# Probing machine-learning classifiers using noise, bubbles, and reverse correlation

Etienne Thoret[*1,4], Thomas Andrillon[3], Damien Léger[2], Daniel Pressnitzer[1]

[1] Laboratoire des systèmes perceptifs, Département d'études cognitives, École normale supérieure, PSL University, CNRS, 75005 Paris, France.

[2] Université de Paris, APHP, Hotel Dieu, Centre du Sommeil et de la Vigilance & EA 7330 VIFASOM, Paris 75006, France.

[3] Turner Institute for Brain & Mental Health and School of Psychological Sciences, Monash University, Melbourne 3168, Australia.

[4] Aix Marseille Univ, CNRS, PRISM, LIS, ILCB, Marseille, France.

[*]corresponding author: etiennethoret@gmail.com

**Abstract**

**Background:** Many scientific fields now use machine-learning tools to assist with complex classification tasks. In neuroscience, automatic classifiers may be useful to diagnose medical images, monitor electrophysiological signals, or decode perceptual and cognitive states from neural signals. However, such tools often remain black-boxes: they lack interpretability. A lack of interpretability has obvious ethical implications for clinical applications, but it also limits the usefulness of these tools to formulate new theoretical hypotheses.

**New method:** We propose a simple and versatile method to help characterize the information used by a classifier to perform its task. Specifically, noisy versions of training samples or, when the training set is unavailable, custom-generated noisy samples, are fed to the classifier. Multiplicative Gaussian noise, so-called "bubbles", or additive uniform noise are applied to the input representation. Reverse correlation techniques are then adapted to extract either the potent information, defined as the parts of the input dataset that have the most weight in the classification decision, and represented information, which correspond to the input features most representative of each category.

**Results:** The method is illustrated for the classification of written numbers by a convolutional deep neural network; for the classification of speech versus music by a support vector machine; and for the classification of sleep stages from neurophysiological recordings by a random forest classifier. In all cases, the features extracted are readily interpretable.

**Comparison with Existing Methods:** Quantitative comparisons show that the present method can match state-of-the art interpretation methods for convolutional neural networks. Moreover, our method uses an intuitive and well-established framework in neuroscience, reverse correlation. It is also generic: it can be applied to any kind of classifier and any kind of input data.

**Conclusions:** We suggest that the method could provide an intuitive and versatile interface between neuroscientists and machine-learning tools.

**Keywords:** Data analysis – Interpretability – Deep neural networks – Automatic classifiers – Reverse correlation – Auditory models – Sleep stages classification

**Highlights:**

- The heuristics of black-box classifiers can be probed with noisy inputs
- The relevant features can be visualised in the input representation space
- The method applies to any kind of data such as 2D images or 1D time series
- It applies to any classifier such as deep neural networks, support vector machines, random forests

## Introduction

Applications of machine-learning techniques permeate more and more scientific fields, with rapid and sometimes unexpected success (LeCun et al., 2015; Jordan & Mitschell, 2015; Kriegeskorte & Douglas, 2018; Richards et al., 2019). At the same time, it is becoming a widely-acknowledged issue that many of these tools are often used as black boxes, even though they would benefit from being interpreted (Molnar, 2020; Doshi-Velez & Kim, 2017). For instance, if a Deep Neural Network (DNN) was used to make life-changing decisions such as deciding on an intervention based on medical imagery, both the clinicians and the patients would have a clear desire to know the rationale that motivated the decision. Also, the power of classifiers to detect useful information in large datasets holds many promises to improve theoretical models. but then, understanding at least to some extent the classifier's operation is crucial (Zihni et al., 2020).

Understanding what a complex classifier does after being trained on possibly millions or billions of samples is usually hard. It is hard for a reason: if the task that the classifier solves had a known explicit solution, then there probably would not have been any incentive to develop the classifier in the first place. In addition, modern techniques involve artificial network architectures with interconnected layers, each including highly non-linear operations (Sejnowski, Kienker, & Hinton, 1986). A lot of the computational power of such algorithms lies in cascades of feed-forward and feed-back non-linear operations. Unfortunately, human reasoning seems most at ease to generate intuitions with linear processes, and not for complex combinations of non-linear ones.

As a consequence, designing methods to interpret machine learning tools is a fast-growing field of research of its own right, often designated under the term *Explainable AI* (Guidotti et al., 2018). It has dedicated journals within the machine learning community (e.g. *Distill*) and an associated DARPA challenge (*XAI*). Recent reviews covering the types of methods exist (Molnar, 2020), also covering more specifically the feature visualization approach taken here (Olah et al., 2017). Within this context, our aim is not to outperform the state-of-the art specialized interpretability methods, but rather to provide a general tool that will hopefully be intuitive to neuroscientists, as it is based on familiar methods for this community. The manuscript describes the method, provides an open

software library to use it, and shows examples of application, demonstrating how it can achieve useful results.

The gist of the method is to try and reveal the input features used by an automatic classifier to achieve its task *without any knowledge* about the classifier. As such, it is what is termed an "agnostic" method of explanation: it does not attempt to describe mechanistically the operation of a specific classifier, which it considers as a black box. Even if the classifier's details are available, they are not considered as they may be too complex to understand intuitively. Rather, the aim is to relate features of the input space to the classifier's decisions, that is to identify the features that participate the most to the classifier's performance. Such a problem is closely related to issues that neuroscientists and experimental psychologists have been addressing for years: providing useful insights of, for instance, human perception, without a full knowledge of the highly complex and non-linear underlying processes implemented by the brain.

The method we propose is directly inspired from the reverse correlation techniques developed for studying human audition (Ahumada et al., 1971) and vision (Neri et al., 1999; Gosselin & Shyns, 2001, 2003). Reverse correlation is based on linear systems analysis (Wiener, 1966). It uses stochastic perturbations of a system to observe its output. If the system were linear, an average of the inputs weighed by the observed outputs would be able to fully characterize the system. However, even for the highly non-linear systems studied by neuroscience, reverse correlation has a track-record of successful applications. For neurophysiology, averaging input stimuli according to neural firing rates has been used to describe neural selectivity (Ringach & Shapeley, 2004 for a review). For psychophysics, averaging input stimuli according to participant's decisions has revealed stimulus features on which such decisions are made for detection or discrimination tasks (Ahumada et al., 1971; Gosselin & Shyns, 2001, 2002). In this spirit, it seems appropriate to investigate the potential of reverse correlation to probe automatic classifiers, as its advantages and limitations are already well understood for non-linear systems.

One important benefit of the reverse correlation framework is its complete independence from the underlying classifier's architecture. Unlike efficient but specific methods tuned to a classifier's

architecture (see Guidotti et al., 2018 for a review), the reverse correlation can be used to probe any algorithm that separates the input data into distinct classes. Even for the currently popular agnostic interpretability methods, this is not always the case: Class Activation Mappings (Zhou et al., 2016) are specific to convolutional networks; LIME (Ribeiro et al., 2016) and RISE (Petsiuk et al., 2018) highlight features of specific examples which may or may not be representative of the classification task in general. Also, the method can operate in the same representation space used as an input to the classifier, and can be applied to any type of representation (2D images, 1D time series such as audio, higher-dimensional brain imaging data, for instance).

The outline of the method is as follows. First, a set of stochastic inputs are generated, by introducing noise on the training dataset when available, or, when unavailable, by generating pseudo-random samples designed to cover the input space. The noise takes two forms: multiplicative low-pass noise, known as "bubbles" (Gosselin & Shyns, 2001, 2002), or additive noise, as is more classically the case for reverse correlation. Second, the inputs are sorted according to the classification results. Third, inputs belonging to the same class are grouped together, with some refinements of the standard reverse correlation methods inspired by signal detection theory (Green & Swets, 1966) to weigh the results with the variability observed after classification. The two variants of input noise, multiplicative or additive, aim to probe two kinds of possibly overlapping but not necessarily identical input features: (1) the *potent* features, probed with multiplicative noise, which correspond to the parts of the input dataset that have the most weight in the classification decision and (2) the *represented* features, probed with additive noise, which correspond to the input features most representative of each category. In the machine-learning literature, these would loosely correspond to the "attribution" versus "feature visualization" problems (Ohla et al., 2017). In psychophysics, the idea of "potent information" has been introduced by Gosselin et al. (2002) and can be contrasted with the older notion of "prototypical information" (Rosch, 1983).

## 1. Material and Methods

## 1.1. Probing potent features

Following the psychophysical literature (Gosselin & Shyns, 2002), we term "potent features" the subsets of the input set that are the most positively correlated with the decision taken by the classifier. The aim of this first method is to uncover where such features are located over the whole input dataset. From now on, we assume that the classifier has been trained and is available to the probing method.

### 1.1.1. Procedure

To identify potent features, the input space is pseudo-randomly sampled with multiplicative low-pass filtered noise applied to training samples. A reverse correlation analysis is then applied to identify the noise masks correlated with the classification decision. The algorithm is directly inspired by the "bubbles" method (Gosselin & Shyns, 2001), originally designed to characterize the visual features underlying human behavioral performance for image classification tasks. Having low-pass noise allows to control for the resolution of the probing, enabling a more efficient search with lower resolution. The multiplicative method also focuses the probing to features actually present in the input dataset, further increasing efficiency.

We present two sub-variants of the method, to account for the availability or not of the training set to the end user: 1a) multiplicative lowpass noise is applied to the training set; 1b) multiplicative lowpass noise is applied to broadband noise generated in the input stimulus space. We now describe the algorithm, jointly for 1a) and 1b). The algorithm is illustrated in Figure 1. A software repository written in the Python programming language is provided, including code for all of the examples presented, and available at https://github.com/EtienneTho/proise.

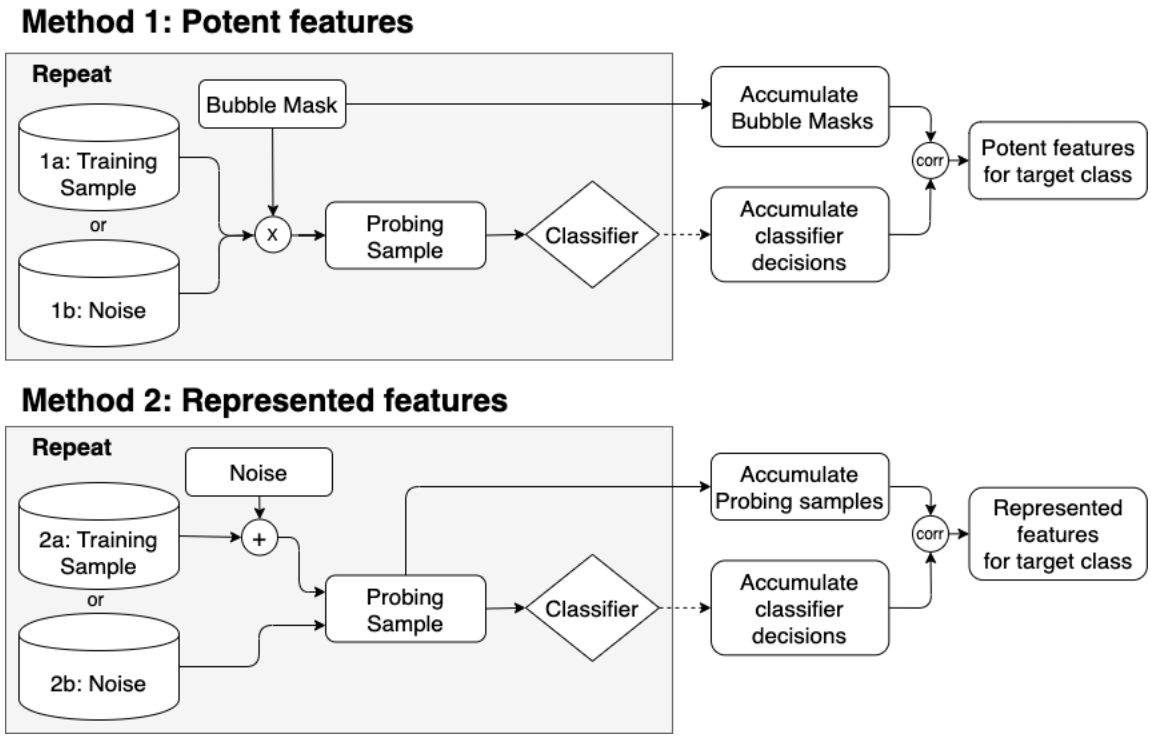



***Fig. 1*** *Summary of the two probing methods. Both methods have two variants depending on the availability of the training set.* ***Method 1, Potent features:*** *Training samples (1a) or noise (1b) are multiplied by bubble masks to obtain the probing samples that are fed to the classifier. The vectors of bubble masks values at each point of the input space are then correlated with the vector of classifier's decisions to obtain the potent features map.* ***Method 2:*** *Training samples combined with additive noise (2a) or simply noise (2b) are used as probing samples fed to the classifier. The vector of probing samples values are then correlated with the vector of classifier's decisions to obtain the represented features map. For methods 1b) and 2b), the noise is arbitrary but can be derived from a limited set of testing samples (see text for details). Also, in all cases, the classifier's decisions can be expressed as the binary class decision or as continuous probabilities of class belonging when available.*

We now provide a textual description of the main algorithm and its variants.

For each pass:

1. A bubble mask is generated. This consists of a mask in the input space, of dimension N, consisting of randomly positioned N-dimensional Gaussian windows called bubbles (see Figure 2). The number of bubbles is defined probabilistically, to avoid spurious correlations

between the presence of a bubble in one part of the representation with the absence of bubbles in other parts of the representation[1]. In practice, the probability that there is a bubble at one point in the input space is set by the probability $P_{\text{bubble}}$ fixed over the whole input space. This probability and the size of the bubbles in terms of the Gaussian standard deviations are hyperparameters of the algorithm, which can be arbitrarily chosen to fine-tune the output resolution. In practice, an input array of dimension N populated by zeroes except for *nbBubbles* unit values is convolved with N-dimensional Gaussian windows. The resulting mask is denoted *BubbleMask*.

2. The probing data is generated. For variant a), the probing data is one exemplar of the training dataset, randomly chosen. For variant b), the probing data is an N-dimensional activation noise (see section 2.1.2 for details and a suggested technique to obtain an appropriate coverage of the input space). The probing data is denoted *ProbingData*.
3. The probing sample is obtained by multiplying the bubble mask with the probing data: *ProbingSample = BubbleMask * ProbingData.*
4. The probing sample is fed to the classifier and the output class is recorded. The probing sample is labeled 1 if it classified in the target class, 0 otherwise. For each probing sample, we therefore obtain a vector *C* of 1s and 0s describing the classifier's decisions. Alternately, when available, the continuous probability of the probing sample to belong to the target category can be recorded in *C*.

Reverse correlation analysis:

For each point *i*, in the input space, the potent map for a target, $D_{i,C}$, is computed as the correlation between the bubble masks values at point *i, BubbleMask*$_i$ and the classifier's decisions *C.* It should be noted that the analysis is performed on the bubble masks, and not on the probing samples themselves. The Pearson or Spearman methods can be used to estimate the correlation coefficient.

[1] We wish to thank an anonymous reviewer for this suggestion.

### 1.1.2. Generation of probing samples activations when the training set is unavailable

As mentioned above, if the training set for the classifier is unavailable to the end user, probing samples can be generated from noise in the input space. The choice of the noise distributions is a free parameter of the method. The simplest choice is to draw samples from a uniform distribution at each point of the input space, with distribution boundaries covering the full range of valid input values. However, this sometimes leads to uneven coverage of the output categories, for instance if the classifier's boundaries are especially complex or if the decision algorithm is highly non-linear. Obviously, categories not covered by probing samples cannot be interpreted.

In this case, we suggest to generate pseudo-random probing samples by first randomly inversing the dimensions of the latent space of a Principal Component Analysis (PCA) trained on a representative set of inputs relative to the classifier's task. These can be obtained for instance from a test set. There are few use cases where no exemplars from test set are available, as the test set basically defines the task of the classifier. After the PCA, noise can be generated in the low dimension space and inverted to obtain noise in the input space. The important point here is that, in a whitened representation such as a PCA, it is possible to systematically sample the latent input space by sampling each PCA dimensions with unit-normalized Gaussian noise. Conversely, it is more difficult to homogeneously sample the original feature space without any a priori knowledge of its latent structure.

In any case, the qualitative goal during the generation of probing samples is to achieve a balanced coverage of all output categories after classification. Iterative choices when generating such probing samples, using a PCA as just described or directly from the raw input space or from any other method, may actually participate in the understanding of the classifier's features.

### 1.1.3. Statistical testing

The potent features maps illustrate, in the input space, the location of the features used by the classifier to assign samples to a given category. Visual inspection may be sufficient to get a

qualitative understanding of the classifier's operation. However, in some cases, it may be desirable to assess statistically the relevance of each feature appearing in the maps.

There are many options to assess the significance of such data, from which we outline here one possible methodological choice. First, the maps can be shuffled by running the algorithm described above many times while randomly assigning output categories to each sample. For each point in the actual map, a *t*-test (or a non-parametric equivalent) is applied to compare the map value with the mean of the shuffled data. Maps are usually high-dimensional so a correction for multiple comparisons is recommended. Again, several standard choices exist, including Bonferroni correction, cluster permutation (Maris & Oostenveld, 2007), or False Discovery Rate (FDR) (Benjamini & Hochberg, 1995). As the application of these statistical procedures is standard, in the illustrative examples discussed below we focus on the description of our novel method and only provide the raw feature maps, without statistics.

## 1.2 Probing represented features

We define "represented features" as the parts of the input space that would be the most representative of a given class. We estimate them by using additive noise on the input, followed by reverse correlation, in a way reminiscent of receptive fields measurement in neurophysiology or decision template estimation in psychophysics.

### 1.2.1 Procedure

To build the represented features map, the input space is randomly perturbed with additive noise. The aim is to probe the whole of the input space, even for features that are not represented in the training dataset, with the highest possible resolution. Then, all probes classified as members of the same category are combined, in a direct adaptation of the classic reverse correlation method. Here, we introduce two differences with classic reverse correlation. First, for generality, we do not distinguish correct classifications from false positives or false negatives. This would require to know the training dataset or to have a large labeled test dataset. Again, we propose two sub-variants of the algorithm depending on the availability of the training dataset: a) broadband noise in the input space

is added to the training set; b) broadband noise is generated in the input space. The algorithm is illustrated in the bottom part of Figure 1.

We now provide a textual description of the algorithm and its variants.

For each pass:

1. The probing sample is generated. For a), the probing sample is one randomly chosen exemplar of the training dataset, with noise added. The goal is to perturb the input to introduce variability, so that only the most salient information (to the classifier) remains in the reverse correlation average. For b), the probing sample is an N-dimensional noise. The probing sample is denoted *ProbingSample*.
2. The probing sample is fed to the classifier and the output class is recorded. The probing sample is labeled 1 if it classified in the target class, 0 otherwise. We therefore obtain a vector $C$ of 0s and 1s recording whether each probing sample has been assigned to the target class. As for potent features, continuous probabilities may be used instead of binary class decisions.

Reverse correlation analysis:

For each point, $i$, in the stimulus space, the represented map for a target $D_{i,C}$, is computed as the correlation between the probing sample masks array values at point $i$, $ProbingSample_i$, and the classification decisions vector $C$.

### 1.2.2 Dimensionality reduction

Depending on the architecture of the classifying pipeline and input space, the estimation of the represented map with reverse correlation can be resource-intensive. A standard technique to improve efficiency when training a classifier is to reduce the number of dimensions of the input space, for instance by using PCA. (e.g., Patil et al., 2012). Here too, the probing and reverse correlation analysis can be performed in a space with reduced dimensionality before inverting back to the original input space.

For the generation of probing noise in variant b), the same method described in section 2.1.2 apply, with the potential use of PCA to shape the noise for a balanced coverage of all output classes.

### 1.2.3 Statistics

The statistical analysis of represented maps can be done with the same generic tools as for potent features maps, described in section 2.1.3.

# 2 Results

To illustrate the methods introduced above, we present three different use cases covering different types of input data (2D images, 1D speech, multi-channel electrophysiological data) and decision type (binary versus multiclass decisions). First, we interpret the classification of handwritten digits, using a visual task (2-D pixel input space) performed with a convolutional deep neural network. Second, we interpret the classification of speech versus music, using an audio task (1-D audio time series converted to a 4-D auditory model) performed by a support vector machine. Third, we interpret a sleep scoring algorithm aiming at classifying electrophysiological data into sleep stages (3 classes) using 75 spectral features extracted from 30-s time epochs fed to a random decision forest classifier. Although voluntarily simple, the first two examples are used to introduce the main features of our method didactically. The last example illustrates an application of our method to a concrete neuroscientific problem, as sleep stages classification from polysomnographic recordings is a time-consuming task that would greatly benefit from reliable and interpretable automation (Andrillon et al., 2020). A benefit of our interpretation method is its generality, so the examples covered here are also intended to showcase the ingredients needed for many use cases relevant to neuroscience.

## 2.1 Digits classification

### 2.1.1 Classification task

In this first example, we classify visual samples of handwritten digits from the *MNIST* database (Deng, 2012). This is a standard database for evaluating image classification algorithms in the machine-learning community. It is composed of handwritten digits, from 0 to 9, with 60000 samples in the training set and 10000 samples in the test set. Each sample is a two-dimensional greyscale image with pixels values between 0 and 1.

Many algorithms can now successfully perform this classification task. Here we trained a Convolutional Neural Network (CNN) to discriminate between digits, with the following architecture: 2D-convolutional layer (3, 3), Max Polling layer (2, 2), 2D-convolution layer (3, 3), flattening layer, dense layer with 10 outputs and a softmax activation. Three training iterations were run. As expected, a high classification accuracy of 97% was obtained on the test set.

### 2.1.2 Potent features maps

We first describe the probing of the trained CNN using Method 1, aimed at uncovering the potent features maps. We compared the cases for which the training set was available or not, and used the PCA-based method introduced in section 1.1.2 to generate probing samples when only the test set was available. Figure 2 illustrates the kind of probing samples obtained in these various cases.

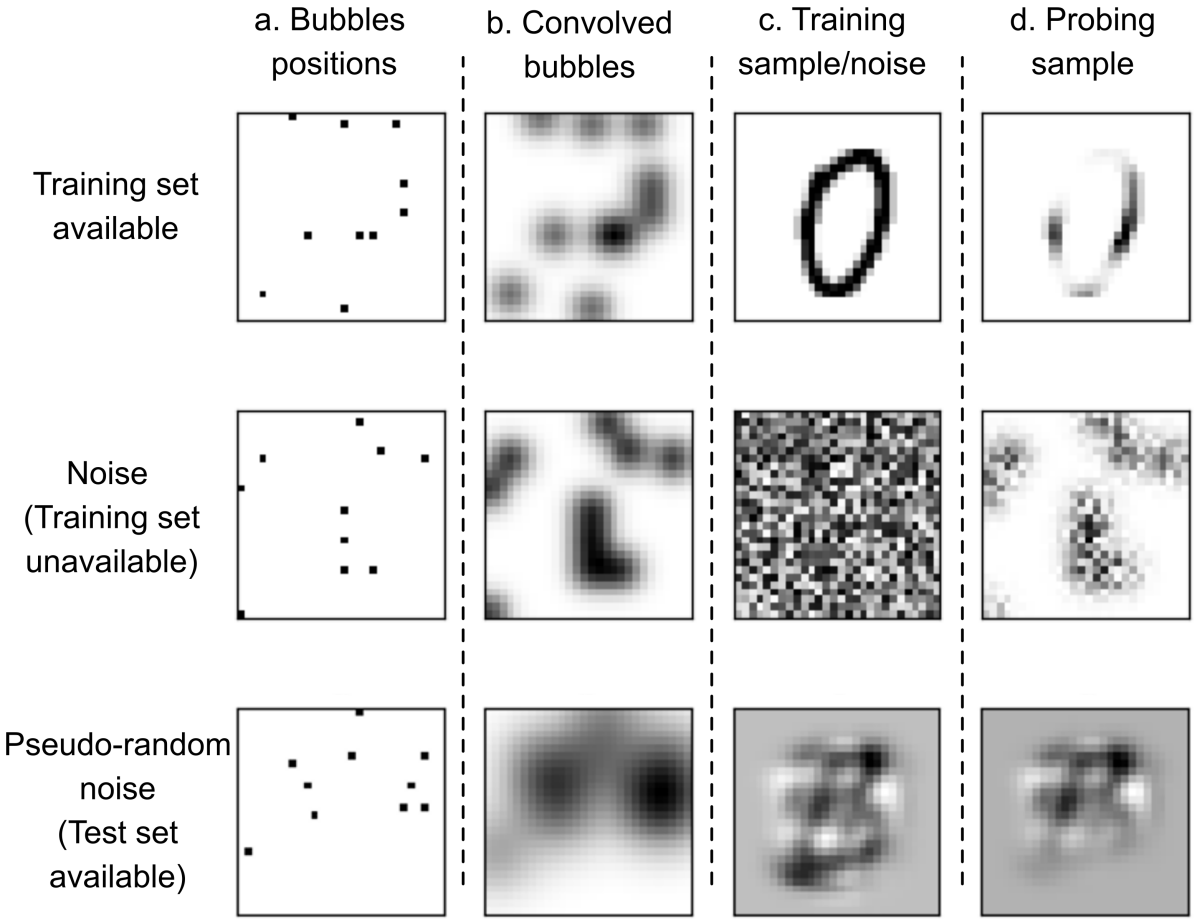



***Fig. 2. Illustration of the probing method with bubbles on 2-D images. Top row: construction of the probing samples for potent features with the training set available. a)*** *Random placement of bubbles* ***b)*** *Convolution with 2-D Gaussian distributions to obtain the bubble mask* ***c)*** *A training sample* ***d)*** *Bubble mask applied to the training sample to obtain the probing sample.* ***Middle row: potent features***

***with training set unavailable, uniform noise.*** *Columns as above, except that the training sample is replaced by uniform noise. The resulting probing samples do not cover the output categories efficiently (see text), likely because of their qualitative differences with digits.* ***Bottom row: potent features with training set unavailable, dimension-reduced noise.*** *Here the training sample is replaced by noise generated in a reduced dimension space extracted from PCA over the test set. The resulting probing sample is not a recognizable digit but shares visual features with actual digits.*

To estimate the maps, 60000 probing samples were used with a 30% probability to have a bubble at each location on each sample. The standard deviation of each bubble was 4 pixels. We used the continuous probability belonging to a class obtained at the output of the CNN to correlate with the bubble masks. As Pearson correlation coefficients are used to compute the maps, such a choice of continuous probabilities should result in more robust estimations than correlations with binary class information, which clearly violates the assumptions underlying the correlation analysis. This implementation should be favored when the continuous probabilities are readily available. Here, the probabilities were obtained with the function 'predit_proba' from the Tensorflow/Keras distribution (Chollet et al., 2015).

Figure 3 shows the resulting potent features maps, expressed as the correlation between the bubble mask and the classification decision in the 2-D input space (the visual image). The maps are intended to highlight regions of the input representation that are weighed most in the classification decisions. These regions do not necessarily correspond to the actual shape of a digit (which will be targeted by represented features later on), for two reasons. First, the bubbles were chosen here with a relatively low resolution. Second, bubble masks were multiplied with actual input samples before being fed to the classifier, but the features of these samples are not taken into account in the reverse correlation as it was performed on the masks themselves.

Interestingly here for the digit "1" for instance, the potent features are actually *around* the digit: knowing that there are no active pixels in such surrounding regions appears to be the most efficient cue for deciding that the uniquely narrow shape of "1" was the input. For the digit "7", the

potent map highlights the top-right corner, which corresponds to the position of a sharp angle unique to “7”. In summary, while the potent features maps may not make immediate intuitive sense on their own, they do orient the analysis of the input set towards regions of interest, especially when interpreted along representative input samples as illustrated in Figure 3. Moreover, if the task was now to classify e.g. “7” versus all other digits, then the input space could be weighed to emphasize the top-right corner to simplify the new classifier.

We now compare the method’s outputs when the training set is available or not. The availability of training data is expected to provide faster and more robust convergence towards the features of interest. When the training set is unavailable, Method 1b can still be deployed. Here we used pseudo-random noise in a PCA-reduced space obtained from the test set (see section 1.1.2) as probing noise. An initial attempt with simple Gaussian noise led to unbalanced classification and no decision in 4 digit categories, which prevented interpretation of these categories. We used $10^4$ test set samples to build the PCA space, retaining 100 dimensions for the PCA. $10^4$ probing samples were then generated using normalized Gaussian noise in the PCA space and inverting back to the 2D original input space. This stochastic sampling procedure led to decisions covering the 10 categories in a more balanced fashion. A systematic investigation of the effect of these three parameters (number of test samples, PCA dimensions, number of probing samples) is provided in Supplemental Figures 1 and 2. As can be seen from Figure 3, the potent information obtained with Method 1b was highly similar to that uncovered by Method 1a, as confirmed by the strong correlations between maps across these methods with or without training set available (average Pearson coefficients for the 10 maps: M = .89 (SD = .10), $df = 783$, all $p < 10^{-3}$).

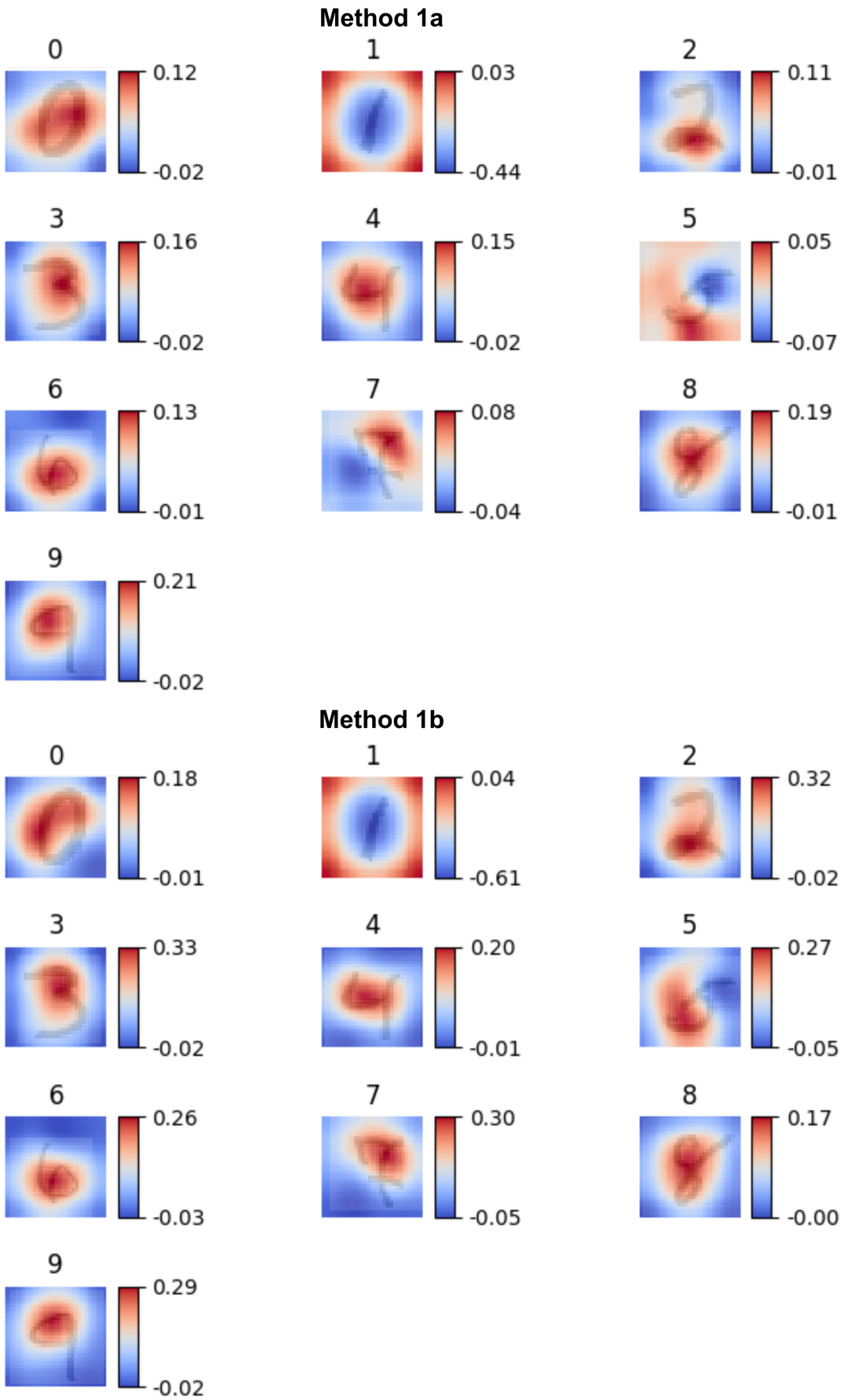

Method 1a
0
0.12
-0.02
1
0.03
-0.44
2
0.11
-0.01
3
0.16
-0.02
4
0.15
-0.02
5
0.05
-0.07
6
0.13
-0.01
7
0.08
-0.04
8
0.19
-0.01
9
0.21
-0.02
Method 1b
0
0.18
-0.01
1
0.04
-0.61
2
0.32
-0.02
3
0.33
-0.02
4
0.20
-0.01
5
0.27
-0.05
6
0.26
-0.03
7
0.30
-0.05
8
0.17
-0.00
9
0.29
-0.02

***Fig. 3*. Potent features maps for a CNN classifying handwritten digits.** *Each panel shows the potent features for a given target class (digit), expressed as correlation coefficients in the original 2-D input space. To facilitate interpretation, a typical test sample has been overlayed (dark shade) on each map. The top panels correspond to Method 1a with the training set available and the bottom panels correspond to Method 1b with the training set unavailable. Regions in red highlight the locations of the input space for which active pixels are associated to classification in the digit's class.*

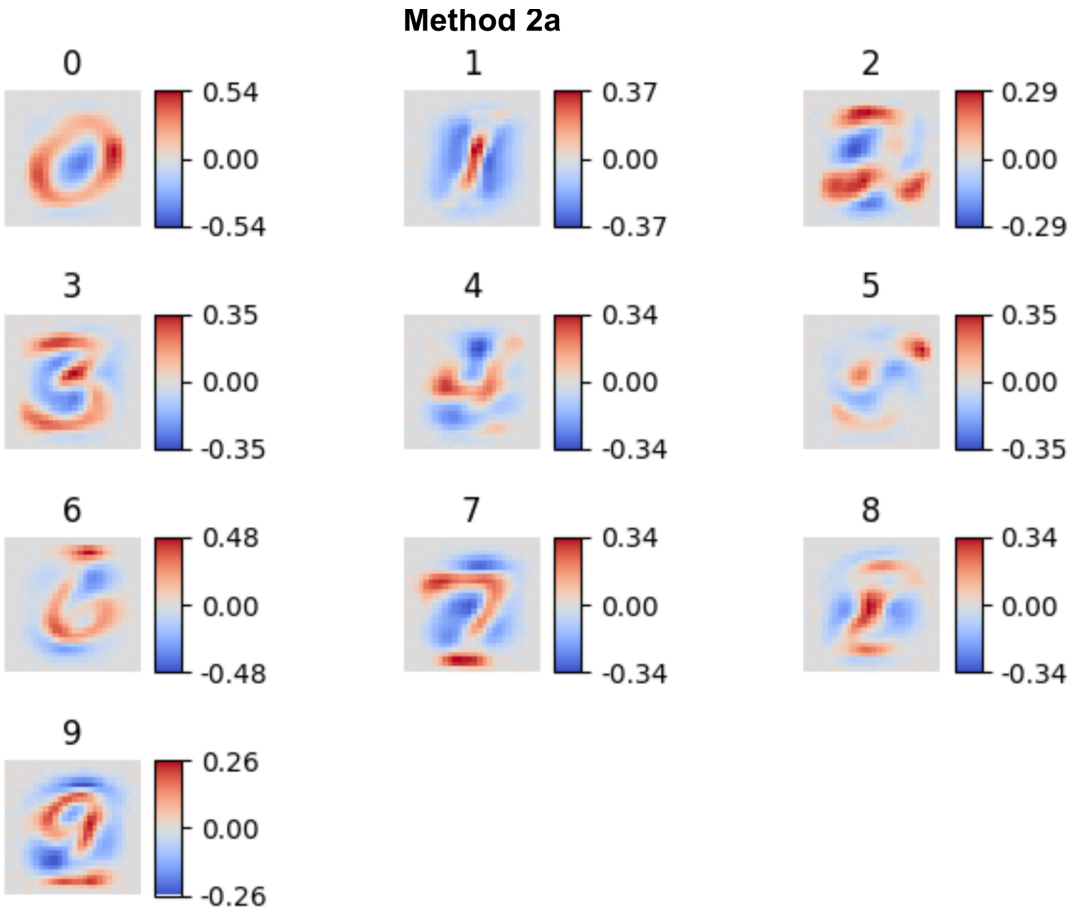
### 2.1.3 Represented features maps

We now probed the same trained CNN with Method 2, aimed at uncovering represented features maps. Such maps are can be interpreted as weighted averages of probing samples themselves, obtained by reverse correlation between the CNN's decisions and the input features. We used $10^4$ probing samples for Method 2a and the same number of pseudo-random samples for Method 2b.

Figure 4 shows the resulting represented features maps. Qualitatively, these maps look very different from the potent feature maps. This is mostly because the probing samples were directly generated in the input space, without any low-pass filtering, allowing the expression of finer details. Also, as expected with a reverse correlation approach, the represented features maps match the "best" stimulus of a category. As a result, the category of each map is readily identifiable, as it visually resembles the written digits' representations. In addition, 'negative' regions in blue further specify which features are expected to be absent to classify a given digit. Altogether, the represented features maps behave as expected in a digit classification task, and without any information about the classifier, it was able to recover representative digit-like shapes for each category.

We tested method 2b by generating probing samples in a reduced PCA-derived space, as for Method 1b. Again, Methods 2a and 2b provided strongly correlated maps (average Pearson coefficients for the 10 maps: M = .68 (SD = .18), $df = 783$, all $p < 10^{-3}$). It can nevertheless be noted that Method 2b tended to focus on the center of the input space. In particular, some border pixels were never associated with one or the other classification decision, leading to missing values when

computing the correlations.



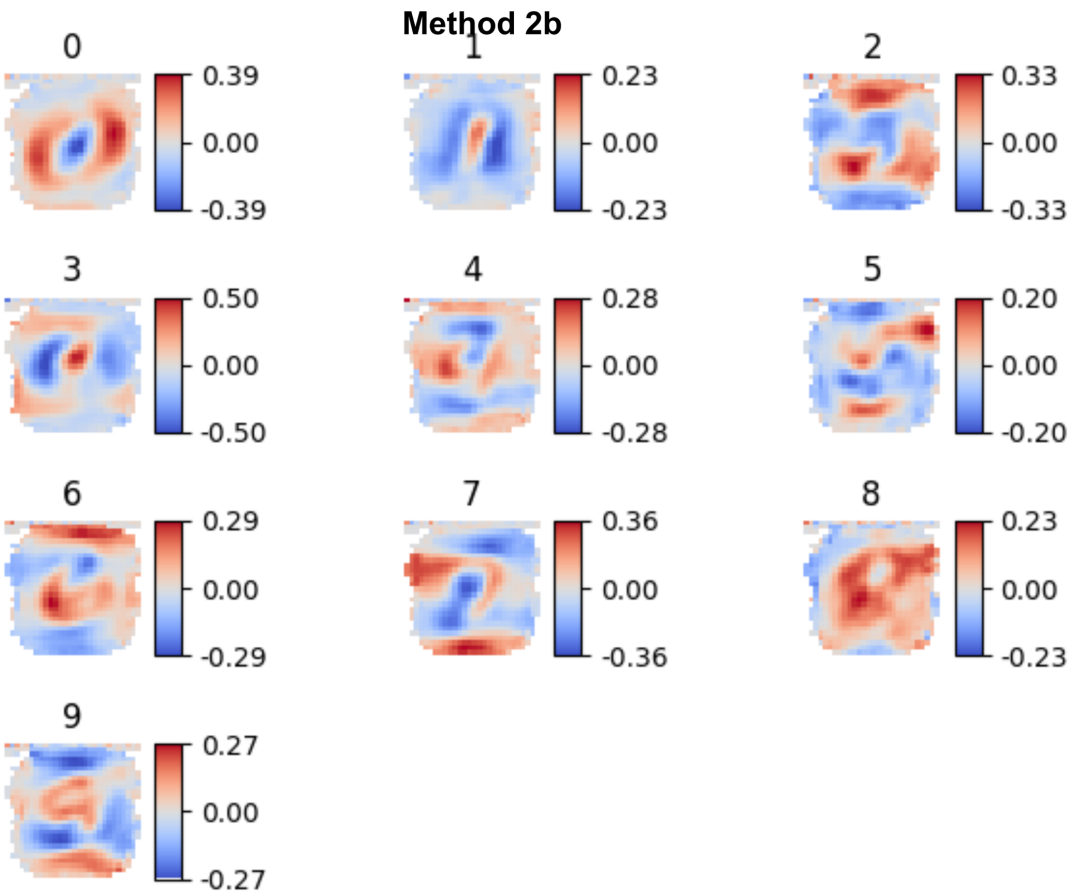



***Fig. 4. Represented features maps for a CNN classifying handwritten digits.*** *The different panels show the represented features for each target class (digit), expressed as correlation coefficients in the original 2-D input space. The top panels correspond to Method 2a with the training set available and the bottom panels correspond to Method 2b with the training set unavailable. The red portions of the maps indicate the input features most associated with a given class. They qualitatively resemble each digit. The blue portions of the maps indicate the input features that are most reliably not present for a given class.*

#### 2.1.4 Quantitative evaluation

So far, we have qualitatively described the output of the probing methods, and used such descriptions to gain some insights on the classifier's decisions, as would be the case in realistic use cases. However, as mentioned in the Introduction, there are already established interpretation methods for some classifiers' architectures. In particular, as we used a CNN for the digit classification task, we could compare the output of our algorithms to the Class Activation Mapping method (CAM, Zhou et al., 2016), a state-of-the art technique to visualize the features of a trained CNN.

To do so, we first adapted a metric proposed by Petsiuk et al. (BMVC, 2018) to evaluate the relevance of the visualized information. The gist of the metric is to evaluate the importance of features present in an interpretation mask by systematically adding or removing features from the mask. If the mask captured relevant features, then adding mask features should improve classification, whereas removing mask features should impair classification. Here, we used the potent features maps as interpretation masks.

In practice, the masks were first binarized by *z*-scoring the distribution of each mask and attributing the value 1 to the pixels which were above the mean and 0 otherwise. Then, for each sample of the test set, we added or deleted a proportion of features in the mask, i.e. mask values were randomly flipped from 1 to 0 or the reverse. Finally, masks were then blurred again, as in Petsiuk et al. (2018). This was achieved through a convolution with a 2D Gaussian window (standard deviation

of .3 pixels). Finally, masks were multiplied with testing samples. For each given proportion of insertions or deletion, the accuracy of the classifier is tested and the Area Under the Curve (AUC) of the accuracy vs. proportion curve was used as the performance metric. In the case of the insertion, the higher the AUC, the better the interpretation. Conversely, in the case of the deletion, the lower the AUC, the better the interpretation.

We applied this method on all of the samples of the test set for the MNIST task. For insertion on a potent features mask, we observed a mean AUC=.75 (SD=.23). For deletion on potent features mask, we observed a mean AUC=.14 (SD=.31). The difference between the two values was highly significant (two-sided t-test, t(19998)=158.58, $p < 10^{-5}$, $d_{cohen}$=2.24), demonstrating that the potent features were relevant for the classification task.

We then generated interpretation masks using the CAM method (Zhou et al., 2016). We used the default parameters of a grad-CAM implementation (https://raghakot.github.io/keras-vis/vis.visualization/#visualize_cam), selecting the gradient based class activation map (grad-CAM) that maximizes the outputs of all filters in the last convolution layer of the CNN as the mask being tested for each sample of the test set. From this mask we then applied the same steps as for the bubbles method to compute the insertion and deletion AUC metrics, i.e. thresholding and blurring. For insertion with a mask obtained with CAM, we observed a mean AUC=.56 (SD=.29). For deletion, we observed a mean AUC=.19 (SD=.30). The difference between the two values was highly significant (two-sided t-test, t(19998)=88.29, $p < 10^{-5}$, $d_{cohen}$=1.25), showing that the CAM masks also captured relevant information for the classification task, as expected. However, the difference between insertion and deletion AUC was smaller for the CAM masks than for the potent features masks (two-sided t-test, insertion: $t$(19998)=-51.03, $p < 10^{-5}$, $d_{cohen}$=.72; deletion: t(19998)=10.91, $p < 10^{-5}$, $d_{cohen}$=.72). This suggests that our method outperformed CAM for the quantitative metric chosen and the example tested here.

### 2.2 Speech vs. music

In this second example, we classified audio samples in a speech versus music task, to vary the input space and classifier's architecture. We used the GTZAN database composed of 132 excerpts of speech and music (Tzanetakis & Cook, 2002). The database was preprocessed to create samples with a fixed duration of 5 seconds, leading to a dataset of 768 samples. Those samples were randomly separated into a training set (691 excerpts) and a test set (77 excerpts, 10% of the dataset).

Following Patil et al. (2012), who performed an automatic classification of the musical timbre of short audio samples, sounds were first processed by an auditory model (Chi et al., 2005). The idea is to cast the input space into a representation that is interpretable in terms of auditory processing, unlike the raw waveform representation. Briefly, a filterbank corresponding to cochlear tonotopy was initially applied, followed by a 2-D Fourier analysis of the resulting time-frequency representation. The model output thus represents temporal modulations and spectral modulations contained in the input sound (Chi et al., 2005, Elliot & Theunissen, 2009). The 4-D resulting arrays, with dimensions of time, frequency, scale of spectral modulations, and rate of temporal modulation, are termed here Spectro-Temporal Modulation representations (STM). We averaged the time dimension over the 5s of each sample. Next, we applied a PCA to reduce dimensionality (30976 dimensions in our implementation: 128 frequency channels x 11 scales x 22 rates, reduced to150 dimensions to preserve 98% of the variance).

For classification, the output of the reduced PCA was fed to a Support Vector Machine (SVM) with a Radial Basis Function (RBF). All of these steps are identical to Patil et al. (2012), to which the reader is referred to for further details, as the specifics of the classifier are not critical to illustrate the probing method. Briefly, a grid search on the RBF was performed to determine the best set of parameters and the classifier accuracy was tested with a 10-fold cross-validation. We obtained an average classification accuracy, i.e. whether the classifier is classifying the STM of a sound to the correct music or speech class, of 94% (SD = 6%) with the 10-fold cross-validation and 98% on the test set.

Figure 5 shows the potent feature maps for the speech versus music classification task. For each case, we used 691 probing samples and a probability of .4 for the bubbles with standard deviation of

10 Hz in the frequency dimension, 5 Hz in the rate dimension, and 5 cycles/octave in the scale dimension. As the task is a binary classification, the maps for speech and music are simply mirror images of each other. we only present the one corresponding to speech (the reddest parts) vs. music (the bluest parts). Here, in order to test another implementation of the analysis framework, correlations were obtained using the binary decisions of the classifier and not continuous probabilities.

The potent regions of the auditory model STM representation appear to be mostly visible in the frequency dimension: speech can be best classified by looking at the input in a broad frequency range around 500 Hz, corresponding roughly to the position of the first formant in speech (Peterson & Barney, 1952). For the other dimensions, the classification depends on slow positive rates and low scales. In other words, the difference between speech and music was in the presence of slow modulations and broad spectral shapes for speech. Again, this matches prosodic and syllabic features of speech, together with the broad spectral shape of formants. By construction the two maps are complementary, but for music, a richness in spectrum, including high and low frequency regions, associated with fine spectral details (high scales) is characteristic of musical instruments (Elhilali, 2019), which have been designed to go beyond the physical constraints imposed by voice production. This is consistent with the blue parts of Figure 5.

In the case of the Method 1b, with the training set unavailable, we used pseudo-random noise generated in a PCA-reduced representation obtained with the test set. A PCA was first applied to the test set to reduce it to 50 dimensions and a uniform Gaussian white noise was generated on the 50 dimensions to generate samples in the reduced space. Each random reduced sample was then transformed into the original input space by applying the inverse PCA transformation. This procedure allowed to generate noisy samples with distribution relevant regarding the representative set of data relevant to the classification task. The information obtained by the two methods strongly correlate (average Pearson coefficients: M = .70, $df = 30974$, all $p < 10^{-5}$), even though differences are visible on the frequency dimension.

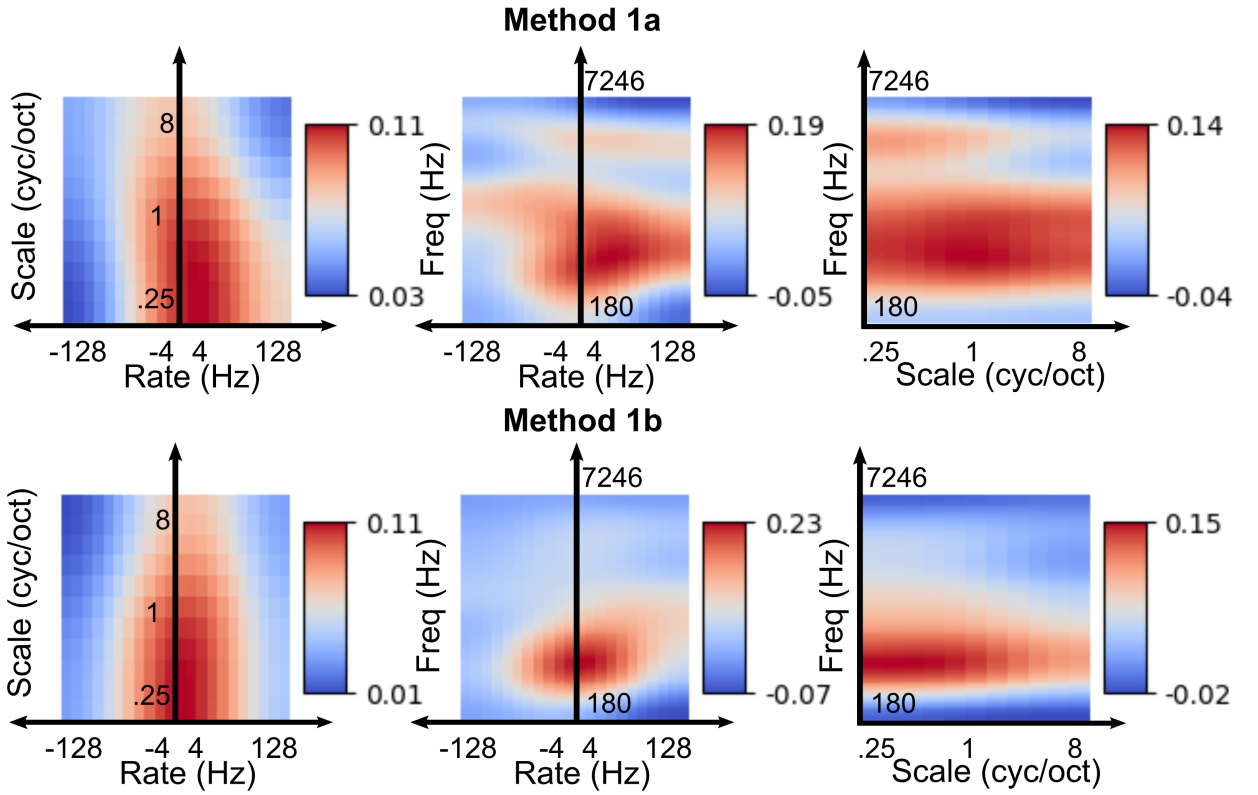



***Fig. 5. Potent maps for speech and music in the STM representation.*** *The 4-D auditory model representations are averaged over time and projected in the remaining three dimensions (frequency, scale, rates). Maps are expressed as correlation coefficients between the bubble masks and the binary classifier's decision. Method 1a uses the training set while Method 1b uses pseudo random noise. As this is a binary classification task, the speech and music masks are simply opposite versions of each other.*

Figure 6 shows the represented feature maps for the speech versus music classification task. Compared to Figure 5, these maps show in finer details what a speech sound or a music sound may look like in the STM representation, or, similarly, they indicate the "average" speech and music sounds learnt by the SVM. For speech, some formantic structure is now visible on the frequency dimension, associated with low rates typical of prosodic modulations (middle panels). These formantic regions extend to higher scales (right panels), perhaps because formants are superimposed on a harmonic structure for vowel sounds. Conversely, musical sounds more typically contain high modulation rates and spectral scales. These observations are consistent with previous analyses of STM representations (Elliott & Theunissen, 2009; Chi et al., 2005). Lastly, the information obtained

by methods a) and b) again strongly correlate (average Pearson coefficients for the 3 maps: M = .89, $df = 30974$, all $p < 10^{-5}$).

It should be noted that represented features depend on the acoustic characteristics of speech and music, but they also depend on the task of the classifier. Probing a classifier trained to discriminate speech from e.g. environmental sounds would likely provide different represented features for speech. This result may seem like a limitation of the method, but it also highlights the way an automatic classifier performs a binary task. This may be an important difference to keep in mind when comparing classifiers with human perception, which has to perform many concurrent tasks in parallel. For human perception, "opportunistic features" that depend both on sensory information and the task at hand have been suggested for auditory timbre recognition, a task not unlike the one probed here (Agus et al., 2019).

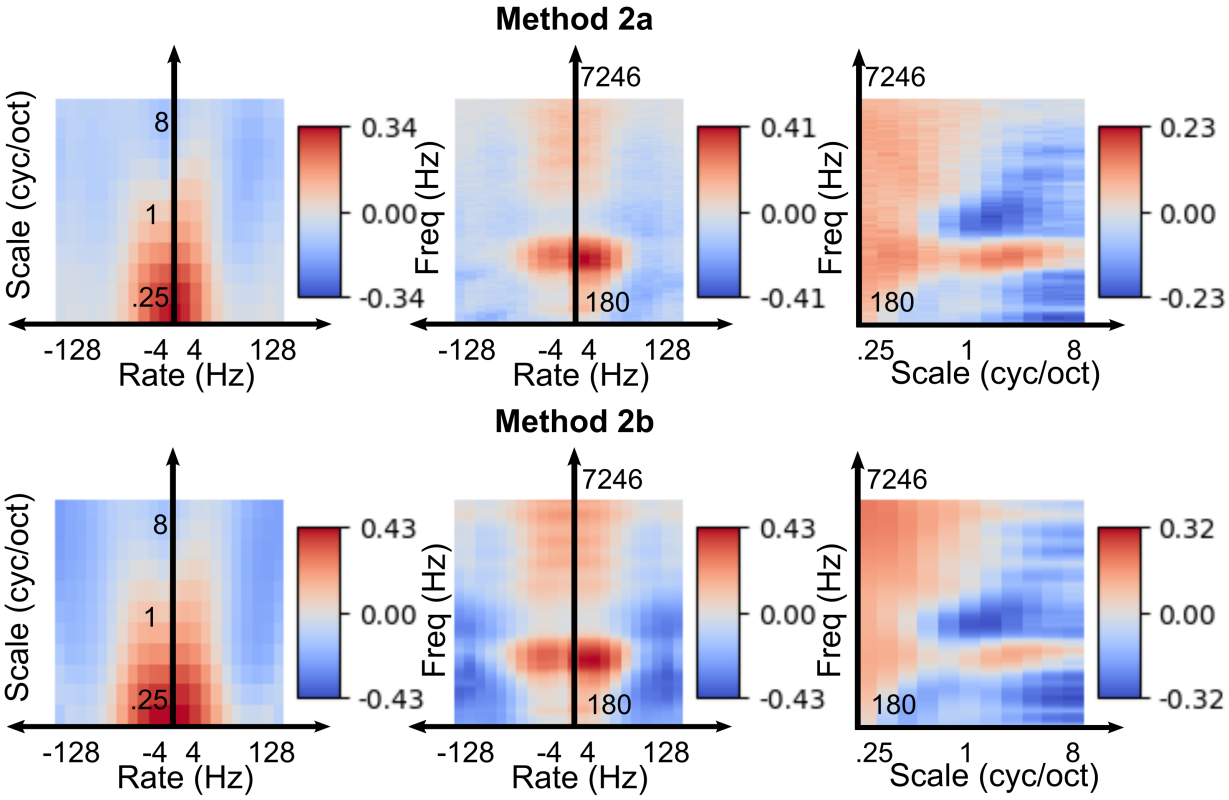



***Fig. 6. Represented STM representations for speech and music****. The 4-D STM representations are averaged over time and projected in the remaining three dimensions (frequency, scale, rates). Details as in Figure 5.*

533 **2.3 Classification of sleep stages**

534 In this third and last example, we interpreted the classification of sleep stages from
535 neurophysiological recordings by a random forest classifier. This represents yet another type of input
536 data and classifier architecture, with a direct link to a real and non-trivial classification problem.

537 We used the Sleep Physionet database, composed of 153 polysomnographic recordings (PSG)
538 from 78 subjects (41 females and 37 males) (Kemp et al., 2000). From this database, for simplicity,
539 15 subjects were randomly chosen for the training and 15 others for the testing. Each recording
540 includes 2 electroencephalographic (EEG), 1 electrooculographic (EOG) and 1 electromyographic
541 (EMG) derivations. The data is segmented in epochs of 30s, and each of these epochs have been
542 scored by human experts in 6 categories, according to established guidelines (Rechtschaffen & Kales,
543 1968): wake, NREM stage 1-4 and REM. There was also 1 additional non-sleep category
544 (movements) and 1 unscored category (total number of original categories: 8). Four of the 6 sleep
545 stages being subdivision of NREM sleep (NREM stage 1-4), we re-labelled the epochs according to
546 the three distinct physiological states: wake, NREM and REM sleep. Movement and unscored epochs
547 were discarded. The database was then preprocessed to extract the Welsh Power Spectrum Density
548 (PSD) between 0.5Hz and 30Hz on 30s samples of the EEG channels (Chambon et al., 2018). We
549 obtained a total of 12932 samples for the training and 13110 for the testing. Each sample had 75
550 frequency features. We then followed a published classification pipeline (Chambon et al., 2018) by
551 applying first a PCA to reduce the number of features to 30 and then trained a Random forest with a
552 grid search for the best hyperparameters to predict the sleep stage. The training led to a classification
553 accuracy of 99.1% on the training set and 74.9% on the test set, which is not perfect but higher than
554 the chance level of 33%.

555 Figure 7 shows the potent features maps expressed as correlation visualized in the 1-D input
556 space, the EEG power spectrum. For each case, 10000 probing samples were used, with a probability
557 of 0.3 to have a Gaussian bubble in each bin, with a standard deviation of 3 bins. Interestingly, the
558 potent features reflect what is known of the physiological differences between wake, NREM sleep
559 and REM sleep. In particular, wake led to negative correlation coefficients in the lower frequency

range (<5Hz) and positive correlation in the higher frequency range (>20Hz). This is in line with the predominance of fast, small-amplitude desynchronized rhythm and the absence of slow, large-amplitude synchronized oscillations in wakefulness. Conversely, NREM sleep showed positive correlation coefficients for the delta (1-6Hz) and sigma (~12Hz) range, which correspond to the frequency bands of the hallmark events of NREM sleep: slow waves (1-4Hz) and sleep spindles (11-16Hz). There are not many potent features for the REM category, which reflects the fact that REM is best categorized using a combination of EEG, EMG and EOG signals and not just the EEG data considered in this illustration.

In the case of Method 1b, a pseudo random noise obtained from the inversion of a 30-dimension PCA trained on the test set was used to probe the classifier. The potent information obtained in the two cases correlate strongly (average Pearson coefficients for the 3 maps: M = .97 (SD = .01), $df = 73$, all $p < 10^{-3}$).

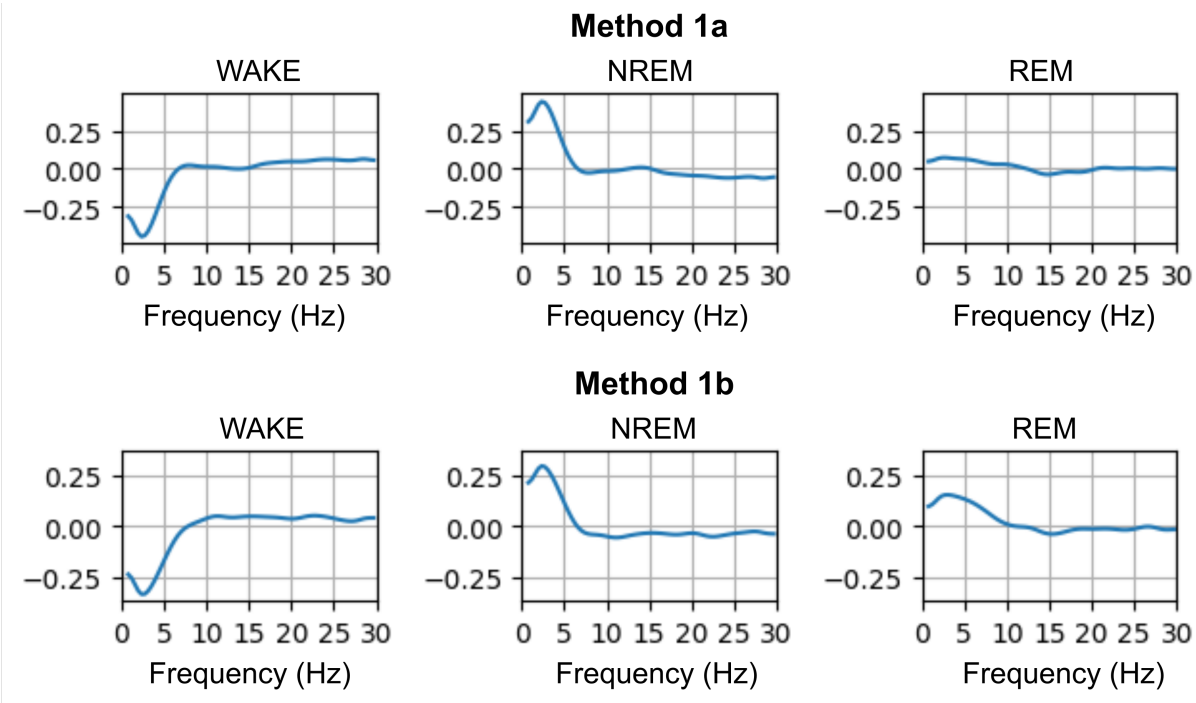



***Fig. 7. Potent features maps for sleep stages classification from polysomnography and a random forest classifier.*** *The panels show the potent information expressed as correlations in the input space to the classifier, that is, the power spectrum of the EEG channels of the polysomnographic recording. Positive values correspond to frequency regions where high power is associated to the target sleep*

*stage, whereas negative values correspond to frequency regions where low power is associated to the target sleep stage.*

Figure 8 shows the represented features obtained for the different sleep stages. As for the potent features, the features are aligned with the literature on sleep physiology. Indeed, quiet eyes-closed wakefulness is characterized by predominant alpha oscillations (8-11Hz) and the represented features for wake shows a peak within this frequency band. NREM is characterized by the presence of slow waves (1-4Hz) and sleep spindles (11-16Hz) and the represented features show again positive coefficients for these frequency bands. Finally, in addition to the absence of wake and NREM features, REM sleep is mostly characterized at the level of the EOG and EMG, which were not included in the classifier. Therefore, the represented (and potent) features for REM sleep unsurprisingly show less frequency-specific effects compared to wake or NREM sleep. It is interesting to observe that the represented features map provides somewhat different information than the potent features map. This reinforces the idea that the two methods can be complementary.

The represented information obtained for Method 2a and 2b correlate well stage (average Pearson coefficients for the 3 maps: M = .71 (SD = .05), $df = 73$, all $p < 10^{-3}$) although we can note specific discrepancies between the two methods in particular for the wake class.

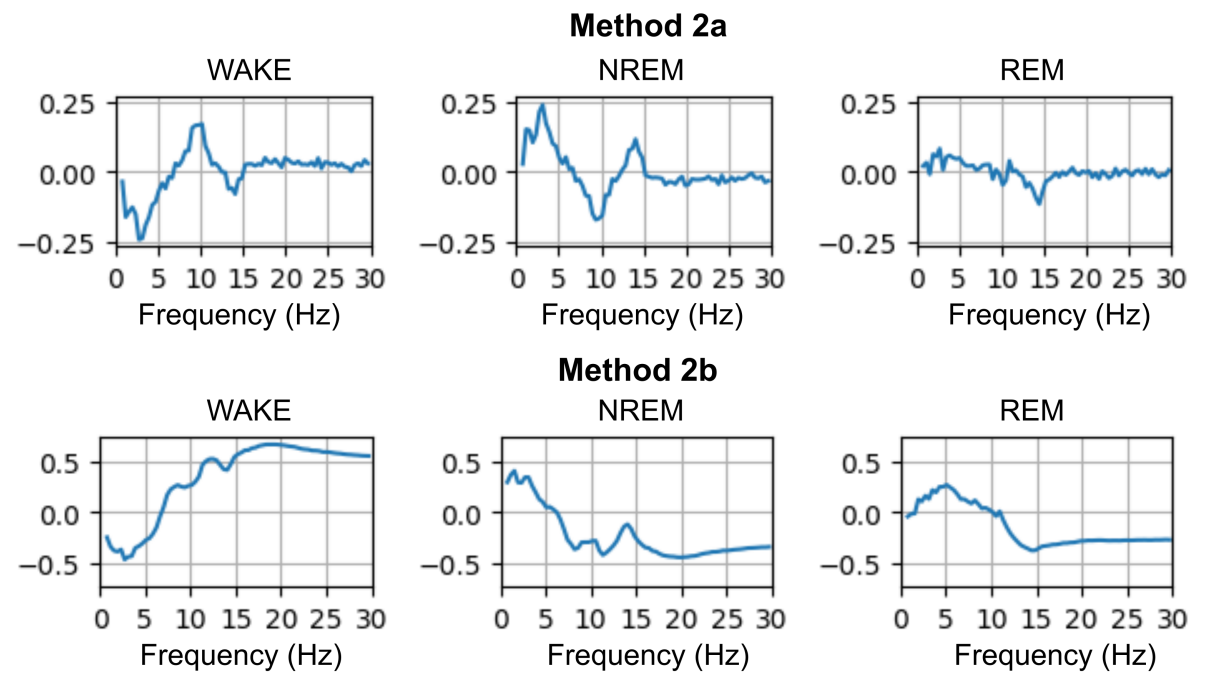


***Fig. 8. Represented features maps for a random forest classifying sleep stages from polysomnography power spectrum.*** *The plots show the represented information for each point of the input space, obtained with Method 2a (top) and Method 2b (bottom).*

# 3 Discussion

## 3.1 Summary

The method presented in this paper used a reverse correlation framework to visualize the input features discovered by an automatic classifier to reach its decisions. Over three examples using different kinds of classifiers (a CNN; an SVM with RBF; a random decision forest) and using different kinds of input representations (2-D visual images; 1-D audio samples converted to a 4-D auditory model; multi-channel 1-D EEG data converted to 1-D spectral features), we illustrated how the method could highlight relevant aspects of a classifier's operation. Moreover, by combining standard noise perturbation techniques with so-called bubbles (Gosselin & Shyns, 2002), we showed that the probing method can be focused either on potent features, related to the decision strategy of the classifier, or on represented features, related to the output classes' main characteristics.

## 3.2 Benefits

In the context of neuroscience and experimental psychology, we believe that there are many benefits in using a reverse correlation framework to interpret classifiers, as a way to complement other more specialized machine-learning interpretation techniques that will undoubtedly remain the preferred tools for purely data-science applications (Zhou et al., 2016; Ribeiro et al., 2016; Petsiuk et al., 2018; Borji & Lin, 2019; Xu et al., 2018).

First and foremost, reverse correlation is a familiar tool in the field of neuroscience and experimental psychology. It has proved useful to gain insights about stimulus features relevant to neural activity, at the single neuron level (Eggermont et al., 1983; Neri & Levi, 2006), network level (Arnal et al., 2015; Adolphs et al., 2005; Ringach & Shapley, 2010), or perceptual decision level (Gosselin & Shyns, 2001; Venezia et al., 2016). Applying it to interpret classifiers amounts to

translating and applying a familiar toolbox to another, conceptually similar problem of characterizing a black-box system, this time a computer-based algorithm instead of a brain or a person.

Second, the method is fully agnostic by design. It operates by default on the input space of the classifier, whatever this space might be. It does not make assumptions on the classifier's architecture or inner operations. Focusing on the input space rather than the classifier's architecture is especially desirable in situations where the classifier is not the main interest of study, but rather, the structure of the input dataset is.

Third, it can be applied to classifiers that have not been designed by the user, as it does not even require the availability of the training dataset. Access to labeled input data is helpful in improving the efficiency of the method, for instance by allowing to shape the perturbation noise, but this is a mild constraint: there are no interesting situations we can think of for which both the classifier and the type of data to classify would be unknown.

Finally, the output of the method is a visualization (with statistical evaluation if required) in the input space. Such a representation should make intuitive sense to the user, and the features discovered can be interpreted *a posteriori* in terms of attributes of the stimuli. If the representation does not make intuitive sense, then one possible benefit of the method is to help re-cast the input space into a more meaningful representation, as was done here in the audio example for which the waveform samples were pre-processed with an auditory model. This idea is further detailed in the "Perspectives" subsection.

### 3.3 Limitations

There are also limitations associated to the use of a reverse correlation approach to interpret automatic classifiers. Broadly speaking, these limitations follow those already described for reverse correlation in neuroscience.

First, reverse correlation is inspired from the analysis of linear systems, whereas machine-learning classifiers often rely on a cascade of non-linear operations to achieve computational power. The issue of non-linearity is well-described already in the reverse correlation literature, and its consequences have been clearly described (Theunissen et al., 2000). There are extensions to the

reverse correlation technique to describe lower-order non-linear interactions in the input space (Neri & Heeger, 2002). Such extensions could be applied to the interpretation of classifier's features. Interestingly, the reverse correlation approach bears some similarities with the "distillation" method from the machine learning literature (Hinton et al., 2015). Distillation consists in mimicking the behavior of a black-box classifier with an easily-interpretable classifier, such as a linear one (linear SVM, etc.). Both techniques can thus be viewed as attempting to find linear approximations of a classifier's operations, but their precise relationship remain to be investigated.

Second, the method has a number of free parameters: the probability and size of bubbles; the space to generate the probing noise with reduction dimension methods such as PCA when the training set is available. These are hyperparameters that, in our implementation, are not algorithmically constrained. In the examples above, the parameter space was explored heuristically. One suggested heuristic was to try and cover all output classes in a more or less balanced manner with the probing set. We have not developed any fitness criterion, i.e. a way to quantify the efficiency of the method for a given set of parameters. We do provide a quantitative metric based on an insertion/deletion technique (Petsiuk et al., 2018) that could conceivably be used as a fitness criterion if needed. However, we would argue that the iterative process for parameter tuning can be part of the interpretation process since finding the right probing structure in itself provides some information on the structure of the dataset. Also, assessing whether the discovered features make intuitive sense relies mostly on the knowledge and goals of the user.

Third, the method implicitly assumes that there are no invariances by translation in the classifier's algorithm. With reverse correlation, each point of the input space is treated independently of all others, so a feature discovered in one sub-part of the input space will not impact other, perhaps similar features in other sub-parts. This assumption is obviously falsified by CNN architectures, which are purposely designed to incorporate such invariances. In the CNN example illustrated here with digits recognition, this limitation was circumvented by the fact that all digits in the probing set were roughly spatially aligned. For the SVM on audio data, a time-averaging over the time dimension achieved a similar effect. For the EEG and sleep classification, spectral analysis was used. Thus,

mitigation strategies are available: a rough alignment of the probing data (spatially or temporally) should be sufficient for the reverse correlation to produce meaningful results, or a transform to a more robust representation. Another possible direction to address these invariance issues is to generate the probing noise in an appropriate space. Using a pre-processing PCA partly achieves this.

Fourth, here we probed the classifier in the same representation used to train the classifier, even though if that representation was not necessarily the raw input representation. This can be extended by probing the classifier in a space other than the training input space and applying an additional transform to feed the train classifier. For instance, as has been suggestion in visual studies using bubbles, spatial frequencies could be sampled with bubbles (e.g., Willenbockel et al., 2010; Caplette et al., 2014), transformed back into pixel images and fed to a classifier trained on images.

### 3.4 Perspectives

The probing method is technically applicable to any classifier's architecture with any kind of input data. It is thus beyond the scope of this final section to list all possible use cases in the context of neuroscience. We will simply provide a few suggestions, to illustrate perspectives of improvement and the kind of problems that could benefit from the probing method.

In the case of studying perceptual decisions, one possible insight gained from interpreting a classifier is the exploration of the input representation fed to the classifier. The hypothesis is that, the more appropriate the representation, the more explainable the classifier should be. For instance, one could assume that the massively non-linear transformations of auditory and visual information that characterize perceptual systems serve to build a stimulus manifold within which perceptual boundaries are approximately linear (Georgopoulos et al., 1986; Jazayeri & Movshon, 2006; Kell et al., 2018). So, with the correct representation, a classifier modeling a perceptual decision process should be easily interpretable, or at least more easily interpretable than if the input representation was not reflecting perceptual processing. It is with this hypothesis in mind that the audio samples of the example illustrated above were first processed with an auditory model. Even though there are successful deep learning models operating on the raw audio waveforms (e.g. Wavenet, Oord et al., 2006), it is not expected that interpreting them in terms of waveform features will be meaningful. For

instance, inaudible phase shifts between frequency components in the input would impact the waveform representation, but should not change the classifier's decision. An auditory model, in contrast, incorporates transforms inspired by the neurophysiology of the hearing system. If the features extracted resemble those available to a human observer, then they should be revealed when probing a classifier. In fact, the ease of interpreting a classifier feature could be a proxy to evaluate an input representation's adequation to a perceptual task.

Another possible application concerns the construction of "ideal observer" models (Geisler, 2004). The idea of an ideal observer model is to compute the best theoretical performance on a task, given a set of assumptions (classically, endowing the ideal observer with unbiased decision criteria, perfect and unlimited memory, and so on). This upper performance boundary is then compared to the observed performance with human participants or neural recordings. When considering classification or discrimination tasks, and when a formal model of the ideal observer is unavailable, it can be of interest to build pseudo-ideal observer models with machine learning classifiers. The advantage of our probing method is then that the classifier's strategy can be directly compared to a reverse correlation analysis of neural or psychophysical data, to ask whether the classifier and the experimental observer used the same decision features.

Finally, the general benefits of interpreting classifiers also apply to the field of neuroscientific applications. In a broad sense, probing is intended to help an expert making sense of a classifier's strategy. If the features discovered through probing fit a theoretical model, this would reassure the expert that the performance relies on reasonable principles, which is especially important in clinical applications. In return, the expert's intuition may also help improve the classifier, for instance by simplifying its input representation through pre-processing, and so hopefully making it less brittle to irrelevant variations in input that may have been picked up by overfitting during training (Goodfellow et al., 2015). The potent features could be particularly useful to reduce the complexity of a classifier. Based on the potent features map, it may be possible to select a subset of important and intelligible features, which can then be used to build a more computationally efficient classifier, for very large dataset and/or for real-time processing.

## 4 Conclusions

We presented a novel method to interpret machine-learning classifiers that is agnostic, versatile and well-suited to applications in the neuroscience domain. Based on the reverse correlation framework, our method uses stochastic perturbation of inputs to observe the classifier's output. It then visualizes, in the input space, the potent and represented features discovered by the classifier for each category. This method can be applied to any kind of classifier. It displays the same well-established benefits and limitations as reverse correlation as applied to psychophysical or neural data. Our hope is that such a method can provide a simple and generic interface between neuroscientists and machine-learning tools.

## 5 References


Ahumada Jr, A., & Lovell, J. (1971). Stimulus features in signal detection. The Journal of the Acoustical Society of America, 49(6B), 1751-1756. https://doi.org/10.1121/1.1912577

Adolphs, R., Gosselin, F., Buchanan, T. W., Tranel, D., Schyns, P., & Damasio, A. R. (2005). A mechanism for impaired fear recognition after amygdala damage. Nature, 433(7021), 68-72. https://doi.org/10.1038/nature03086

Andrillon, T., Solelhac, G., Bouchequet, P., Romano, F., Le Brun, M. P., Brigham, M., ... & Léger, D. (2020). Revisiting the value of polysomnographic data in insomnia: more than meets the eye. Sleep medicine, 66, 184-200. https://doi.org/10.1016/j.sleep.2019.12.002

Arnal, L. H., Flinker, A., Kleinschmidt, A., Giraud, A. L., & Poeppel, D. (2015). Human screams occupy a privileged niche in the communication soundscape. Current Biology, 25(15), 2051-2056.https://doi.org/10.1016/j.cub.2015.06.043

Benjamini, Y., & Hochberg, Y. (1995). Controlling the false discovery rate: a practical and powerful approach to multiple testing. Journal of the Royal statistical society: series B (Methodological), 57(1), 289-300. https://doi.org/10.1111/j.2517-6161.1995.tb02031.x

Borji, A., & Lin, S. (2019). White Noise Analysis of Neural Networks. arXiv preprint arXiv:1912.12106.

Chi, T., Ru, P., & Shamma, S. A. (2005). Multiresolution spectrotemporal analysis of complex sounds. The Journal of the Acoustical Society of America, 118(2), 887-906. https://doi.org/10.1121/1.1945807

Elhilali, M. (2019). Modulation Representations for Speech and Music. In Timbre: Acoustics, Perception, and Cognition (pp. 335-359). Springer. https://doi.org/10.1007/978-3-030-14832-4_12

Elliott, T. M., & Theunissen, F. E. (2009). The modulation transfer function for speech intelligibility. PLoS computational biology, 5(3). https://doi.org/10.1371/journal.pcbi.1000302

Eggermont, J. J., Johannesma, P. I. M., & Aertsen, A. M. H. J. (1983). Reverse-correlation methods in auditory research. Quarterly reviews of biophysics, 16(3), 341-414. https://doi.org/10.1017/s0033583500005126

Caplette, L., West, G., Gomot, M., Gosselin, F., & Wicker, B. (2014). Affective and contextual values modulate spatial frequency use in object recognition. Frontiers in psychology, 5, 512.

Chollet and others (2015) Keras, https://keras.io

Deng, L. (2012). The mnist database of handwritten digit images for machine learning research [best of the web]. IEEE Signal Processing Magazine, 29(6), 141-142.https://doi.org/10.1109/msp.2012.2211477

Geisler, W. S. (2004). "Ideal Observer analysis," in Visual Neurosciences, eds L. Chalupa and J. Werner (Boston, MA: MIT press), 825–837.

Georgopoulos, A. P., Schwartz, A. B., & Kettner, R. E. (1986). Neuronal population coding of movement direction. Science, 233(4771), 1416-1419. https://doi.org/10.1126/science.3749885

Gosselin, F., & Schyns, P. G. (2002). RAP: A new framework for visual categorization. Trends in Cognitive Sciences, 6(2), 70-77. https://doi.org/10.1016/s1364-6613(00)01838-6

Gosselin, F., & Schyns, P. G. (2001). Bubbles: a technique to reveal the use of information in recognition tasks. Vision research, 41(17), 2261-2271. https://doi.org/10.1016/s0042-6989(01)00097-9

Gosselin, F., & Schyns, P. G. (2003). Superstitious perceptions reveal properties of internal representations. Psychological science, 14(5), 505-509. https://doi.org/10.1111/1467-9280.03452

Goodfellow, I. J., Shlens, J., & Szegedy, C. (2014). Explaining and harnessing adversarial examples. arXiv preprint arXiv:1412.6572. https://doi.org/10.5220/0006123702260234

Green, D. M., & Swets, J. A. (1966). Signal detection theory and psychophysics (Vol. 1). New York: Wiley.

Guidotti, R., Monreale, A., Ruggieri, S., Turini, F., Giannotti, F., & Pedreschi, D. (2018). A survey of methods for explaining black box models. ACM computing surveys (CSUR), 51(5), 1-42. https://doi.org/10.1145/3236009

Hinton, G., Vinyals, O., & Dean, J. (2015). Distilling the knowledge in a neural network. arXiv preprint arXiv:1503.02531.

Jazayeri, M., & Movshon, J. A. (2006). Optimal representation of sensory information by neural populations. Nature neuroscience, 9(5), 690-696. https://doi.org/10.1038/nn1691

Jordan, M. I., & Mitchell, T. M. (2015). Machine learning: Trends, perspectives, and prospects. Science, 349(6245), 255-260. https://doi.org/10.1126/science.aaa8415

Kell, A. J., Yamins, D. L., Shook, E. N., Norman-Haignere, S. V., & McDermott, J. H. (2018). A task-optimized neural network replicates human auditory behavior, predicts brain responses, and reveals a cortical processing hierarchy. Neuron, 98(3), 630-644. https://doi.org/10.1016/j.neuron.2018.03.044

Kriegeskorte, N., & Douglas, P. K. (2018). Cognitive computational neuroscience. Nature neuroscience, 21(9), 1148-1160. https://doi.org/10.1038/s41593-018-0210-5

LeCun, Y., Bengio, Y., & Hinton, G. (2015). Deep learning. nature, 521(7553), 436-444. https://doi.org/10.1038/nature14539

Lillicrap, T. P., & Kording, K. P. (2019). What does it mean to understand a neural network?. arXiv preprint arXiv:1907.06374.

Maris, E., & Oostenveld, R. (2007). Nonparametric statistical testing of EEG-and MEG-data. Journal of neuroscience methods, 164(1), 177-190. https://doi.org/10.1016/j.jneumeth.2007.03.024

Neri, P., Parker, A. J., & Blakemore, C. (1999). Probing the human stereoscopic system with reverse correlation. Nature, 401(6754), 695-698. https://doi.org/10.1038/44409

Neri, P., & Heeger, D. J. (2002). Spatiotemporal mechanisms for detecting and identifying image features in human vision. Nature neuroscience, 5(8), 812-816. https://doi.org/10.1038/nn886

Neri, P., & Levi, D. M. (2006). Receptive versus perceptive fields from the reverse-correlation viewpoint. Vision research, 46(16), 2465-2474. https://doi.org/10.1016/j.visres.2006.02.002

Oord, A. V. D., Dieleman, S., Zen, H., Simonyan, K., Vinyals, O., Graves, A., ... & Kavukcuoglu, K. (2016). Wavenet: A generative model for raw audio. arXiv preprint arXiv:1609.03499.

Patil, K., Pressnitzer, D., Shamma, S., & Elhilali, M. (2012). Music in our ears: the biological bases of musical timbre perception. PLoS computational biology, 8(11). https://doi.org/10.1371/journal.pcbi.1002759

Peterson, G. E., & Barney, H. L. (1952). Control methods used in a study of the vowels. The Journal of the acoustical society of America, 24(2), 175-184. https://doi.org/10.1121/1.1906875

Petsiuk, V., Das, A., & Saenko, K. (2018). Rise: Randomized input sampling for explanation of black-box models. arXiv preprint arXiv:1806.07421.

Rechtschaffen, A., & Kales, A.A. (1968). A manual for standardized terminology, techniques and scoring system for sleep stages in human subjects. Brain information service.

Richards, B. A., Lillicrap, T. P., Beaudoin, P., Bengio, Y., Bogacz, R., Christensen, A., ... & Gillon, C. J. (2019). A deep learning framework for neuroscience. Nature neuroscience, 22(11), 1761-1770. https://doi.org/10.1038/s41593-019-0520-2

Ribeiro, M. T., Singh, S., & Guestrin, C. (2016). " Why should i trust you?" Explaining the predictions of any classifier. In Proceedings of the 22nd ACM SIGKDD international conference on knowledge discovery and data mining (pp. 1135-1144). https://doi.org/10.1145/2939672.2939778

Ringach, D., and Shapley, R. (2004). Reverse correlation in neurophysiology. Cogn. Sci. 28, 147–166. doi: 10.1207/s15516709cog2802_2

Rosch, E. (1983). Prototype classification and logical classification: The two systems. New trends in conceptual representation: Challenges to Piaget's theory, 73-86.

Theunissen, F. E., Sen, K., & Doupe, A. J. (2000). Spectral-temporal receptive fields of nonlinear auditory neurons obtained using natural sounds. Journal of Neuroscience, 20(6), 2315-2331. https://doi.org/10.1523/jneurosci.20-06-02315.2000

Tzanetakis, G., & Cook, P. (2002). Musical genre classification of audio signals. IEEE Transactions on speech and audio processing, 10(5), 293-302. https://doi.org/10.1109/tsa.2002.800560

Venezia, J. H., Hickok, G., & Richards, V. M. (2016). Auditory “bubbles”: Efficient classification of the spectrotemporal modulations essential for speech intelligibility. The Journal of the Acoustical Society of America, 140(2), 1072-1088. https://doi.org/10.1121/1.4960544

Willenbockel, V., Sadr, J., Fiset, D., Horne, G. O., Gosselin, F., & Tanaka, J. W. (2010). Controlling low-level image properties: the SHINE toolbox. Behavior research methods, 42(3), 671-684.

Wiener, N. (1966). Nonlinear problems in random theory. Nonlinear Problems in Random Theory, by Norbert Wiener, pp. 142. ISBN 0-262-73012-X. Cambridge, Massachusetts, USA: The MIT Press, August 1966.(Paper), 142. https://doi.org/10.1063/1.3060939

Xu, T., Garrod, O., Scholte, S. H., Ince, R., & Schyns, P. G. (2018). Using psychophysical methods to understand mechanisms of face identification in a deep neural network. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops (pp. 1976-1984). https://doi.org/10.1109/cvprw.2018.00266

Zihni, E., Madai, V. I., Livne, M., Galinovic, I., Khalil, A. A., Fiebach, J. B., & Frey, D. (2020). Opening the black box of artificial intelligence for clinical decision support: A study predicting stroke outcome. Plos one, 15(4), e0231166.https://doi.org/10.1371/journal.pone.0231166

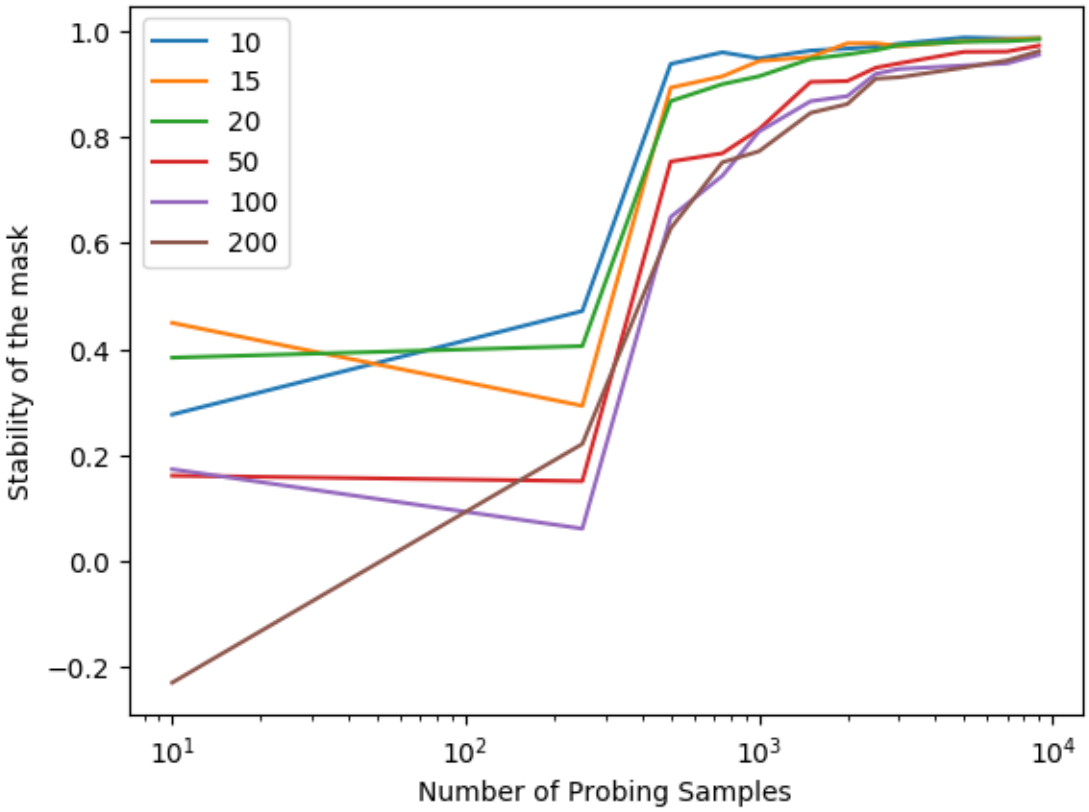
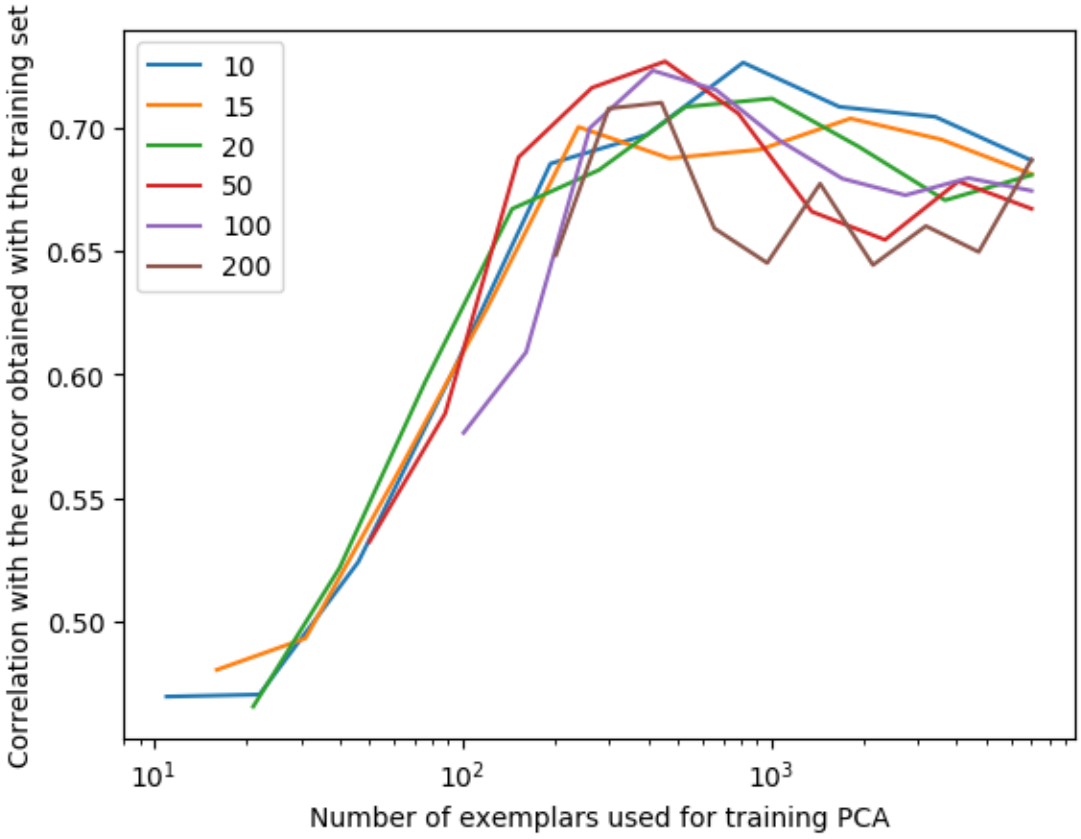
Zhou, B., Khosla, A., Lapedriza, A., Oliva, A., & Torralba, A. (2016). Learning deep features for discriminative localization. In Proceedings of the IEEE conference on computer vision and pattern recognition (pp. 2921-2929). https://doi.org/10.1109/cvpr.2016.319

**Acknowledgements:** *Research supported by grants ANR-16-CONV-0002 (ILCB), ANR-11-LABX-0036 (BLRI) and the Excellence Initiative of Aix-Marseille University (A*MIDEX) (ET), ANR-10-LABX-0087 IEC and ANR-10-IDEX-0001-02 PSL (DP). TA was supported by the Human Frontier Science Program (LT000362/2018-L).*

**Declarations of interest:** none.

## Supplementary Materials

**Supplemental Figure 1.** Influence of the number of probing samples on the stability of the estimation in the digit classification example (Method 1b, section 2.1.2). The stability corresponds to the correlation between the interpretation maps obtained between two successive number of probing samples. Each curve corresponds to a different number of PCA dimensions used to generate the pseudo random noise. An order of magnitude of $10^3$ to $10^4$ generated probing samples seems sufficient to obtain a stable estimation. It is noticeable that the maps obtained with a smaller number of PCA dimensions converge more quickly than those obtained with more dimensions.



**Supplemental Figure 2.** Influence of the number of exemplars used to obtain the PCA for generating the pseudo-random probing samples in the digit classification example (Method 1b, section 2.1.2). Each curve corresponds to a different number of dimensions retainef in the PCA. $10^2$ to $10^3$ test samples seem sufficient when the training set is unavailable to converge toward a map correlating with the reference one obtained when the training set is available.