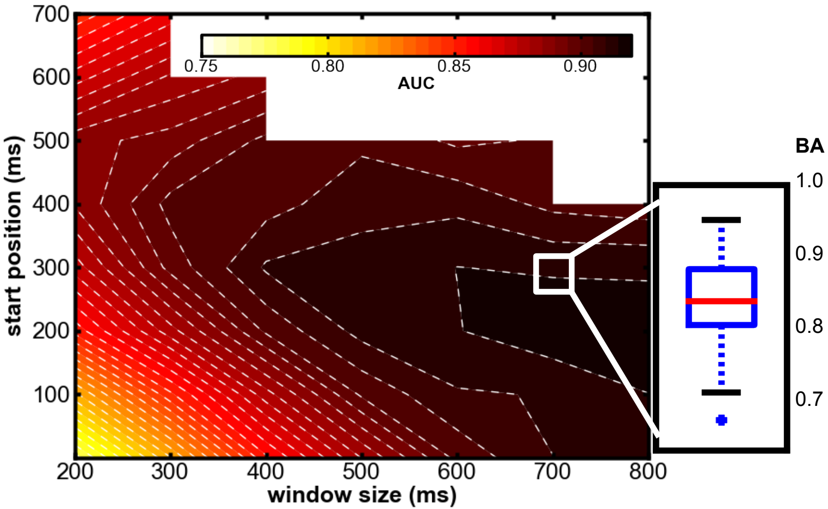
Balanced Relative Margin Machine 3/3



Online classifier evaluation



Segmentation analysis



Installation

required dependencies:

- Python2.7
- YAML
- NumPy
- SciPy

optional dependencies:

- matplotlib (visualizations)
- scikit-learn (classifiers)
- PyQt4 (GUIs)
- LIBSVM, LIBLINEAR, MDP, ... (algorithm interfaces)

download (git clone https://github.com/pyspace/pyspace.git)
python setup.py ⇒ configuration folder in home directory

Installation

required dependencies:

- Python2.7
- YAML
- NumPy
- SciPy

optional dependencies:

- matplotlib (visualizations)
- scikit-learn (classifiers)
- PyQt4 (GUIs)
- LIBSVM, LIBLINEAR, MDP, . . . (algorithm interfaces)

download (git clone https://github.com/pyspace/pyspace.git)

python setup.py ⇒ configuration folder in home directory

Installation

required dependencies:

- Python2.7
- YAML
- NumPy
- SciPy

optional dependencies:

- matplotlib (visualizations)
- scikit-learn (classifiers)
- PyQt4 (GUIs)
- LIBSVM, LIBLINEAR, MDP, . . . (algorithm interfaces)

download (git clone https://github.com/pyspace/pyspace.git)
python setup.py ⇒ configuration folder in home directory

Why use pySPACE instead of other libraries or software?

- no scripting ⇒ usable for neuroscientists
- automatic parallel processing (e.g. on cluster)
- other libraries can be integrated
- a lot of available algorithms
- no separation between preprocessing, classification, parameter optimization, and evaluation
- easy exchange of processing schemes
- real open source
- working on cluster