PAIN® 155 (2014) 864–867

www.elsevier.com/locate/pain

Topical review

Decoding the matrix: Benefits and limitations of applying machine learning algorithms to pain neuroimaging

Maria Joao Rosa[a,b,c,*], Ben Seymour[b,d]

[a] Centre for Computational Statistics and Machine Learning, Computer Science Department, University College London, London, UK
[b] Center for Information and Neural Networks, National Institute of Information and Communications Technology, Osaka, Japan
[c] Department of Neuroimaging, Centre for Neuroimaging Sciences, Institute of Psychiatry, King's College London, London, UK
[d] Computational and Biological Learning Lab, Department of Engineering, Cambridge University, Cambridge, UK

Sponsorships or competing interests that may be relevant to content are disclosed at the end of this article.

1. Introduction

A unique and puzzling observation about pain is its distributed processing amongst multiple cortical and subcortical brain areas, informally called the pain matrix. This is challenging and frustrating—challenging because it puts the onus on neuroscientists to determine how each area and its interactions contribute to pain processing, but frustrating given that no individual area can act as an objective biomarker for subjective pain. Recently sophisticated new analytical methods that can decode complex patterns in data offer potentially more promising ways to interrogate pain related brain activity. Here we consider whether and how these methods might contribute to basic and clinical pain neuroscience, and whether their limitations outweigh their promise.

2. Principles of decoding

In principle, any physiological measure that correlates with pain, such as heart rate, can be used for prediction (decoding). The problem is that most measures lack sensitivity and specificity, such that we might need to pool the predictive capabilities of multiple data. Neuroimaging techniques, such as functional magnetic resonance imaging (fMRI) and magneto-/electroencephalography (M/EEG) allow noninvasive recording of brain signals from a large number of spatially distributed sensors (eg, voxels/electrodes) and time points. Traditionally, these data have been analysed using simple linear regression models, treating the signals from each sensor independently [8]. Accordingly, the approach has been to predict signal changes, averaged over trials/subjects, at the level of individual sensors from a set of known variables (eg, experimental conditions). Such analyses are called univariate: the analysis of one sensor does not affect the analysis of any other (Fig. 1A). Although univariate analyses have proven powerful to test which brain regions respond to different aspects of pain processing, they do not capture the complex spatiotemporal dynamic between regions that characterises brain function.

By contrast, multivariate statistical analyses do not treat each sensor independently and are in principle more suitable than univariate methods for assessing information encoded in spatial and temporal dependencies among regions. For this reason, the use of multivariate techniques for neuroimaging, in particular multivariate pattern analyses (MVPA), also known as brain decoding [13,25], has been growing over recent years [19]. Brain decoding is usually based on machine learning (ML) techniques [3]. ML is a branch of artificial intelligence concerned with learning from data and focussed on making predictions. ML has an extremely broad application domain, from computer vision to stock market analysis. In neuroimaging, ML tries to identify spatial and/or temporal patterns in the data from multiple sensors that best discriminate between 2 or more conditions (classification) or predict continuous variables (regression) on unobserved data. Brain decoding therefore seems particularly well suited for clinical applications, having the potential to identify diagnostic biomarkers for different medical conditions [14] and for the development of brain–machine interfaces [24].

3. Decoding methodology

ML-based decoding analysis of brain imaging data entails a series of steps that include feature extraction and selection, training and testing, and pattern interpretation (Fig. 1B).

Feature extraction consists of selecting the data (features) that will be used to learn the ML model. In neuroimaging, it is common to treat the sensors' data as features and use whole-brain signals as input to the model [21]. An alternative is to first run a sensorwise

[*] Corresponding author at: Department of Neuroimaging, Centre for Neuroimaging Sciences, Institute of Psychiatry, King's College London, London, UK. Tel.: +44 (0) 20 3228 3066.

E-mail address: maria.rosa@kcl.ac.uk (M.J. Rosa).

http://dx.doi.org/10.1016/j.pain.2014.02.013

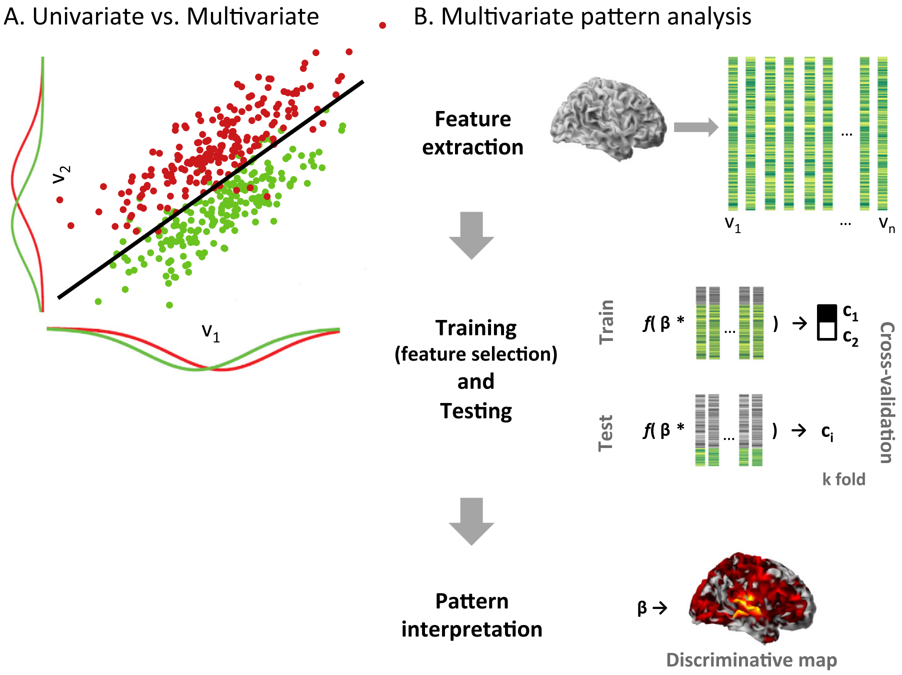
A. Univariate vs. Multivariate
v_2
v_1
B. Multivariate pattern analysis
Feature extraction
v_1 ... v_n
Training (feature selection) and Testing
Train
$f(\beta *$ ... $) \rightarrow$ c_1 c_2
Test
$f(\beta *$ ... $) \rightarrow$ c_i
Cross-validation
k fold
Pattern interpretation
β →
Discriminative map

Fig. 1. (A) Toy example of the response of 2 sensors (v_1 and v_2) for 2 different conditions (green and red). If each sensor is analysed independently (univariate analysis), it is not possible in this case to reliably distinguish between the 2 conditions. Distributions of the responses of each sensor for the 2 conditions (green and red curves) overlap significantly (particularly for v_1). However, using a linear classifier (multivariate pattern analysis or brain decoding), it is possible to estimate a linear function (black line) that can reliably discriminate between the 2 conditions using data from the 2 sensors simultaneously. (B) Multivariate pattern analysis (brain decoding) steps. These steps include feature extraction and possibly selection (extracting and selecting a subset of brain signals from sensors v_1 to v_n) as well as training and testing a machine learning (ML) model using cross-validation (CV). It is important to note here that when there is no a priori reason for choosing a particular subset of features, feature selection should be performed using training data only (ie, within CV) to avoid double dipping (ie, the use of both training and testing data to select relevant features), which can strongly bias estimates of the generalization error [16]; for a comprehensive discussion on circularity in feature selection, see [16]. The k-fold CV breaks the data into k blocks of observations. The model is then trained using $k - 1$ blocks and tested on the remaining block. This procedure is repeated k times. Common choices for k are $k = 5$ and 10. When k equals the total number of samples, it is called leave-one-out CV (LOOCV). LOOCV is approximately unbiased for the true (expected) prediction error because the data are trained on all observations minus 1. However, because the training sets are almost identical, the predictions are highly correlated, and therefore the LOOCV prediction error estimate tends to have higher variance than does the error estimate from k-fold CV. CV can also be used for choosing the value of tuning parameters from the ML algorithm or feature selection approach, as long as we hold out a validation set (independent from the training and testing data used for CV) to assess the model generalization error. However, in situations where there are not enough data, model selection and parameter tuning can be done in an inner CV performed on every training set of the main CV (known as nested CV). This approach can, however, be computationally expensive. We illustrate here a linear classifier (classification model) where the predictions of 2 classes, c_1 (eg, chronic pain patients) and c_2 (eg, healthy control), are given by a linear combination of the input features, ie, $f(\beta \times \text{features}) \rightarrow c_i$, where β are the model parameters, and grey regions indicate the blocks of data not being used in this cross-validation fold. The final step refers to interpreting the predictive patterns by displaying them, for example, in the form of discriminative maps. Note that the model predictions are based on all features.

univariate analysis and then use the estimated regression coefficients as features [9]. More recently, connectivity-based features, such as correlation between time series of different regions, have also been used [2,1].

Feature selection is an optional step where nonmeaningful/noisy features can be precluded from the analysis. Feature selection can rely on dimensionality reduction methods [22] or on prior knowledge (eg, regions of interest) [12]. It can also rely on measures such as the predictive accuracy/error of the model [10] or be part of the model itself [30].

After extracting features, the next step is to train and test the model. By training, we mean learning the parameters of the model using data. By testing, we mean estimating the model's extrasample/prediction error, ie, the average generalization error when the algorithm is applied to an independent test sample from the same distribution of the training data (eg, a different sample of subjects or experimental conditions). A common approach to do this is to use cross-validation (Fig. 1B). Cross-validation uses part of the available data to train the model and part to test it. It is important to emphasize that if the test set data are not independent from the training data, the cross-validation error does not reflect the true generalization error of the model. The choice of algorithm depends on the problem, and different properties—such as its predictive accuracy, interpretability, and reproducibility—may be more or less important depending on the application domain [20].

Finally, in neuroscience, it is important to assess the role that each feature played in the predictions (pattern interpretation). For linear algorithms, it is easy to represent the parameters of the model in the original data space and to visualize the importance of each feature (magnitude of its parameter) in what is called a discriminative map (Fig. 1B).

4. Decoding pain

The first study to use brain decoding in the context of pain [18] demonstrated the feasibility of using ML on whole-brain fMRI data from healthy individuals to predict self-reported thermal pain. Using similar approaches to decode different aspects of pain perception, Brodersen et al. [4] demonstrated that the joint activation of a set of brain regions, including the pain matrix, provided more accurate trial-by-trial predictions of thermal pain perception than any single region. The spatial patterns of activity in insular and cingulate cortex have been found to highly overlap for pain felt on one's hand and observed on another person's hand [7]. Cecchi et al. [6] illustrated the advantage of combining ML and psychophysics to predict the temporal evolution of thermal pain perception from within-subject fMRI data. Despite these successes, the translation of brain decoding into real-world clinical applications depends on its ability to make between-subject predictions. One of the first studies to address this issue [5] showed that ML