Pattern Recognition for Neuroimaging Data

Edinburgh, SPM course
April 2019

C. Phillips, GIGA – *In Silico* Medicine, ULiege, Belgium

c.phillips@uliege.be
http://www.giga.uliege.be
Overview

• Introduction
  – Pattern recognition
  – Univariate & multivariate approaches
  – Data representation

• Pattern Recognition
  – Machine learning
  – Validation & inference
  – Weight maps & feature selection
  – Applications: groups & fMRI

• Conclusion & Toolboxes
Overview

• Introduction
  – Pattern recognition
  – Univariate & multivariate approaches
  – Data representation

• Pattern Recognition
  – Machine learning
  – Validation & inference
  – Weight maps & feature selection
  – Applications: groups & fMRI

• Conclusion & Toolboxes
Pattern recognition concept

- Pattern recognition aims to find *patterns/regularities* in the data that can be used to take actions (e.g. make predictions), aka. machine learning, AI,...

- Types of Learning:
  - supervised learning: trained with labeled data (classification & regression)
  - unsupervised learning: trained with unlabeled data (clustering)
  - reinforcement learning: actions and rewards (robotics)

Digit Recognition   Face Recognition   Recommendation Engines
Pattern recognition framework

**Input** (patterns)

0123456789

**Output** (labels)

0 1 2 3 4 5 6 7 8 9

Computer-based procedures that learn a function $f$ from a series of examples

**Learning/Training Phase**

Generate a function or classifier $f$ such that $f(\text{input}) = \text{label}$

**Testing Phase**

Prediction

Test Example

3

Training Examples:

0 1 2 3 4 5 6 7 8 9
Classification model

Class 1
Label = patient

Class 2
Label = controls

New subject

Predictive function: $f$

Training

Prediction: Class membership
Regression model

Training:

Score = 35
Score = 30
Score = 20
Score = 25
Score = 20
Score = 23
Score = 20
Score = 30

New subject

Testing:

Predictive function: $f$

Prediction: Score = 28
Mass-univariate vs Pattern recognition

Standard Statistical Analysis (mass-univariate)

- Voxsel-wise GLM model estimation
- Independent statistical test at each voxel
- Correction for multiple comparisons

Pattern Recognition Analysis (multivariate & predictive)

- Training Phase
  - Volumes from task 1
  - Volumes from task 2
  - New example
- Testing Phase
  - Predictive map (classification or regression weights)
  - Predictions: task 1 or task 2

Very different meaning!
Neuroimaging data

Ex. fMRI time series = 3D array of time series.
= time series of 3D fMRI’s
= 4D image

About the same for a series of structural MRIs
Neuroimaging data features

Data dimensions
- dimensionality of a “data point”, aka. features = #voxels considered
- number of “data points”, aka. samples = #scans/images considered
Neuroimaging data features

Types of features:

- **fMRI:** BOLD signal, contrast image, connectivity maps/matrix, ...
- **sMRI:** GM maps, volume change map, cortical thickness, ...
- PET images
- EEG/MEG
Advantages of pattern recognition

Accounts for the spatial correlation of the data (multivariate aspect)
- images are multivariate by nature.
- can yield greater sensitivity than conventional (univariate) analysis.

Enable classification/prediction of new samples
- ‘Mind-reading’ or decoding applications
- Clinical application

Haynes & Rees, 2006
Overview

• Introduction
  – Pattern recognition
  – Univariate & multivariate approaches
  – Data representation

• Pattern Recognition
  – Machine learning
  – Validation & inference
  – Weight maps & feature selection
  – Applications: groups & fMRI

• Conclusion & Toolboxes
Different classifiers will compute different hyper-planes!
Regression model

Extract Features

Score = 10

Feature 1

Feature 2

Predicted Scores

S1 = 10

S2 = 15

S3 = 22

S4 = 30

4 2

4 2

Extract Features
Linear predictive models

- Linear predictive models (classifier or regression) are parameterized by a weight vector $\mathbf{w}$ and a bias term $b$.

- The general equation for making predictions for a test example $x_*$ is:

$$f(x_*) = \mathbf{w} \times x_* + b$$

- In the linear case $\mathbf{w}$ can be expressed as a linear combination of training examples $x_i$ ($N =$ number of training examples).

$$\mathbf{w} = \sum_{i=1}^{N} a_i x_i$$

Parameters learned/estimated from training data
Weight maps

= predictive patterns!

\[ f(x_*) = w \times x_* + b \]

- Shows the relative contribution of each feature for the decision
- No local inferences can be made!
Neuroimaging data

Problem: $\text{#features} \gg \text{#samples}$

$\rightarrow$ "ill posed problem"

Possible solutions:

- Fewer features
  $\rightarrow$ ROIS, feature selection, searchlight

- Regularization & Kernel Methods
Regularization

- **Regularization** is a technique used in an attempt to **solve ill-posed problems** and to **prevent overfitting** in statistical/machine learning models.

- Regularized methods find $w$ minimizing an objective function consisting of a data fit term $E$ and a penalty/regularization term $J$

$$\min_{w \in \mathbb{R}^p} \{ E(w) + \lambda J(w) \}$$

Data fit term = **loss function** $L$

- Many machine learning algorithms are particular choices of $L$ and $J$ (e.g. Kernel Ridge Regression (KRR), Support Vector Machine (SVM)).

Regularization parameter

The **regularisation term** $J$
The role of regularization

- Weight maps for classifying fMRI images during visualization of pleasant vs. unpleasant pictures.

- All models used a square loss + a different type of regularization.

Baldassarre et al. (2017)
Kernel approaches

Mathematical trick!

- powerful and unified framework (e.g. classification & regression)

Consist of two parts:

- Use of a kernel function
  - kernel matrix (mapping into the feature space)

- Learning algorithm operating with kernel

Advantages:

- Computational shortcut ➔ computational efficiency

- Kernel trick (linear & non-linear) + regularization ➔ efficient solution of ill-conditioned problems.
Kernel matrix

= “similarity measure” between any pair of sample \( \mathbf{x} \) and \( \mathbf{x}^* \)

The “kernel function”

- simple similarity measure
  = a dot product \( \rightarrow \) linear kernel

- more general measures
  = Gaussian, polynomial,... \( \rightarrow \) non-linear kernel

Linear kernel

= dot product

= \((4 \times -2) + (1 \times 3)\)

= -5
Linear classifier prediction

General equation: making predictions for a test example $\mathbf{x}_*$ with kernel methods

Primal representation:
$$f(\mathbf{x}_*) = \mathbf{w} \times \mathbf{x}_* + b$$

Dual representation:
$$f(\mathbf{x}_*) = \sum_{i=1}^{N} a_i \mathbf{x}_i \times \mathbf{x}_* + b$$

Signed distance to boundary (classification)
Predicted score (regression)

Example of kernel methods: Support Vector Machines (SVM), Kernel Ridge Regression (KRR), Gaussian Process (GP), Kernel Fisher Discriminant, Relevance Vector Regression
Multi-kernel learning

- Multiple Kernel Learning (MKL) can be applied to combine different sources of information (e.g. multimodal imaging or ROIs) for prediction.

- In MKL, the kernel $K$ can be considered as a linear combination of $M$ “basis kernels”.

$$K(x, x') = \sum_{m=1}^{M} d_m K_m(x, x')$$

with $d_m \geq 0$, $d_m = 1$

- MKL models simultaneously learn the kernel weights ($d_m$) and the associated decision function ($w$, $b$) in supervised learning settings.
Support Vector Machine

SVM = “maximum margin” classifier

Data: \( \langle x_i, y_i \rangle, i=1,..,N \)
Observations: \( x_i \in \mathbb{R}^d \)
Labels: \( y_i \in \{ -1, +1 \} \)

Support vectors have \( \alpha_i \neq 0 \)
SVM vs. GP

SVM

→ Hard binary classification
  – simple & efficient, quick calculation but
  – NO ‘grading’ in output \{-1, 1\}

Gaussian Processes

→ probabilistic model
  – more complicated, slower calculation but
  – returns a probability \([0, 1]\)
  – can be multiclass

Other machines out there:
  ex. tree-based, deep learning,
Overview

• Introduction
  – Pattern recognition
  – Univariate & multivariate approaches
  – Data representation

• Pattern Recognition
  – Machine learning
  – Validation & inference
  – Applications: groups & fMRI

• Conclusion & Toolboxes
Validation principle

Data set: Samples = \{features, labels\}

\[
\begin{array}{c|c|c|c|c|c}
\text{Training set} & \text{features:} & \text{var 1} & \text{var 2} & \text{var 3} & \ldots & \text{var } m \\
\hline
1 & 1 & \text{gray} & \text{black} & \text{white} & \ldots & \text{gray} \\
2 & -1 & \text{gray} & \text{black} & \text{white} & \ldots & \text{white} \\
3 & -1 & \text{gray} & \text{white} & \text{black} & \ldots & \text{gray} \\
\vdots & \vdots & \vdots & \vdots & \vdots & \ldots & \vdots \\
i & 1 & \text{white} & \text{black} & \text{white} & \ldots & \text{gray} \\
\end{array}
\]

\[
\begin{array}{c|c|c|c|c}
\text{Test set} & \text{True label} & \text{Predicted label} \\
\hline
i+1 & 1 & 1 \\
i+2 & 1 & -1 \\
\vdots & \vdots & \vdots \\
i & 1 & -1 \\
\end{array}
\]

Accuracy evaluation
M-fold cross-validation

• Split data in 2 sets: “train” & “test”
  ➔ evaluation on 1 “fold”

• Rotate partition and repeat
  ➔ evaluations on M “folds”

• Applies to scans/events/blocks/subjects/…
  ➔ Leave-some-out (LSO) approach

• Accumulates metric over the M “folds”.

Diagram: Data partitioning for M-fold cross-validation.
Confusion matrix & accuracy

Confusion matrix = summary table

Accuracy estimation

- Class 0 accuracy, $p_0 = A/(A+B)$
- Class 1 accuracy, $p_1 = D/(C+D)$
- Total accuracy, $p = (A+D)/(A+B+C+D)$

Other criteria

- Sensitivity = $D/(C+D)$
- Specificity = $A/(A+B)$
- Positive Predictive Value, PPV = $D/(B+D)$
- Negative Predictive Value, NPV = $A/(A+C)$
Accuracy & Dataset balance

Watch out if #samples/class are different!!!

Example: Classes A/B with 80/20 samples each

→ observed $a_{\text{tot}} = 70\%$ overall accuracy but

• within class A ($N_A = 80$), excellent accuracy (85%)
• within class B ($N_B = 20$), poor accuracy (10%)

→ balanced accuracy $a_{\text{bal}} = 47.5\%$!

Good practice:

Report

• class accuracies [$a_0$, $a_1$, ..., $a_C$]
• balanced accuracy $a_{\text{bal}} = (a_0 + a_1 + \ldots + a_C)/\#\text{Classes}$
Regression validation

“Mean squared error” (MSE):
- Squared error in one fold: 
  \[ SE_n = (y_n - f(x_n))^2 \]
- Across all CV folds: 
  \[ R(f, X) = MSE = \frac{1}{N} \sum_{n=1}^{N} (y_n - f(x_n))^2 \]
- Out-of-sample “mean squared error” (MSE)

Other measure:
Correlation between:
- predictions (across folds!), and
- ‘true’ targets
Inference by permutation testing

- **\( H_0 \):** “class labels are non-informative”
- **Test statistic = CV accuracy** (total or balanced)
- **Estimate distribution of test statistic under \( H_0 \)**:
  - Random permutation of labels
  - Estimate accuracy
  - Repeat M times
- **Calculate p-value** as
  \[
p = \frac{1}{M} \sum_{m=1}^{M} (a_m^{\text{perm}} \geq a^{\text{true}})
\]
Overview

• **Introduction**
  – Pattern recognition
  – Univariate & multivariate approaches
  – Data representation

• **Pattern Recognition**
  – Machine learning
  – Validation & inference
  – Weight maps & feature selection
  – Applications: groups & fMRI

• **Conclusion & Toolboxes**
Weight vector interpretation

Weight vector

- weight (or discrimination) image!
- how important each voxel is
- for which class “it votes” (mean centred data & $b=0$)

$$W = [0.89 \ 0.45]$$

$$b = -2.8$$
Weight maps for different masks

Linear machine $\rightarrow$ Weight map

Different mask/ROI $\rightarrow$ different feature set $\rightarrow$ different weight map
Feature selection

- 1 sample image ➞ 1 predicted value
- use ALL the voxels ➞ NO thresholding of weight allowed!

Feature selection:
- *a priori* mask or ‘filtering’
- Multiple Kernel Learning
- Sparse methods
- (Search Light)
- Recursive Feature Elimination/Addition

MUST be independent from test data!
Overview

• Introduction
  – Pattern recognition
  – Univariate & multivariate approaches
  – Data representation

• Pattern Recognition
  – Machine learning
  – Validation & inference
  – Weight maps & feature selection
  – Applications: groups & fMRI

• Conclusion & Toolboxes
Application & designs

Levels of “inference”

• within subject ≈ FFX with SPM
  ➔ ‘decode’ subject’s brain states
  ➔ multiple images, e.g. fMRI time series

• between subjects ≈ RFX with SPM
  ➔ ‘classify’ groups, e.g. patients vs. controls
    or regress subjects’ parameter
  ➔ 1 (or few) image(s)/subject
Within subject, fMRI

Activation design → decode stimuli

- Block or event-related design?
- How to account for haemodynamic function?
Within subject, fMRI

Rely on raw BOLD signal per event/block → one label per image!

- 1 volume = 1 sample

Data Matrix =

![Diagram showing data matrix with labels C1, C1, C1, BL, BL, BL, C2, C2, C2, BL, BL, BL]
Within subject, fMRI

Rely on raw BOLD signal per event/block → one label per image!

- 1 volume = 1 sample, or
- average over N volumes

Data Matrix =

```
    C1  C1  C1  BL  BL  BL
    C2  C2  C2  BL  BL  BL
```

Single volumes
Within subject, fMRI

Rely on contrast image per event/block

• 1 contrast = 1 sample
• implicit averaging

“Least Squares All” (LSA)
“Least Squares Unitary” (LSU)
“Least Squares Separate” (LSS)

Abdulrahman & Henson, Neuroimage, 2016
Between subjects

Design
• 2 groups: group A vs. group B
• 1 group: 2 conditions per subject (e.g. before/after treatment)
• 1 group: 1 target score

⇒ Extract 1 (or a few) summary image(s) per subject, and classify/regress

Example:
• contrast (a-fMRI), ICA/correlation map (rs-fMRI)
• GM/Jacobian maps (sMRI)
• FA/MD maps (DWI)
• PET
• etc.
Overview

• Introduction
  – Pattern recognition
  – Univariate & multivariate approaches
  – Data representation

• Pattern Recognition
  – Machine learning
  – Validation & inference
  – Weight maps & feature selection
  – Applications: groups & fMRI

• Conclusion & Toolboxes
<table>
<thead>
<tr>
<th><strong>Univariate</strong></th>
<th><strong>Multivariate</strong></th>
</tr>
</thead>
<tbody>
<tr>
<td>• 1 voxel</td>
<td>• 1 volume</td>
</tr>
<tr>
<td>• target → data</td>
<td>• data → target</td>
</tr>
<tr>
<td>• look for difference or correlation</td>
<td>• look for similarity or score</td>
</tr>
</tbody>
</table>
| • General Linear Model | • Specific machine (SVM, GP, ...)
| • GLM inversion | • training & testing cross-validation |
| • calculate contrast of interest | • estimate accuracy of prediction |
Conclusions

Key points:

• NO local (voxel/blob) inference ➞ CANNOT report coordinates nor thresholded weight map

• Require cross-validation (split in train/test sets) ➞ report accuracy or MSE

• MUST assess significance of accuracy ➞ permutation approach

• Could expect more sensitivity (~like omnibus test with SPM)

• Different questions & Different designs!?
Existing toolboxes

In Matlab

• The Decoding Toolbox, https://sites.google.com/site/tdtdecodingtoolbox/
• Pattern Component Modelling Toolbox (PCMtoolbox), https://github.com/jdiedrichsen/pcm_toolbox
• MVPA by cross-validated MANOVA, https://github.com/allefeld/cvmanova

In Python

• pyMVPA, http://www.pymvpa.org/
• Nilearn, http://nilearn.github.io/
• Brain Imaging Analysis Kit (BrainAIK), https://brainiak.org/
PRoNTo

Pattern Recognition for Neuroimaging Toolbox
http://www.mlnl.cs.ucl.ac.uk/pronto/
with references, manual, demo data, course, etc.

Afternoon workshop

More about
• Weight interpretation
• Machines & “multi-kernel learning”
• Nested CV & parameter optimization
• Feature extraction
• ...

And practical demo of PRoNTo:
• fMRI & group analysis
• GUI and batching
Thank you for your attention!

Any question?

Thanks to the PRoNTo Team for the borrowed slides. 😊


• Schrouff J et al. (2013) PRoNTo: Pattern Recognition for Neuroimaging Toolbox, Neuroinformatics.
