

## Obtaining scalable and accurate classification in large-scale spatio-temporal domains

Igor Vainer · Sarit Kraus · Gal A. Kaminka · Hamutal Slovin

Received: 21 March 2010 / Revised: 7 July 2010 / Accepted: 16 July 2010 /  
Published online: 4 November 2010  
© Springer-Verlag London Limited 2010

**Abstract** We present an approach for learning models that obtain accurate classification of data objects, collected in large-scale spatio-temporal domains. The model generation is structured in three phases: spatial dimension reduction, spatio-temporal features extraction, and feature selection. Novel techniques for the first two phases are presented, with two alternatives for the middle phase. We explore model generation based on the combinations of techniques from each phase. We apply the introduced methodology to data-sets from the Voltage-Sensitive Dye Imaging (VSDI) domain, where the resulting classification models successfully decode neuronal population responses in the visual cortex of behaving animals. VSDI is currently the best technique enabling simultaneous high spatial (10,000 points) and temporal (10 ms or less) resolution imaging from neuronal population in the cortex. We demonstrate that not only our approach is scalable enough to handle computationally challenging data, but it also contributes to the neuroimaging field of study with its decoding abilities. The effectiveness of our methodology is further explored on a data-set from the hurricanes domain, and a promising direction, based on the preliminary results of hurricane severity classification, is revealed.

**Keywords** Classification · Spatio-temporal · Application · Brain imaging · Neural decoding · Visual cortex · Hurricane satellite imagery

---

I. Vainer (✉) · S. Kraus · G. A. Kaminka  
Department of Computer Science, Bar-Ilan University, 52900 Ramat Gan, Israel  
e-mail: iv@013.net

H. Slovin  
The Leslie and Susan Gonda Multidisciplinary Brain Research Center,  
Bar-Ilan University, 52900 Ramat Gan, Israel

H. Slovin  
The Mina and Everard Goodman Faculty of Life Sciences,  
Bar-Ilan University, 52900 Ramat Gan, Israel

## 1 Introduction

There is much interest in applying machine learning in domains with large-scale spatio-temporal characteristics. Examples range from learning patterns and discriminating cognitive brain states using functional Magnetic Resonance Imaging (fMRI) [1, 4, 10, 16, 18–20, 23, 33, 34], to developing techniques for classification of brain signals in Brain Computer Interfaces (BCI) [5, 14, 22, 25, 32, 35], performing automated video classification [30], computer worm detection [26] and many more.

However, many existing techniques prove insufficient when the data is temporally and spatially large (consisting of a large number of variables associated with locations in space, whose values may change over a relatively long period of time). Classification in such cases often becomes computationally infeasible. Raw data collected along the time course in a high-resolution space results in hundreds of thousands of features, for which classical, straight-forward machine learning approaches become ineffective in practice. While there have been attempts at addressing these challenges (e.g., [5, 34]), they have proven insufficient.

In this work, we present a methodology for both overcoming the scalability challenge and exploiting the spatio-temporal properties of the data for classification. Our methodology is based on common machine learning elements and is comprised of three phases. First, we present a greedy *pixel selection* technique, i.e. choosing the most discriminative spatial characteristics within the full spatial range in a sample's space, based on the random subspace method [13]. Second, we provide two alternative *feature extraction* procedures, applied to the spatially reduced samples produced by the first phase: features as pixels in time and spatial averaging of pixel groups based on inter-pixel correlation. Finally, we employ a simple and yet effective *feature selection* based on information gain filtering. We evaluate the methods in two distinct domains, described in the next paragraphs.

First, we evaluate our methodology in the neuroimaging domain and demonstrate how it can be used to decode neuronal population responses in the visual cortex of monkeys, collected using Voltage-Sensitive Dye Imaging (VSDI) [24]. VSDI is capable of measuring neuronal population responses at high spatial (10,000 pixels of size  $60 \times 60$  to  $170 \times 170 \mu\text{m}^2$  each) and temporal (10 ms or less) resolutions. The data generated consists of tens of thousands of pixels (numeric values, correlated to locations in space), rapidly changing during the time course. Using the methods we develop, it is possible to carry out high-quality classification of such massive data, in a computationally feasible manner. We show how to identify those specific properties of the data that carry the most discriminative nature. While first attempts to decode neuronal population responses collected using VSDI were performed in [3], no machine learning methods were used—a specially designed statistical approach of pooling rules was developed (relying on the amplitude of the response and other neuronal characteristics). To the best of our knowledge, this is the first time where machine learning techniques are applied in the VSDI imaging domain.

We then further evaluate the methods using hurricane data. Here, we analyze historical data of the Atlantic region—satellite images and hurricane tracks—in attempt to classify the hurricane severity group by generating a data-set based on plain periodical satellite shots along the time course. The kind of application we explore here examines how our methodology handles the challenges proposed in a domain highly different than the VSDI.

The rest of this document is organized as follows: Sect. 2 is the core section of this work which describes the three phase methodology for spatio-temporal classification. Section 3 presents the empirical evaluation in the VSDI domain, including thorough analysis of the experiment results. In Sect. 4, we introduce the first results of applying our methods in the hurricanes domain, after presenting the challenges and discussing the differences between

the two domains. The insights on the performance of our methodology are discussed in detail in Sect. 5. Section 6 contains a review of related work. Section 7 concludes.

## 2 Spatio-temporal classification process

In this section, we present a *three phase methodology* for building scalable models for spatio-temporal data classification. To describe our methodology, we first formalize the problem in Sect. 2.1. We then provide a brief overview of the main phases of the methodology used and describe each of the phases in detail. Finally, we provide a detailed example run of the algorithms.

### 2.1 Problem formalization

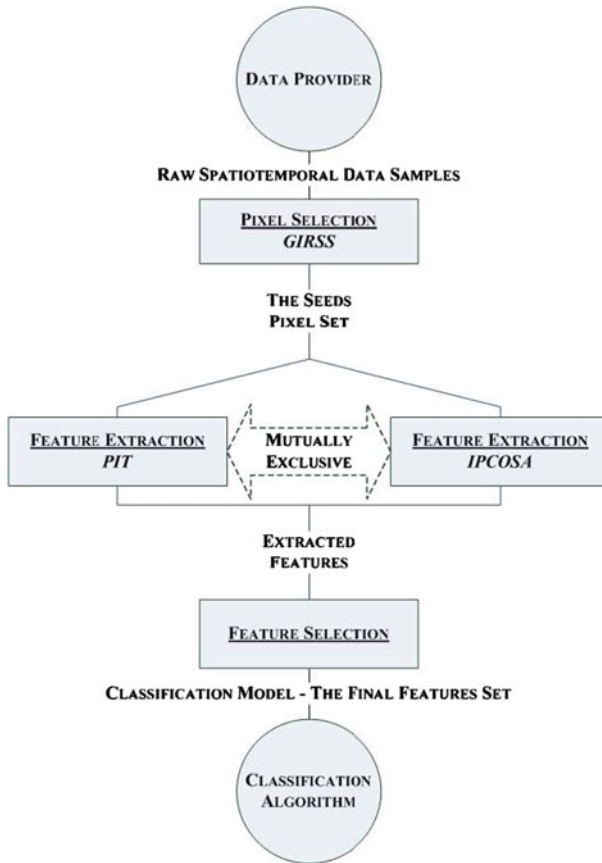
A spatio-temporal domain contains  $n$  pixels that constitute the global pixel set  $P = \{p_1, p_2, \dots, p_n\}$ . Every pixel  $p_i$ ,  $i \in \{1, \dots, n\}$  represents a concrete location in space, in which a series of  $m$  contiguous values in time is measured. The intervals between each two consequent values in time are equal. In turn,  $p_i^t$ ,  $i \in \{1, \dots, n\}$ ,  $t \in \{1, \dots, m\}$  indicates the specific time-frame  $t$  along the time course, at which the value of  $p_i$  is measured. In fact,  $p_i^t$  represents the pixel-in-time combination of pixel  $p_i$  and time  $t$ .

A finite training samples set of size  $k$  in the spatio-temporal domain is defined as:  $S = \{s_1, s_2, \dots, s_k\}$ , where a single sample  $s_l$ ,  $l \in \{1, \dots, k\}$  is a set of vectors:  $s_l = \{\overline{p}_1, \dots, \overline{p}_n\}$ , where a vector  $\overline{p}_i = \{v_1^i, \dots, v_m^i\}$ ,  $v_t^i \in \mathbb{R}$ ,  $t \in \{1, \dots, m\}$  denotes the actual  $m$  values along the time course, measured for the pixel  $p_i$  in the sample  $s_l$ . Each training sample  $s_l \in S$  is labeled with a class label  $c \in C$ . For an infinitely large universal set  $U$  of all possible unlabeled samples  $u = \{\overline{p}_1, \dots, \overline{p}_n\}$ ,  $u \in U$ , the classification problem is to build a model that approximates classification functions of the form  $f : U \rightarrow C$ , which map unclassified samples from  $U$  to the set of class labels  $C$ .

In the next sections, we describe each of the phases of our methodology in detail. Section 2.2 presents *GIRSS*, a technique for selecting the pixels that have the most discriminative characteristics within the full spatial range of a sample's space. Next, in Sect. 2.3, we introduce two alternative techniques for extracting the features from the pixels selected in the first phase—the *PIT*, a simple pixel-in-time approach, and the *IPCOSA*, a spatial averaging method based on inter-pixel correlation. The third phase described in Sect. 2.4 presents an effective application of feature selection on the product of the second phase, to further improve the abilities of the remaining features that constitute the generated models. Figure 1 sketches the outline for our methodology.

### 2.2 Pixel selection via greedy improvement of random spatial subspace

Below, we describe *GIRSS* (Greedy Improvement of Random Spatial Subspace). The technique uses common machine learning tools in order to reveal the most informative pixels, which will define the features to be used for classification. The discriminative nature of the selected pixels stems from analyzing their measured values along the sample time course. Due to the high spatial and temporal resolutions of the domains in question, the data is comprised of hundreds of thousands of data-points. Hence, using the most granular, basic values of the sample's space as features will lead to an extremely high dimensional feature space, rendering classification, or even feature dimensionality reduction techniques, unfeasible.



**Fig. 1** The outline of the classification process methodology

We present here a greedy approach based on the random subspace method [13]. The method selects (by iterative refinement) the set of pixel subsets from which we can eventually derive the sought-after pixel set.

### 2.2.1 The *GIRSS* algorithm

In Algorithm 1, we randomly generate  $r$  pixel subsets of a requested size  $u$  (number of pixels in a subset). Handling small pixel subsets yields an easier handling of a reduced spatial dimension. However, in order to cover a large portion of pixels (features) in the data and to determine their usefulness, we need to rely on a wide-enough exploration of such subsets.

The classification capabilities of each of the generated pixel subsets are roughly evaluated using our pixel set evaluation method (Algorithm 2). This method is a heuristic for giving an evaluation score to a pixel subset, which in fact builds a small classification model based on it. Here, pixel values in time (all pixel-time pairs) are defined as features (step 2), as was done in [19]. Then, a feature selection heuristic based on information gain (InfoGain, as implemented in [29]) is applied, selecting only the features with positive InfoGain scores (step 3). This usage of InfoGain for ranking features by mutual information with respect to

the class is inspired by [34]. The resulting feature set is cross-validated using linear-kernel SVM (as implemented in [29]) to obtain an evaluation score (cross-validation accuracy of the evaluated set). The produced scores are then ordered in a descending order, and the greedy phase begins.

---

**Algorithm 1** Greedy Improvement of Random Spatial Subspace—*GIRSS* ( $S, C, u, r$ )

---

**Input:** Sample set  $S$ , label set  $C$ , size of random spatial subspace  $u$ , number of random spatial subspaces  $r$

**Output:** Pixel set  $P^* = \{p_1^*, p_2^*, \dots, p_u^*\}$  (top  $u$  spatial subspace representatives).

1. Initialize pixel subsets evaluation scores vector:  $Z[1 : r] \leftarrow 0$
  2. **for**  $i = 1$  to  $r$  **do**:
    - (a) Generate the random permutation vector:  $n^i = \text{permute}(\{1, 2, \dots, n\})$
    - (b) Generate the index vector:  $d^i = \{n_1^i, n_2^i, \dots, n_u^i\}$
    - (c) Select pixel subset (random spatial subspace) indicated by  $d^i$ :  $\tilde{P}^{d^i} \subset P$
    - (d) Save the pixel subset's evaluation score:  

$$Z[i] \leftarrow \text{evaluatePixelSet}(S, C, \tilde{P}^{d^i}, u)$$
  3. Produce sorted indices vector  $I_Z[1 : r] \leftarrow \text{indices}(\text{sort}(Z[1 : r]))$  to contain indices of  $Z[1 : r]$  in the order matching the sorted scores of  $Z[1 : r]$  (highest scores leading).
  4. Initialize the set of pixel subsets  $\Gamma$  with the highest-ranked pixel subset:  

$$\Gamma \leftarrow \{\tilde{P}^{d^{I_Z[1]}}\}$$
  5. Initialize  $z$  with the score of the highest-ranked pixel subset:  $z \leftarrow Z[I_Z[1]]$
  6. **for**  $j = 2$  to  $r$  **do**:
    - (a)  $\Gamma' \leftarrow \Gamma \cup \{\tilde{P}^{d^{I_Z[j]}}\}$
    - (b)  $P' \leftarrow \text{extractHighestRankedPixels}(S, C, \Gamma', |\Gamma'|, Z[1 : r])$
    - (c)  $z' \leftarrow \text{evaluatePixelSet}(S, C, P', u)$
    - (d) if  $z' > z$ , update the  $\Gamma$  and its score:  $z \leftarrow z', \Gamma \leftarrow \Gamma'$ .
  7.  $P^* \leftarrow \text{extractHighestRankedPixels}(S, C, \Gamma, u, Z[1 : r])$
- 

The goal of the greedy phase is to produce one desirable pixel set based on the randomly generated sets. It is desirable to consider all possible subsets of any combinations of the randomly generated sets. However, since we consider large-scale problems, this will lead to exponential time complexity. Thus, we propose a greedy approach. The evaluation of a pixel depends on the value of the subset it belongs to and its own contribution, which leads to several greedy-based decisions as described below.

First, we consider the randomly generated pixel sets according to their rank. This is motivated by the preference given to pixels that belong to highly ranked sets. We maintain a set  $\Gamma$  of pixel subsets, of which the desirable pixel set can be derived at any time. Initially,  $\Gamma$  is initialized with the highest-ranked pixel subset (along with its evaluation score). In each iteration over the ranked pixel subsets list, the next subset in the list joins  $\Gamma$ . A set of pixels of size  $u$  is then extracted from  $\Gamma$  (refer to Algorithm 3), and evaluated (again, using Algorithm 2). If the resulting evaluation score is higher than the existing evaluation score of  $\Gamma$ , the current pixel subset remains in  $\Gamma$  (this is the greedy step). Otherwise, it is discarded.

Discarding sets that are not capable of producing good feature sets is motivated by the importance of considering the value of single pixels. Note, however, that a given pixel may belong to more than one subset, and thus highly valued pixels belonging to a greedily discarded subset may still belong to the final generated set. This way  $\Gamma$  maintains only those

pixel subsets along the way which are capable to produce a highly evaluated pixel subset (whose size is equal to the size of any of the pixel subsets in  $\Gamma$ ), in any requested time. Finally, when the iteration over the pixel subsets is over, the desirable set of pixels is extracted from  $\Gamma$  to serve as the pixel selection.

The decision to use a linear-kernel SVM was based on empirical evaluation. We had experimented with a selection of classifiers other than SVM, and with SVM classifiers with various kernels—including Radial Basis Function (RBF) and polynomial kernels with varying exponents. We finally decided on the linear-kernel SVM since it provided good accuracy with low run-times.

---

**Algorithm 2** Pixel Set Evaluation—*evaluatePixelSet* ( $S, C, P', u$ )

---

**Input:** Sample set  $S$ , label set  $C$ , sorted pixel set  $P'$ , size of the random spatial subspace  $u$ .

**Output:** Evaluation score  $z$ .

1.  $P'' \leftarrow p_i \in P' \mid i \in \{1, \dots, \min(u, |P'|)\}$ .
  2. Extract feature-set:  $F = \{p_j^t \mid t \in \{1, \dots, m\}, \forall p_j \in P''\}$  over the sample set  $S$ .
  3. Perform feature-selection in  $F$  to obtain reduced feature set  $F'$ , using *InfoGain* ( $S, F, C$ ), producing scores:  $IG(p_j^t), \forall p_j^t \in F$ . Select only features having  $IG(p_j^t) > 0$ .
  4.  $z \leftarrow$  Accuracy score of a 10-fold cross-validation of  $F'$  applied on  $S$  using *SVM* ( $S, F', C$ ).
- 

The extraction of the highest-ranked pixels set from  $\Gamma$  (Algorithm 3), at any stage of *GIRSS*, is done as follows: each individual pixel subset in  $\Gamma$  is turned into a feature set, where pixel values in time are defined as features (step 2a). An InfoGain-based feature selection is applied on this feature set, and the InfoGain scores for each feature are taken (step 2b). The score for each individual pixel is calculated by averaging (along the number of pixel instances) the weighted averages of InfoGain scores (along the pixel’s time course in each of the feature sets) (step 2c). The evaluation score of each pixel subset in  $\Gamma$  is used as the weight for computing the grand-average, effectively giving higher weight to pixels and features stemmed from highly evaluated pixel subsets.

---

**Algorithm 3** Highest-Ranked Pixels Extraction—*extractHighestRankedPixels* ( $S, C, \Gamma, u, Z[1 : r]$ )

---

**Input:** Sample set  $S$ , label set  $C$ , set of pixel subsets  $\Gamma = \{P_1, P_2, \dots\}$ , size of the random spatial subspace  $u$ , pixel subsets score vector  $Z[1 : r]$ .

**Output:** Top  $u$  ranked pixels  $p_{I_\rho[l]} \in P, l \in \{1, \dots, u\}$ .

1. Initialize pixels score vector:  $\rho[1 : n] \leftarrow 0$  and pixels instances vector:  $t[1 : n] \leftarrow 0$ .
  2. **for**  $\forall P_i \in \Gamma$  **do**:
    - (a) Extract feature-set:  $F = \{p_j^t \mid t \in \{1, \dots, m\}, \forall p_j \in P_i\}$  over the sample set  $S$ .
    - (b) Rank features in  $F$  using *InfoGain* ( $S, F, C$ ) producing scores:  $IG(p_j^t), \forall p_j^t \in F$ .
    - (c) **for**  $\forall p_j \in P_i$  **do**:  $\rho[j] = \frac{\rho[j] \cdot t[j] + Z[i] \cdot \frac{\sum_{t=1}^m IG(p_j^t)}{m}}{t[j] + 1}, t[j] = t[j] + 1$ .
  3. Produce sorted pixel indices vector  $I_\rho[1 : n] \leftarrow indices(sort(\rho[1 : n]))$  to contain indices of  $\rho[1 : n]$  in the order matching the sorted scores in  $\rho[1 : n]$  (highest scores leading).
-

### 2.2.2 Complexity analysis

The time and space complexity of Algorithm 2, the *evaluatePixelSubset*, is linear in the number of all combinations of pixel-in-time pairs per sample—i.e., in the number of basically defined features. Therefore, the first two steps in the algorithm have a cost of  $O(kmu)$ , and so is the InfoGain feature selection step (which is linear in the number of features). An efficient, state-of-the-art SVM implementation is linear in the number of samples (inherently, features), so our cross-validation using the SVM also has a bound of  $O(kmu)$ . Thus, the overall time and space complexity of Algorithm 2 is  $O(kmu)$ .

In *extractHighestRankedPixels*, Algorithm 3, the size of  $\Gamma$  is bounded by  $r$ , so the algorithm's single loop is performed at most  $r$  times. The cost of each iteration of the loop is  $O(kmu)$ , for the reasons stated earlier in the analysis of Algorithm 2 (feature set extraction, InfoGain ranking and values averaging). Besides the loop, we initialize and sort a vector of length  $n$ . This results in the overall time complexity of the algorithm of  $O(n \log n + rkm)$ . The space complexity is different: we only need  $O(n)$  storage for the scores vector, and an additional  $O(kmu)$  space for a single loop iteration. Therefore, the space complexity of the algorithm is  $O(n + km)$ .

The time complexity of the main Algorithm 1, the *GIRSS*, is analyzed as follows. Each iteration of the first loop, repeated  $r$  times, has a cost of  $O(n)$  added to the cost of Algorithm 2, the *evaluatePixelSubset*, resulting in an iteration cost of  $O(n + km)$ . Therefore, the loop's total cost is  $O(r(n + km))$ . Each iteration of the second loop, which is also repeated  $r$  times, has a cost of Algorithm 3, the *extractHighestRankedPixels*, added to the cost of *evaluatePixelSubset*—resulting in the total cost of  $O(r(n \log n + rkm))$ . Lastly, we sort a vector of length  $r$  once, a step that costs  $O(r \log r)$ . Overall, the time complexity of *GIRSS* is  $O(r(\log r + n \log n + rkm))$ .

As for the space complexity of *GIRSS*, it is  $O(r + n)$  for the evaluation scores vector and random permutation vectors generation,  $O(n + km)$  for the calls to Algorithm 3, and  $O(kmur)$  for the maintenance of the data structures during the executions of the loops inside the *GIRSS*. Altogether, this results in the space complexity of  $O(n + rkm)$ .

We believe that there is space for further complexity reduction. However, we were satisfied with the performance of the presented version during the experimental evaluation, as detailed in Sect. 3, so the current implementation was retained.

## 2.3 Feature extraction

Methods described here are applied on the pixel selection results of the first phase (Sect. 2.2). We present two alternative feature extraction approaches in order to cope with variability evident in different spatio-temporal data-sets. Even when the data-sets originate from the same domain, they can bear different spatial characteristics, expressed in the noise level and the resolution of the signal collected during the data-set construction. The alternatives provided here are each aimed at a different data-sets sector.

### 2.3.1 Features as pixels in time—PIT

One straightforward approach for extracting a feature set  $F$  from a given pixel set  $P^* = \{p_1^*, p_2^*, \dots, p_u^*\}$  over the sample set  $S$  is to define it as all pixel-in-time combinations  $F = \{p_j^t \mid t \in \{1, \dots, m\}, \forall p_j \in P^*\}$ , yielding  $u \cdot m$  features. We call this approach *PIT* (Pixels In Time), and use it in Sect. 2.2 for ranking pixel subsets and feature sets. While for

simpler classification tasks, this is satisfactory—fast, simple and effective (Sect. 3), a method described next is suggested for more complex tasks.

### 2.3.2 Spatial averaging of pixel groups based on inter-pixel correlation

The motivation for this method is to overcome the negative effects of a possibly noisy data by performing a spatial-level averaging of pixels that share a common nature. This requires that the trends of their change along the time course will have similar characteristics. Two questions raised here are

- How to measure similarity between the pixels?
- How to choose “similar” pixels in space, designated for averaging?

The way we measure similarity is by employing Pearson’s product moment coefficient [9] between pairs of pixels. This method is simple, suitable with the type of data we have and was successfully used for calculation of correlation scores in multivariate time series (where correlation is employed for discrimination of target classes [23, 32]).

As a reminder, the Pearson correlation coefficient between vectors  $a$  and  $b$  of length  $n$  is defined as follows:

$$\rho = \frac{\sum_{i=1}^n (a_i - \mu_a)(b_i - \mu_b)}{(n - 1)\sigma_a\sigma_b}$$

where  $\mu_a$  and  $\mu_b$  are the respective means of vectors  $a$  and  $b$ , and  $\sigma_a$  and  $\sigma_b$  are their respective standard deviations. The coefficient  $\rho$  represents the linear correlation between the variables. It ranges from  $-1$  (perfect negative linear correlation) to  $1$  (perfect positive linear correlation), whereas  $0$  indicates no linear correlation (in which case we assume  $a$  and  $b$  are independent).

As for the second question, we perform pixel averaging within groups of “similar” *neighboring* pixels. The reason for this lies in the nature of our data—a non-trivial negative correlation exists between all pixel-pairs correlations and all pixel-pairs distances,<sup>1</sup> showing that higher distances between pixels lead to lower correlations between them. Therefore, choosing neighboring groups of pixels as a whole, having a high inter-group similarity, has the potential to reveal stronger discriminative characteristics—rather than picking individual pixels from the same group.

*The IPCOSA algorithm* We present *IPCOSA* (Inter-Pixel Correlation-based Spatial Averaging; Algorithm 4) below. We show how the neighborhood formation for pixel groups generation is done. This formation is based on a given pixel set, a product from the previous phase introduced in Sect. 2.2—we refer to this set as “the seeds”. First, we calculate a correlation coefficient matrix  $C$  and a distances matrix  $D$  between all pixel pairs (step 3); these matrices are symmetric (only one triangle above or below the diagonal is essential). Then we define the set of pixel subsets  $\Delta$ , which will eventually hold the groups of neighboring pixels that share a similar nature. Next, we employ a graded group formation phase (step 5), where the correlation strength dictates the group formation order: groups having the strongest inter-similarity are generated first, ensuring that the eventually formed groups exploit the similarity property to its full extent (only positive correlation coefficient thresholds are used).<sup>2</sup>

<sup>1</sup> During the experimental evaluation of all VSDI data-sets (Sect. 3), the coefficient between all pixel-pairs correlations and all pixel-pairs distances was within the range of  $\approx -0.45 \pm 0.5$ .

<sup>2</sup> Our choice of  $\tau$  was 0.05 in all our experiments.



The group formation is subject to the following guidelines: a group of pixels must contain at least one seed within it to base the group on. Once chosen, the seed's proximate neighbors' correlation scores are examined. Neighbors with scores that fit the graded correlation threshold join the seed's group. Recursively, the correlation scores of the neighbors of each of the newly joined group members are tested, and additional pixels conforming to the correlation and the proximity requirements join the group. Eventually, a group stops expanding once none of the group members' neighbors fits the requirements. At this step, a formed group joins  $\Delta$ , and its members are no longer available for formation of new groups. A group may consist of a sole seed (step 6). At the end of the group formation phase,  $\Delta$  contains groups of neighboring pixels, each based on one or more seeds. Some groups have stronger inter-similarity than the others, but due to our graded group formation phase, even the weaker groups are generally based on non-negligible positive correlation scores.<sup>3</sup>

At the final phase of our algorithm, the feature extraction is based on  $\Delta$ 's pixel groups: pixel values at each of the points in time are averaged along their spatial dimension—across all pixels within each of the groups of  $\Delta$ .<sup>4</sup> The resulting features represent the average-in-time of similar pixels, as opposed to the pixel-in-time approach presented in Sect. 2.3.1. For seeds pixel set of size  $u$ , there will be at most  $u \cdot m$  features (number of formed groups will not exceed the number of seeds, as each group must contain at least one seed).

*Complexity analysis* For the analysis of time complexity, the *IPCOSA* has three major parts: the initialization and the calculation of the correlation and the distance matrices, the groups formation and the maintenance of  $\Delta$ , and the spatial average calculations. The first part has a cost of  $O((knm)^2)$ . The second part has a complexity of  $O(n)$ , thanks to the fact that the groups formation in grids is done by exploring the finite and bounded set of only the closest neighbors of each of the pixels, where each pixel relation evaluation is done only once—resulting in the number of such evaluations being linear in the number of pixels. Since  $\Delta$  contains at most  $u$  groups (due to the property of at least one seed per group), the cost of the third part is simply  $O(um)$ ,  $u \ll n$ . Overall, the time complexity of *IPCOSA* is  $O((knm)^2)$ .

The space complexity of *IPCOSA* is, therefore,  $O(n^2)$  for building the matrices in the first part,  $O(n)$  for storing the group formation information in  $\Delta$  during the execution of the second part, and  $O(um)$  for the feature generation step in the third part. Altogether, the space complexity is  $O(n^2 + um)$ .

However, it is easy to notice that the version of the algorithm presented here is suboptimal, mainly for the purpose of clarity—there is no need for the initial calculation, nor the storage, of the correlation and the distance matrices. Their values can be computed on demand, while the calculations spread pattern in the grid-formatted space guarantees that only  $O(n)$  calculations will be performed (during the second part of the algorithm). Thus, the actual optimal implementation of *IPCOSA* has complexities of  $O(knm)$  for time and  $O(n + um)$  for space.

## 2.4 Feature selection

To further improve model quality and reduce the feature-space dimensionality, feature selection is applied on the extracted features. InfoGain-based feature selection [29] is applied on the given feature set  $F$  of the samples set  $S$ , producing scores:  $IG(S, f)$ ,  $\forall f \in F$ . Then, only the features with positive InfoGain scores:  $IG(S, f) > 0$  are selected.

<sup>3</sup> As our empirical evaluation of VSDI data shows (Sect. 3), in most cases the weakest formed groups are based on a correlation coefficient of at least 0.4.

<sup>4</sup> Various seeds-based spatial averaging methods were tested during our empirical evaluation of VSDI data, in order to choose the most appropriate method. Please refer to Sect. 5.3 for additional details.

**Algorithm 4** Inter-Pixel Correlation-based Spatial Averaging—*IPCOSA* ( $S, C, P^*, \tau$ )

**Input:** Sample set  $S$ , label set  $C$ , seeds pixel set  $P^*$  of size  $u$ , correlation threshold step  $\tau \in [0, 1]$ .

**Output:** Feature set  $F^*$  over the sample set  $S$ .

1. Set neighboring distance threshold  $\mu$  (e.g. for spatially grid-formatted domains:  $\mu = \sqrt{2}$ ).  $p_1$  and  $p_2$  are neighbors iff  $distance(coords(p_1), coords(p_2)) \leq \mu$ .
2. Initialize correlation coefficient matrix:  $C = 0_{n \times n}$  and distance matrix:  $D = 0_{n \times n}$  (symmetric).
3. **for**  $\forall p_i \in P$  **do**:
  - (a) Vectorize all  $\overline{p_i}$  values of  $p_i$  over the sample set  $S = \{s_1, s_2, \dots, s_k\}$  to produce super-vector of length  $m \cdot k$  with all of concatenated  $\overline{p_i}$  values:
 
$$q_i = \left\langle \left\langle v_1^i, \dots, v_m^i \right\rangle_{s_1} \dots \left\langle v_1^i, \dots, v_m^i \right\rangle_{s_k} \right\rangle$$
  - (b) **for**  $\forall p_j \in P, p_i \neq p_j$  **do** (for every pair  $p_i, p_j$ ):
    - i Vectorize all  $\overline{p_j}$  values of  $p_j$  over the sample set  $S = \{s_1, s_2, \dots, s_k\}$  to produce super-vector of length  $m \cdot k$  with all of concatenated  $\overline{p_j}$  values:
 
$$q_j = \left\langle \left\langle v_1^j, \dots, v_m^j \right\rangle_{s_1} \dots \left\langle v_1^j, \dots, v_m^j \right\rangle_{s_k} \right\rangle$$
    - ii Compute correlation coefficient:  $C_{(i,j)} = correlation(q_i, q_j)$ .
    - iii Compute distance:  $D_{(i,j)} = distance(coords(p_i), coords(p_j))$ .
4. Initialize  $\Delta$ , the set of pixel subsets:  $\Delta \leftarrow \emptyset$ , and  $R$ , the retaining pixel set:  $R \leftarrow P$ .
5. **for**  $r \in \{1, 1 - \tau, 1 - 2\tau, \dots, \tau\}$  **do**:
  - (a) **while**  $\exists p \in R$  s.t.  $p \in P^*$  ( $p$  is a seed) and  $\exists \hat{p} \in R$  s.t.  $C_{(\hat{p},p)} \geq r - \tau$  and  $D_{(\hat{p},p)} \leq \mu$ :
    - i Initialize  $G$ , pixel subset group,  $G \leftarrow \{p\}$ .
    - ii  $R \leftarrow R \setminus \{p\}$
    - iii **while**  $\exists p' \in R$  and  $\exists \tilde{p} \in G$  s.t.  $C_{(\tilde{p},p')} \geq r - \tau$  and  $D_{(\tilde{p},p')} \leq \mu$ :
      - A  $G \leftarrow G \cup \{p'\}$
      - B  $R \leftarrow R \setminus \{p'\}$
    - iv  $\Delta \leftarrow \Delta \cup \{G\}$
6. **for**  $\forall p \in R$  s.t.  $p \in P^*$  ( $p$  is a remaining seed in  $R$ ) **do**:
  - (a)  $R \leftarrow R \setminus \{p\}, G \leftarrow \{p\}, \Delta \leftarrow \Delta \cup \{G\}$
7. Initialize feature-set  $F^*$  over the sample set  $S, F^* \leftarrow \emptyset$ .
8. **for**  $t = 1$  to  $m$  and  $\forall G \in \Delta$  **do**:
  - (a) Define  $f^t$ —the average of values of all pixels in  $G$  at time  $t$ :
 
$$f^t = \frac{\sum_{i=1}^{|G|} v_t^i}{|G|}, s.t. \overline{p_i} = \langle v_1^i, \dots, v_m^i \rangle, v_t^i \in \mathbb{R}, t \in \{1, \dots, m\}, \forall p_i \in G$$
  - (b)  $F^* \leftarrow F^* \cup \{f^t\}$

The motivation: the features produced in Sect. 2.3 are based on pixel selection from Sect. 2.2, where the whole time-spectrum of pixels or pixel groups is preserved. However, points along the time course exist, during which the spatial discriminative nature is not realized (e.g. long before the onset of the signal in VSDI). Not only that these points in time are ineffective for the emphasis of the spatial characteristics, but they sometimes obscure their discriminating potential. InfoGain filtering drops those unwanted features with negligible scores, whose contribution is neutral or negative.

2.5 Example

In this section, we provide a concrete example of our algorithms, applied to a real Gabors data-set—which will be introduced in Sect. 3.1. This example is a part of a single

experimental trial, among the 10 trials performed on the Gabors data-set. The model evaluation described here uses the *GIRSS* pixel selection technique in combination with each of the two feature extraction techniques—the *PIT* and the *IPCOSA*. The particular single trial addressed here was comprised of 10 cross-validated folds and has yielded an average (between the folds) accuracy of 79.74% for  $\{GIRSS, PIT\}$  and of 81.7% for  $\{GIRSS, IPCOSA\}$ . Our description focuses on a single fold among the 10 folds—specifically, fold 4.

### 2.5.1 Phase 1—Pixel selection using the *GIRSS*

#### Input

- *S*: 138 samples are used as the training set in fold 4 (out of the total 153 Gabors samples, where 15 samples are reserved for the test set). Each line in a sample file represents a pixel. There is a rectangular grid of  $100 \times 100$  pixels, thus 10,000 pixel lines in each file. A pixel is indexed by the line’s number (e.g., pixel 101 is represented by line 101). Each line contains 51 values measured for its specific pixel—one value for each of the 51 consecutive time-points in which the value was measured.
- *C*: 6 class labels (0, 1, 2, 3, 4, 5). Each of the 138 training samples from *S* is labeled with one of the 6 labels of the Gabors data-set.
- *u*: 100—The size of a random spatial subspace, i.e. the size of a single pixel subset (in each subset, there will be 100 pixels). This parameter is provided manually.
- *r*: 150—The number of random spatial subspaces, i.e. the number of pixel subsets generated by the algorithm. This parameter is also provided manually.

#### Flow

1. Initialization of the evaluation scores vector  $Z[1 : r]$  with values of 0.
2. Generation of 150 pixel subsets ( $subset_1, \dots, subset_{150}$ ). Each subset represents 100 randomly selected pixels. For example,  $subset_1$  contains the following 100 pixel indices: 70, 128, 200, 359, 415, 422, 483, 489, 619,  $\dots$ , 9, 664, 9, 681, 9, 845.
3. Each of the 150 pixel subsets is being evaluated, and given an evaluation score, using Algorithm 2. This algorithm defines the feature set of each subset as a pixel-in-time combination, using each pixel’s complete time range. For instance, the feature set of  $subset_1$  contains the following features:

$$\begin{array}{ccccccc}
 pixel_{70}timepoint_1 & pixel_{70}timepoint_2 & \dots & pixel_{70}timepoint_{51} \\
 pixel_{128}timepoint_1 & pixel_{128}timepoint_2 & \dots & pixel_{128}timepoint_{51} \\
 \vdots & \vdots & \vdots & \vdots \\
 pixel_{9845}timepoint_1 & pixel_{9845}timepoint_2 & \dots & pixel_{9845}timepoint_{51}
 \end{array}$$

Measured values of pixel 70 at time-point 1, pixel 70 at time-point 2, and so on—until pixel 9845 at time-point 51— are picked for each of the 138 training samples. After the features of the composed training set are screened via the application of the Info-Gain filter, only the features with positive scores remain in the feature set. In the case of  $subset_1$ , only 384 pixel-in-time combinations (out of  $100 \cdot 51 = 5,100$  potential features) remain. Therefore, the partial list of these features is as follows (ordered by the evaluation score—the right-most column):

*pixel*<sub>6948</sub>*timepoint*<sub>33</sub> 0.66  
*pixel*<sub>6847</sub>*timepoint*<sub>39</sub> 0.586  
*pixel*<sub>6847</sub>*timepoint*<sub>34</sub> 0.578  
*pixel*<sub>7640</sub>*timepoint*<sub>38</sub> 0.563  
 ⋮ ⋮  
*pixel*<sub>2659</sub>*timepoint*<sub>43</sub> 0.138  
*pixel*<sub>5426</sub>*timepoint*<sub>31</sub> 0.131

The samples defined in this way are 10-fold cross-validated using the SVM. The accuracy score of the cross-validation is the output of Algorithm 2. In case of *subset*<sub>1</sub>, the accuracy score is 69.57. Evaluation scores vector  $Z[1 : r]$ , therefore, contains the value of 69.57 at its first index (corresponding to *subset*<sub>1</sub>), the value of 72.46 at its second index (corresponding to *subset*<sub>2</sub>), and so on, until the last subset—*subset*<sub>150</sub>, the value of which is 72.46.

- Having each of the pixel subsets given a score, the scores are sorted and stored in the indices vector  $I_Z[1 : r]$ . In our example, the partial list of the produced evaluation scores is (ordered by the evaluation score—the right-most column):

*subset*<sub>39</sub> 79  
*subset*<sub>124</sub> 78.26  
*subset*<sub>64</sub> 78.26  
*subset*<sub>35</sub> 77.54  
 ⋮ ⋮  
*subset*<sub>1</sub> 69.57  
 ⋮ ⋮  
*subset*<sub>90</sub> 58.7  
*subset*<sub>87</sub> 57.25  
*subset*<sub>128</sub> 51.45

Therefore, the vector  $I_Z[1 : r]$  contains: 39, 124, 64, 35, . . . , 1, . . . , 90, 87, 128.

- $\Gamma$  is initialized with the highest-ranked pixel subset, *subset*<sub>39</sub>, and  $z$  is initialized with its score of 79.
- In this loop, the first thing that happens is that *subset*<sub>124</sub> joins  $\Gamma$ , which now holds the pixel subset indices of {39, 124}. Highest-ranked pixels are extracted from  $\Gamma$  using Algorithm 3. Since the weight of *subset*<sub>39</sub> is 0.79, and the weight of *subset*<sub>124</sub> is 0.78, an example of (positive only) weighted InfoGain scores for the features created during the run of Algorithm 3 is (ordered by the weighted InfoGain score—the right-most column):

*pixel*<sub>7148</sub>*timepoint*<sub>37</sub> 0.58  
*pixel*<sub>7047</sub>*timepoint*<sub>30</sub> 0.48  
*pixel*<sub>7148</sub>*timepoint*<sub>39</sub> 0.47  
 ⋮ ⋮  
*pixel*<sub>3741</sub>*timepoint*<sub>47</sub> 0.19

The subset eventually produced by Algorithm 3 contains the following pixel indices in  $I_\rho[1 : n]$  (highest-ranked pixels first): 7148, 7542, 7737, 7047, 7453, 7843, 6945, 6930, 3328, . . . , 2463, with the corresponding scores of each pixel in the list being: 0.18, 0.17, 0.15, 0.14, 0.13, 0.11, 0.106, 0.105, 0.104, . . . , 0.002. The first 100 pixels from this set are evaluated by Algorithm 2, producing the evaluation score of 79.71. Since this score

is higher than the currently leading score of 79,  $subset_{124}$  remains in  $\Gamma$ , and  $z$  is updated with the new score.

In the next iteration,  $subset_{64}$  joins  $\Gamma$ , which now holds the pixel subset indices of {39, 124, 64}. Algorithm 3 extracts the following ranked pixel indices into  $I_\rho[1 : n]$ : 7148, 7542, 7737, 6851, 7447, 6848, 7047, 7453, 7843, . . . , 2463. The evaluation score given by Algorithm 3 using the first 100 pixels is 81.88, thus  $z$  is updated again, and  $\Gamma$  is retained as is.

At the third iteration,  $subset_{35}$  joins  $\Gamma$ , the pixel subset indices in which become {39, 124, 64, 35}. However, the evaluation score produced this time is 79.71—lower than  $z$ . Following this event,  $subset_{35}$  is discarded and is not retained in  $\Gamma$ . Similarly, the next subset in line,  $subset_{74}$ , is being discarded too.

In a similar way, all 150 subsets are joining  $\Gamma$ , tested, and eventually retained or discarded. Right before the last iteration,  $\Gamma$  holds the pixel subset indices of {39, 124, 64, 134, 11, 47, 59, 48, 73, 29, 90}, and  $z$  is 89.13. The addition of the last subset,  $subset_{128}$ , to  $\Gamma$ , yields a result lower than  $z$ , thus the final  $\Gamma$  remains as it was prior to the last loop iteration.

7. At this stage, Algorithm 3 is applied on the final  $\Gamma$ , and the following highest-ranked 100 pixels are extracted from it: 7148, 7542, 7345, 7737, 7139, 7240, 7538, 6851, 7738, . . . , 5226. These pixels are the seeds set— $P^*$ , the output of  $GIRSS$ .

*Output*  $P^*$ , as detailed above.

### 2.5.2 Phase 2 (alternative 1)—Feature extraction using the PIT

In *PIT*, the features are defined as every pixel-in-time combination on the given pixel set. Using  $P^*$  as the input, the produced feature set contains the following features:

$$\begin{array}{cccc}
 pixel_{7148}timepoint_1 & pixel_{7148}timepoint_2 & \dots & pixel_{7148}timepoint_{51} \\
 pixel_{7542}timepoint_1 & pixel_{7542}timepoint_2 & \dots & pixel_{7542}timepoint_{51} \\
 \vdots & \vdots & \vdots & \vdots \\
 pixel_{5226}timepoint_1 & pixel_{5226}timepoint_2 & \dots & pixel_{5226}timepoint_{51}
 \end{array}$$

### 2.5.3 Phase 3 (alternative 1)—Selection of PIT-extracted features

After applying the InfoGain filter to the features of the composed training set, 2251 pixel-in-time combinations (out of  $100 \cdot 51 = 5,100$  features) remain. Therefore, the partial list of these features is as follows (ordered by the evaluation score—the right-most column):

$$\begin{array}{cc}
 pixel_{7148}timepoint_{37} & 0.742 \\
 pixel_{7542}timepoint_{34} & 0.689 \\
 pixel_{7052}timepoint_{34} & 0.671 \\
 \vdots & \vdots \\
 pixel_{3265}timepoint_{13} & 0.128
 \end{array}$$

Training a classification model using the resulted feature set and applying the model to the 15 reserved test samples results in the final accuracy of 86.67% for the combination of {*GIRSS*, *PIT*} in fold 4 of this trial.

### 2.5.4 Phase 2 (alternative 2)—Feature extraction using the IPCOSA

#### Input

- $S, C$ : Same as the  $S$  and  $C$  input parameters for  $GIRSS$  (phase 1).
- $P^*$ : The seeds pixel set produced by  $GIRSS$  in phase 1, pixel indices of which are: 7148, 7542, 7345, 7737, 7139, 7240, 7538, 6851, 7738,  $\dots$ , 5226.
- $\tau$ : 0.05—The correlation threshold step for the graded formation of groups. This parameter is provided manually.

#### Flow

1. Initialization of  $\mu$  as  $\sqrt{2}$  (the pixels are spread over a rectangular grid).
2. Initialization and population of the correlation coefficient matrix  $C$  and the distance matrix  $D$ . For example:  $D(1, 2) = 1, \dots, D(1, 5) = 4, \dots, D(1, 100) = 99, D(1, 101) = 1, D(1, 102) = \sqrt{2}, \dots, D(1, 9901) = 99, \dots, D(9999, 10000) = 1$ . For brevity, we skip concrete examples of correlation coefficients in  $C$ .
3. Initialization of  $\Delta$  with  $\emptyset$ , and of the retaining set  $R$  with  $P$  (the complete 10,000 pixels set).

4. In this loop, the first thing to happen is the exploration of the range of correlation (0.95 – 1]. Every seed from  $P^*$  with a correlation to a neighboring seed in this range forms a pixel subset group  $G$ —with neighboring pixels added recursively to each of the groups of the corresponding seed. In our example, only one group is formed in the first iteration:  $group_{0.95_1} : \{3224, 3325\}$ —the seed of which is 3325. This single produced group joins  $\Delta$ , which now contains:  $\{group_{0.95_1}\}$ . Pixels from this group are removed from the retaining set  $R$ .

Next, the range of correlation (0.9–0.95] is explored. As a result, the following two groups are produced:  $group_{0.9_1} : \{2842, 2843, 2844, 2943, 2944\}$ —the seed of which is 2943, and  $group_{0.9_2} : \{3130, 3131\}$ —the seed of which is 3131. Again, the two new groups join  $\Delta$ , which now contains:  $\{group_{0.95_1}, group_{0.9_1}, group_{0.9_2}\}$ . Pixels from the newly joined groups are removed from the retaining set  $R$ .

In turn, the correlation range of (0.85–0.9] produces 4 new groups:  $group_{0.85_1} : \{2946, 2947, 2948, 2949, 2950, 2951, 2952, 3046, 3047, 3048, 3049, 3050, 3051, 3052\}$ , the two seeds of which are 3047 and 3050;  $group_{0.85_2} : \{3227, 3228, 3229, 3327, 3328\}$ —the seed of which is 3328,  $group_{0.85_3} : \{2651, 2652, 2653\}$ —the seed of which is 2652, and  $group_{0.85_4} : \{3230, 3231\}$ —the seed of which is 3231.

Finally, at the end of the loop,  $\Delta$  contains 78 groups—with the last group, having the weakest inter-correlation, being  $group_{0.45_1} : \{5614, 5615, 5616\}$ —the seed of which is 5615.

5. This phase, responsible for generating pixel groups based on the remaining sole seeds, produces 5 new groups—each based on one of the remaining seeds. These groups ultimately join the set  $\Delta$ : *sole seed group*<sub>1</sub> : ({2881}), *sole seed group*<sub>2</sub> : ({3547}), *sole seed group*<sub>3</sub> : ({6848}), *sole seed group*<sub>4</sub> : ({6377}) and *sole seed group*<sub>5</sub> : ({5769}). Eventually, the set  $\Delta$  contains 83 groups of pixels.
6. Feature set  $F^*$  is initialized with  $\emptyset$ .
7. For every time-point  $t$  from 1 to 51 and for every group  $G$  in  $\Delta$ , feature  $f^t$  in  $F^*$  will represent the average of all pixel values in  $G$  corresponding to time  $t$ . Therefore, in our example, the produced feature set  $F^*$ —containing all group-average-in-time combinations, will hold the following features:

$$\begin{array}{l}
 \text{group0.95}_1 \text{ average, timepoint}_1 \dots \text{group0.95}_1 \text{ average, timepoint}_{51} \\
 \text{group0.9}_1 \text{ average, timepoint}_1 \dots \text{group0.9}_1 \text{ average, timepoint}_{51} \\
 \text{group0.9}_2 \text{ average, timepoint}_1 \dots \text{group0.9}_2 \text{ average, timepoint}_{51} \\
 \vdots \\
 \text{group0.45}_1 \text{ average, timepoint}_1 \dots \text{group0.45}_1 \text{ average, timepoint}_{51}
 \end{array}$$

*Output* The output is  $F^*$ , as detailed above.

### 2.5.5 Phase 3 (alternative 2)—Selection of IPCOSA-extracted features

After filtering via the InfoGain filter, 2043 pixel-in-time combinations (out of  $83 \cdot 51 = 4233$  features) remain. Eventually, the partial list of these features is (ordered by the evaluation score—the right-most column):

$$\begin{array}{ll}
 \text{group0.8}_1 \text{ average, timepoint}_{29} & 1.0242 \\
 \text{group0.8}_4 \text{ average, timepoint}_{33} & 0.9858 \\
 \text{group0.8}_1 \text{ average, timepoint}_{32} & 0.9451 \\
 \vdots & \vdots \\
 \text{group0.5}_2 \text{ average, timepoint}_{24} & 0.1535 \\
 \text{group0.8}_{10} \text{ average, timepoint}_{49} & 0.1465
 \end{array}$$

Training a classification model using the resulted feature set and applying the model to the 15 reserved test samples results in the final accuracy of 100% for the combination of  $\{GIRSS, IPCOSA\}$  in fold 4 of this trial.

## 3 Empirical evaluation using VSDI data

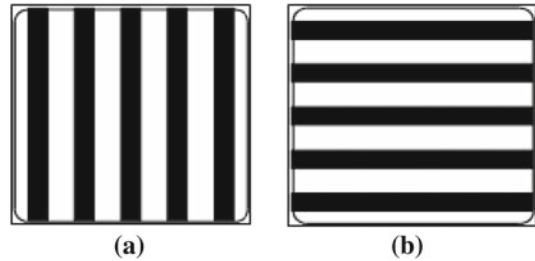
The primary goal in our work is to suggest a combination of effective techniques for obtaining scalable and accurate classification in large-scale spatio-temporal domains. To reach this goal, we demonstrate how our techniques are evaluated in the VSDI domain and applied to VSDI data-sets. The accuracy of the classification is validated by the evaluation of our classification performance. The scalability of our methods is shown by exploring their feasibility from the run-time perspective. This is done by emphasizing the lessons learned from the experience we had with applying approaches similar in nature to the ones reviewed in Sect. 6. Many of these approaches use the most granular values of the sample's space for feature selection and classification, which eventually leads to an extremely high dimensional feature space. Our failure in employing these approaches is compared to the success of showing the feasibility of our methodology. We additionally compare our results to those achieved by a domain expert faced with the same tasks.

### 3.1 Experiment methodology

We first collected three data-sets, each based on a single imaging experiment performed in the visual cortex of one animal and composed from multiple trials. Then, a domain expert was requested to provide an ROI(s) (Region Of Interest) of pixels for each data-set. This was used for the evaluation of the pixel selection technique *GIRSS* presented in Sect. 2.2.

We then constructed classification models using both *GIRSS* and the domain expert (called *Oracle*) in the first phase, in combination with the two feature extraction techniques

**Fig. 2** Stimuli for the Oriented Gratings data-set: **a** *drifted square gratings at vertical orientations*; **b** *drifted square gratings at horizontal orientations*; **c** blank control image (not presented)



in the second phase:  $\{Oracle, GIRSS\} \times \{PIT, IPCOSA\}$ , and with the application of the feature selection (Sect. 2.4) in the third phase. We used several sets of  $u$  and  $r$  parameters for *GIRSS*. Their selection is discussed in Sect. 5.1.

The resulting models were evaluated using a 10-fold cross-validation of the multi-class SVM with linear kernel [29]. Each model's evaluation was performed a number of times (each trial yielding a different random 10-fold division), as specified in the results tables below.

In each experiment, the monkey was shown a set of different visual stimuli, one specific stimulus per trial. Each stimulus presentation was repeated 20–30 times. Neuronal population responses in the visual cortex evoked by the stimulus were recorded using VSDI. The imaged area was divided into a grid of pixels, and population response (summed membrane potentials of all neuronal elements) of each pixel was recorded during the time window of the trial [24]. Each trial in an experiment is a sample in our sample space. A sample consists of all pixels of the recorded area, where a pixel is a time-series of values collected along the time course of the trial.<sup>5</sup> These values are raw data-points—with no averaging across trials, whether in time or space—directly reflecting unprocessed measurement points. Hence, the VSDI decoding we did was performed at a single trial level. Each sample is labeled with a class that represents the appropriate stimulus. The data-sets differ in the number and the type of the presented stimuli, both affecting the complexity of the decoding. Being able to perform successful classification of these data-sets, is being able to “read” what the monkey has seen without seeing it ourselves.

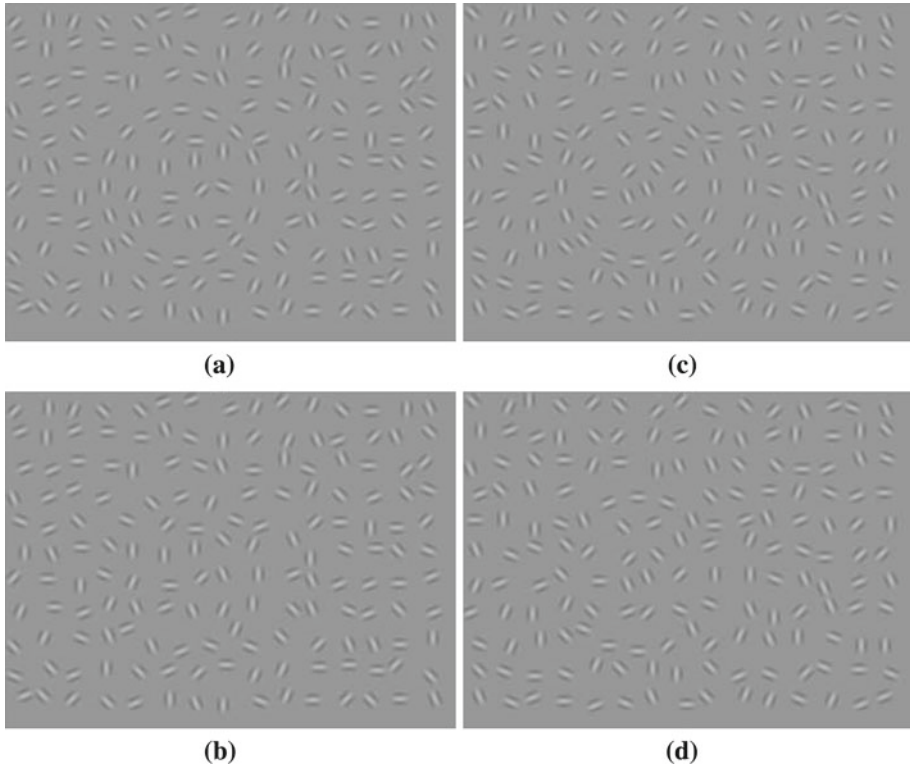
*Data-set 1: ORIENTED GRATINGS (simple).* The monkey was presented with two different drifted square gratings at horizontal and vertical orientations, and a blank control image with no stimulus (Fig. 2). Each of the 293 samples in the data-set had 2162 pixels (a  $46 \times 47$  matrix) along 51 time points. The three classes had almost uniform distribution where the mode class constitutes 34.13% of the population (setting the baseline accuracy using a Zero-R [29]).

*Data-set 2: CONTOURS (moderate).* The monkey was presented with four different Gabor-based Contours in space and a blank control image (see Fig. 3). The four Gabor-based Contours divide into two pairs, where the differences between the classes in each pair are very subtle and hardly noticeable. Each of the 124 samples had 10,000 pixels (a  $100 \times 100$  matrix) along 61 time points. The five classes had almost uniform distribution where the mode class constitutes 23.39% of the population (Zero-R baseline accuracy).

*Data-set 3: GABORS (complex).* The monkey was presented with five different Gabor-based orientations in space and a blank control image (see Fig. 4). Each of the 153 samples had 10,000 pixels (a  $100 \times 100$  matrix) along 51 time points. The

<sup>5</sup> See below for visualization examples of the raw VSDI data.





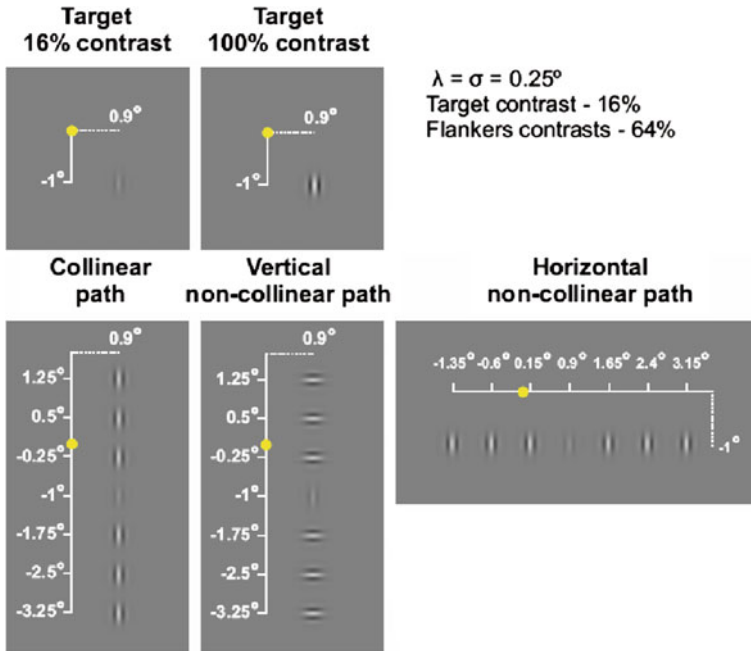
**Fig. 3** Example of stimuli for the Contour data-set: **a** circle 1; **b** masked circle 1; **c** circle 2; **d** masked circle 2; **e** blank control image (not presented)

six classes had almost uniform distribution where the mode class constitutes 18.95% of the population (Zero-R baseline accuracy).

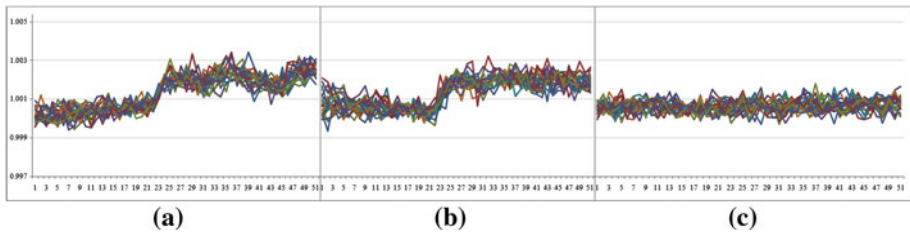
To illustrate the data-sets, we present a fragment of the VSDI raw data. Visualizing data composed of thousands of pixels is not an easy task. Thus, we chose to present only a small number of pixels, taken from a single sample for each of the stimuli types. In Fig. 5 below, we display the values of 1% of 2,162 pixels close to the  $ROI_1$  area of the Oriented Gratings data-set. Each graph in the Figure is based on a single sample out of 293 samples, each of which represents one specific stimulus out of the three possible types. Visual comparison of the different stimuli samples of the Oriented Gratings by a naked eye hints that the classification of this data-set can be a difficult task.

### 3.2 The *Oracle*

We define the *Oracle* pixel selection method, as a best-effort attempt by a human expert to provide a pixel set which, in her professional opinion, has the most potential to successfully discriminate between the different classes of the training samples set. The *Oracle* is asked to manually pick a set of pixels of some size:  $\Omega = \{p_1, p_2, \dots\}$ ,  $\Omega \subseteq P$ , also known as the ROI (Region Of Interest). This set serves as a “gold standard” for comparison with the



**Fig. 4** Stimuli for the Gabor data-set (the numbers and the degrees on the white axes are not part of the stimuli; blank control image not presented). The *circle* represents the fixation point location



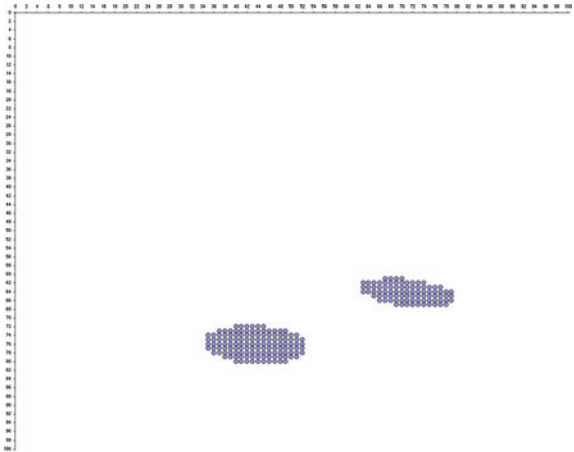
**Fig. 5** Visualization of three samples from the Oriented Gratings data-set, labeled with: **a** class 1; **b** class 2; **c** class 3 (blank control image). Only the values of 1% of the pixels close to the  $ROI_1$  area are displayed for each of the samples along the full interval of sampling

*GIRSS* pixel selection technique. The goal is to build an accurate classification model using the most discriminating pixels.

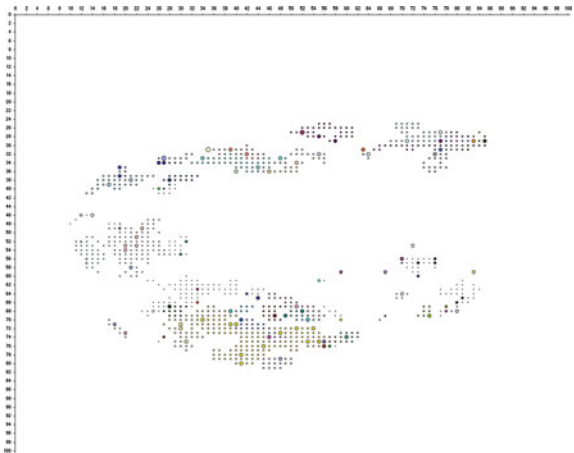
In the experiments, the domain expert was requested to provide an ROI(s) of pixels for each data-set:

- For the Oriented Gratings data-set, we were given a single ROI along the time course of our experiments, the  $ROI_1$ .
- In the case of the Gabors, after the first line of experiments with  $ROI_1$ —the original ROI— we were given  $ROI_2$ , an improved version based on the results of using  $ROI_1$ .
- With the Contours case, three different ROIs of pixels were given in advance, each for individual evaluation by our techniques.

**Fig. 6** Gabors data-set,  $ROI_2(218)$ . The imaged area of pixels is depicted on the grid (all pixels are in V1)



**Fig. 7** Gabors data-set, sample fold result—the imaged area of pixels depicted on the grid. The results of applying *GIRSS* with  $u(100)$  and  $r(125)$  to produce a *seeds* pixel set are shown in *large circles*. Neighborhoods of pixels for averaging are formed around the seeds (*small circles*, having the seeds' colors). The *different sizes* of pixels between the neighborhoods express the strength of the inter-correlation within each neighborhood, compared to the other ones



### 3.3 *GIRSS* vs. human *Oracle* results

In general, there are significant differences between the pixels selected by *GIRSS* and the ROI pixel sets selected by the human domain expert. These differences can be seen by comparing the ROI pixels to the *GIRSS* pixels in the different data-sets.

First, we see differences in the Gabors data-set pixel selection. Fig. 6 shows the best performing *Oracle*'s ROI pixel set,  $ROI_2(218)$ . In contrast, Fig. 7 shows results from a *GIRSS* run.

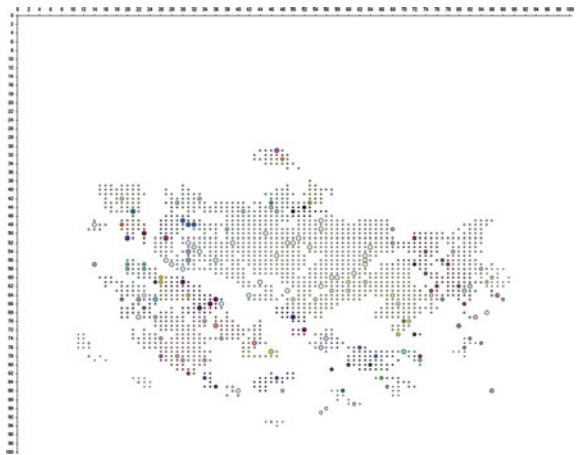
Similar differences between the pixel set selected by the human domain expert, and *GIRSS*, can also be observed in the Contours data-set case. Here, the pixels selected by *GIRSS* cover a wide area spread, again, along multiple sites in the visual cortex.  $ROI_3$ , the best performing Contours ROI is shown in Fig. 8. The results from *GIRSS* are shown in Fig. 9.

With the rather simplistic case of the Oriented Gratings, where the types of the visual stimuli are noticeably different, and the decoding task at hand is far from being complex, the pixel sets produced by *GIRSS* show a wide spread across the cortical area (which is almost

**Fig. 8** Contours data-set,  $ROI_3(155)$ —the best performing *Oracle*'s ROI pixel set. The imaged area of pixels is depicted on the grid



**Fig. 9** Contours data-set, sample fold result—the imaged area of pixels depicted on the grid. The results of applying *GIRSS* with  $u(151)$  and  $r(100)$  to produce a *seeds* pixel set are shown in *large circles*. Neighborhoods of pixels for averaging are formed around the seeds (*small circles*, having the seeds' colors). The *different sizes* of pixels between the neighborhoods express the strength of the inter-correlation within each neighborhood, compared to the other ones

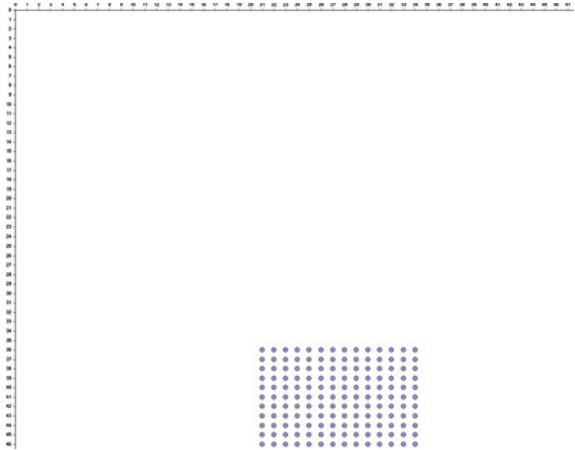


five times smaller than the tested area in the other data-sets). This spread of pixels, being 7.1% of the complete pixel range, seems to almost uniformly cover the whole area. Comparing the accuracies of the pixels selected by *GIRSS* to those provided in the ROI (concentrating in one specific spot), we reveal approximately the same high accuracy results. These findings only strengthen the claim that the Oriented Gratings data-set is a comparatively easy task. The differences are shown in Figs. 10 and 11.

### 3.4 Classification results

The overall results are presented in Tables 1, 2, and 3, in order of increasing difficulty of the learning (as we can see from the Zero-R baseline results). Each table is divided into two sections, which contrast the results for each of the pixel selection methods: The human domain expert (*Oracle*) and the *GIRSS* technique. For each of these, we contrast the average classification accuracy achieved with the two feature extraction techniques, *PIT* and *IPCOSA*. In the tables,  $ROI_x(y)$  denotes an ROI pixel set  $x$  of size  $y$ .

**Fig. 10** Oriented Gratings data-set,  $ROI_1$  (154)—the single Oracle’s ROI pixel set. The imaged area of pixels is depicted on the grid



**Fig. 11** Oriented Gratings data-set, sample fold result—the imaged area of pixels depicted on the grid. The results of applying GIRSS with  $u$ (154) and  $r$ (20) to produce a seeds pixel set are shown in large circles. Neighborhoods of pixels for averaging are formed around the seeds (small circles, having the seeds’ colors). The different sizes of pixels between the neighborhoods express the strength of the inter-correlation within each neighborhood, compared to the other ones



**Table 1** Results in the ORIENTED GRATINGS data-set

		PIT	IPCOSA	$t$ -test $p$
<b>Oracle</b>	$ROI_1$ (154)	<b>95.4 ± 0.4% (10)</b>	79.3 ± 3.2% (10)	0.000000436
<b>GIRSS</b>	$u$ (154), $r$ (20)	<b>94.9 ± 1.3% (10)</b>	88.5 ± 4.0% (10)	0.004

Baseline for accuracy is 34.13%. The last column shows the significance ( $p$ ) of a paired one-tailed  $t$ -test

**Table 2** Results in the CONTOURS data-set

		PIT	IPCOSA	$t$ -test $p$
<b>Oracle</b>	$ROI_1$ (151)	44.9 ± 2.3% (10)	40.2 ± 3.1% (10)	0.0038
	$ROI_2$ (227)	50.6 ± 2.4% (10)	47.6 ± 3.0% (10)	0.0068
	$ROI_3$ (155)	<b>73.3 ± 1.6% (10)</b>	65.7 ± 1.8% (10)	0.000025
<b>GIRSS</b>	$u$ (151), $r$ (100)	71.9 ± 2.8% (10)	72.4 ± 2.7% (10)	0.297
	$u$ (500), $r$ (100)	69.6 ± 2.8% (10)	<b>73.1 ± 2.1% (10)</b>	0.00025

Baseline for accuracy is 23.39%. The last column shows the significance ( $p$ ) of a paired one-tailed  $t$ -test

**Table 3** Results in the GABORS data-set

		PIT	IPCOSA	<i>t</i> -test <i>p</i>
<b>Oracle</b>	$ROI_1$ (104)	55.0 ± 1.5% (10)	57.2 ± 3.3% (10)	0.054
	$ROI_2$ (218)	68.8 ± 1.1% (10)	<b>71.0 ± 2.4% (10)</b>	0.019
<b>GIRSS</b>	$u$ (100), $r$ (150)	79.1 ± 1.7% (10)	<b>81.8 ± 1.4% (10)</b>	0.00015
	$u$ (100), $r$ (125)	78.4 ± 2.1% (10)	<b>81.8 ± 2.3% (10)</b>	0.0006
	$u$ (100), $r$ (100)	79.0 ± 1.8% (10)	80.7 ± 1.7% (10)	0.006

Baseline for accuracy is 18.95%. The last column shows the significance (*p*) of a paired one-tailed *t*-test

The numbers in brackets for *u* (number of pixels in a random pixel subset) and *r* (number of random pixel subsets) are their respective values. The results of the form  $\mu \pm \sigma\%(n)$  have  $\mu$  representing the average accuracy between the trial runs,  $\sigma$  representing the standard deviation and *n* representing the number of trial runs. The entries in bold represent the best accuracies obtained per each pixel selection method of each of the data-sets.

### 4 Classifying hurricane severity

In this section, we present the results from a smaller-scale evaluation of the techniques we presented earlier, on a very different domain than VSDI classification. We apply the techniques to classifying hurricane severity based on weather satellite imagery. Section 4.1 describes the experiment configuration. Section 4.2 discusses the results.

#### 4.1 Hurricane data and experiment configuration

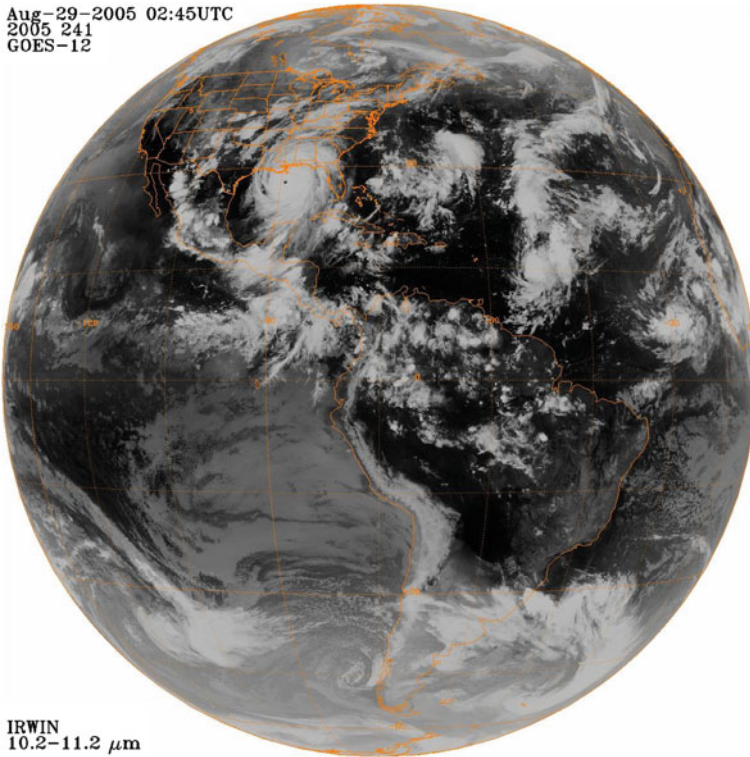
Tropical cyclones are storm systems that originate in tropical areas near the equator, over large bodies of warm water. They form during the *hurricane season*, mostly from June to November. During their life time, tropical cyclones change intensity (strength) and location.

Tropical cyclones in the Atlantic and in the Northeast Pacific are classified into three main groups of ascending severity: tropical depressions, tropical storms, and *hurricanes*. Within the hurricanes group, the Saffir-Simpson Hurricane Scale [21] divides the hurricanes into five categories (denoted simply 1–5) by the intensities of their sustained winds. They produce extremely powerful winds, heavy rain and flood, and when near coastal regions, they are able to cause severe damage. A hurricane typically begins as a low-severity tropical cyclone (i.e., a tropical depression), then moves along its path (called *track*) for a number of days, while changing its strength (respectively, its severity) during its life course, and finally dissipates.

An obviously important task in this domain is to be able to predict the maximum severity that a tropical cyclone will reach, based on initial measurements. As a first step toward this goal, we tackle a more modest task: distinguishing between low- and high- severity cyclones, based on 15 days’ worth of weather satellite images. As we show below, even this modest task is quite challenging, and will serve to evaluate the scope of the techniques we presented.

We utilize satellite images data, taken from the Global ISCCP B1 Browse System (GIBBS) repository<sup>6</sup> [12], a comprehensive weather satellites data resource. The relevant Atlantic area satellite imagery is the “full disk” images (full earth) taken by the Geostationary Operational Environmental Satellites GOES-8 and GOES-12. The infrared images produced by these two satellites are taken from the same angle and are aligned to the same longitude and latitude

<sup>6</sup> The GIBBS repository is available at <http://www.ncdc.noaa.gov/gibbs/>.



**Fig. 12** GOES-12 infrared (*IR*) image of the Atlantic region. Famous hurricane Katrina from the 2005 season can be seen in the South area of the United States, affecting Louisiana, Mississippi, South Florida, etc

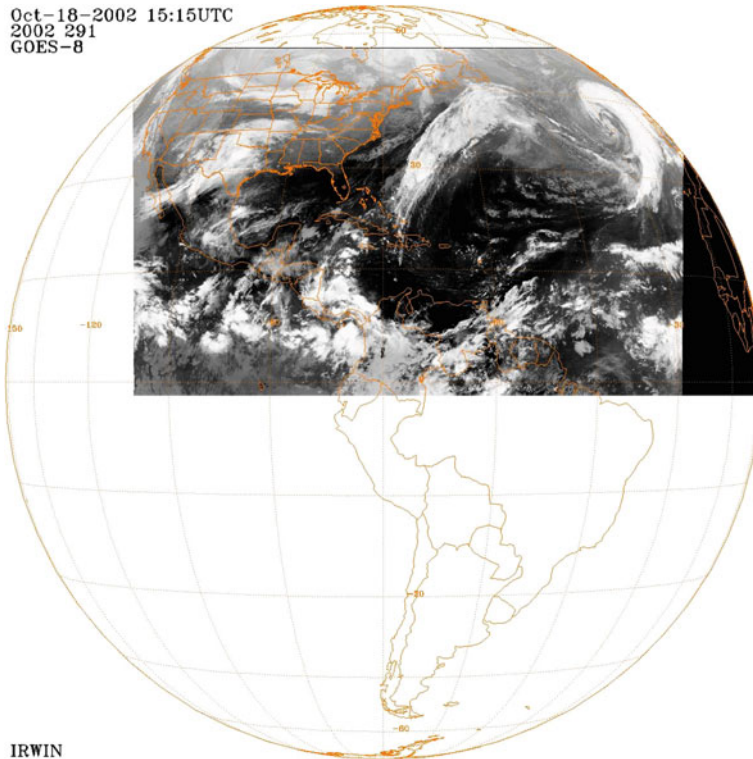
lines. Images are available from the beginning of the hurricane season of 1995 to the current day, taken every 3 h. An example of a full sized image is presented in Fig. 12.

There are a number of challenges involved in processing these images. First, the size of each image is  $1,200 \times 1,200$  pixels, significantly larger than the  $100 \times 100$  VSDI images. In addition, the relative temporal duration is longer (i.e., the number of images in a sequence). The produced data-sets are between 40 and 80 times larger than VSDI.

A second challenge concerns missing, corrupt, or partial data. Unlike the VSDI data available to us, images from specific times may be missing, or incomplete in some fashion. An example of a partial, incomplete image appears in Fig. 13.

A third important challenge concerns the internal coherence and variance of the data itself, both spatially and temporally. In particular, hurricanes appear everywhere within the target regions, take different paths, and last for various durations (a few days to weeks). In comparison, the VSDI classification relied on static regions of activity, of fixed duration. Furthermore, different cyclones of different severity levels sometimes overlap in time and space. In contrast, the VSDI samples are each restricted to a single, isolated type of stimuli (class). To make things worse, the severity of the hurricanes changes unpredictably throughout their lifespan.

Taking these challenges into account, we have built the hurricane data-set as follows. First, each sample was defined to start at the precise start time of an individual hurricane, and to end after a fixed period of 15 days—based on a series of 120 consecutive images (the number



**Fig. 13** An incomplete GOES-8 infrared (*IR*) image of the Atlantic region

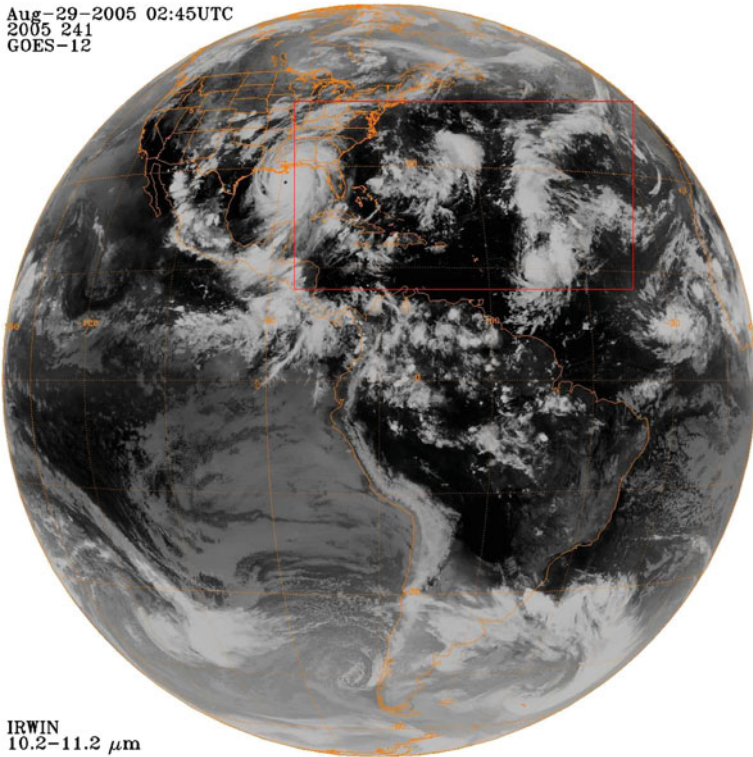
of points in time). The gray-scale values (0–255) of the pixels in the image were taken as the basic values in each image in the sequence.

We manually selected data in which spatio-temporal overlapping between cyclones is minimal (though overlapping still exists in the samples; it cannot be completely discarded). From each “full disk” image, an area of interest of size  $540 \times 300$  pixels was identified as the sample’s spatial dimension. This area within the image was defined according to the geographic location which most of the hurricane tracks pass through. This area is depicted in Fig. 14. The remaining part of the image is mostly irrelevant, as the biggest part of it does not contain hurricane tracks most of the time.

Coping with missing data was carried out as follows. Any pixels missing or corrupt (i.e., a part of a corrupt area) are marked missing. They are then ignored in all processing during the processes described above. These include, among the rest, feature set generation for all classification purposes with feature values marked as missing when using our selected SVM implementation [29] (which is capable of handling missing data), correlation calculation (by ignoring missing data) during the execution of Algorithm 4 (*IPCOSA*), and spatial averaging of pixel groups in *IPCOSA* (feature values are marked as missing and ignored). In a case where a whole image is missing, we treat each of the pixels in the image as individually missing (an edge case of a missing area of pixels, that covers the whole image).

Finally, each sequence’s class was defined as follows—if the severity of the cyclone, along its 15-day duration, was between tropical depression and a category 2 hurricane, and no overlapping hurricanes surpassed category 2, the sequence was labeled *LOW*. However,





**Fig. 14** GOES-12 infrared (*IR*) image of the Atlantic region from Fig. 12, with the area of interest defined by the square

if the severity of the hurricane is category 3 and above, disregarding the severity of the other overlapping hurricanes, the sample was labeled *HIGH*.

Using the above procedure, we have constructed a data-set consisting of 55 samples labeled as *LOW* and of 45 samples labeled as *HIGH*, from the years of 1995 to 2008 (inclusive). The baseline accuracy of this data-set (i.e. ZeroR [29]) is 55%. As we lack a human expert to select pixels for us, we use only the *GIRSS* pixel selection technique. We then fed the results into both the *PIT* and *IPCOSA* feature extraction techniques, for comparative evaluation, with the application of the feature selection (Sect. 2.4) in the third phase. The resulting models were evaluated using a 5-fold cross-validation of the multi-class SVM with linear kernel [29]. Using 5-fold cross-validation instead of using the standard 10-fold cross-validation is due to the long running times of each experiment. Each model's evaluation was performed more than once (each trial yielding a different random 5-fold division), as specified in the results Table 4.

## 4.2 Results

The results for the hurricanes data-set are presented in Table 4. The table shows the results of combining the *GIRSS* pixel selection method with each of the feature extraction techniques, *PIT* and *IPCOSA*. The table caption provides the legend for the entries.

**Table 4** The results of applying *PIT* and *IPCOSA*, on top of *GIRSS* in the hurricane classification domain

		PIT	IPCOSA
<b>GIRSS</b>	$u(1620), r(150)$	64.50 ± 2.2% (4)	62.50 ± 2.29% (4)
	$u(2430), r(100)$	<b>64.67 ± 3.30% (3)</b>	57.67 ± 1.25% (3)
	$u(3300), r(70)$	65.00 ± 4.32% (3)	59.33 ± 1.70% (3)

The numbers in brackets for  $u$  (number of pixels in a random pixel subset) and  $r$  (number of random pixel subsets) are their respective values. The results of the form  $\mu \pm \sigma\%(n)$  have  $\mu$  representing the average accuracy between the trial runs,  $\sigma$  representing the standard deviation and  $n$  representing the number of trial runs. The entry in bold represents the best accuracy obtained. Baseline accuracy is 55%

The table shows that the *PIT* technique outperforms *IPCOSA* in all settings. The reason for this behavior in the VSDI experiments, as will be shown in Sect. 5, is related to high spatial frequency of the data, for which averaging over space has caused the loss of signal. In the case of the hurricanes, this hypothesis for a possible explanation has not been confirmed. Interestingly, settings with small number of pixels yield results that are not inferior to the results produced in settings with large number of pixels, demonstrating the potential scalability of the techniques we introduced. The results are statistically significant when compared to the baseline and appear well above the chance level.

When we take into account the differences between the VSDI and the hurricanes domains, we discover that in practice, even though the two domains share common spatio-temporal characteristics, the key differences between them are fundamental. Nevertheless, these results imply that the methodology presented has the potential to handle classification tasks that originate in extremely varied spatio-temporal domains.

## 5 Discussion

The results above provide a number of insights on the performance of *GIRSS* versus a human domain expert, as well as on the performance of the *PIT* and *IPCOSA* methods for feature selection. We discuss these in detail in the following sections.

### 5.1 *GIRSS* parameters

One important issue to discuss is the selection of the  $u$  and  $r$  parameters, which were manually set in the experiments described above. Initially, we were comparing our performance accuracy to the ROIs provided by the human domain expert (*Oracle*). Since she provided, as the initial ROI, a pixel set of size 104, we compared it to our *GIRSS* technique with  $u = 100$ . In order to make sure, we cover a wide enough selection of pixels, we arbitrarily set  $r = 150$ , which makes for  $u \times r = 1.5$  times the number of pixels.

Once this initial constraint on the parameters was set, we attempted to maintain it, while keeping in mind the selection size of the human domain expert: thus  $u$  was usually around 1–2% of the number of pixels, and  $r$  was adjusted appropriately. In VSDI, the human-selected ROI of the Oriented Gratings (and subsequently, the  $u$ ) was 7% of all the pixels. In the Gabors data-set, it was 1% of the pixels. And in Contours, it was 1.5%.

In the hurricane data-set, we have tried working with  $u = 1, 1.5$  and 2% of all the pixels (hence the three different  $u$  and  $r$  combinations in the results table).  $r$  was set in such a way that  $u \times r = 1.5$  times the number of pixels.

**Table 5** The results of applying *PIT* and *IPCOSA* in the different data-sets

Data-set	Baseline (%)	Best PIT (%)	Best IPCOSA (%)
<b>Hurricanes</b>	55.00	<b>64.67</b>	62.50
<b>Oriented Gratings</b>	34.13	<b>94.90</b>	88.50
<b>Contours</b>	23.39	<b>73.30</b>	<b>73.10</b>
<b>Gabors</b>	18.95	79.10	<b>81.80</b>

The best results in each data-set are shown, together with the baseline results which indicate the complexity of the learning task

### 5.2 PIT vs. IPCOSA

The results of the experiments can lead to conclusions as to the conditions under-which *PIT* and *IPCOSA* can be expected to perform well, and for one of them to outperform the other. We find that when the learning task is relatively easy, *PIT* can be expected to be better. Indeed, notice the change in the relative performance of the two techniques between the different data-sets (Table 5). As the baseline accuracy drops (indicating a more difficult learning task), there is a transition from *PIT* being better, to *IPCOSA* being better.

### 5.3 Spatial averaging contribution

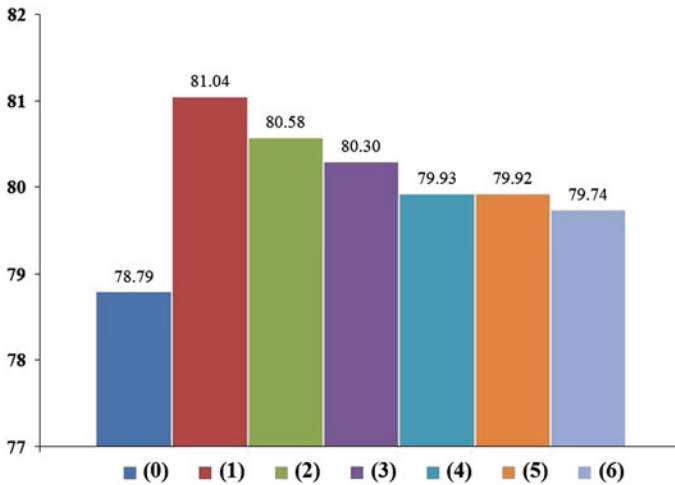
When designing the *IPCOSA* feature extraction technique (Sect. 2.3.2), we examined six different ways for calculating the spatial average value. The question is how to calculate the average of a pixel group having one or more seeds within it and formed in a phase that has a particular graded correlation threshold (the  $\tau$ ). We contrasted the different calculation methods.

Each pixel group is a pixel set  $P' = \{p'_1, p'_2, \dots\}$ . The seeds group is a pixel subset  $P^* = \{p^*_1, p^*_2, \dots\}$  of  $P'$ :  $P^* \subseteq P'$ . The group  $P^- = \{p^-_1, p^-_2, \dots\}$  is defined as:  $P^- = P' \setminus P^*$ . Each pixel  $p$  in any of the defined sets has only one value in time:  $\bar{p} = \langle v_T \rangle$ ,  $v_T \in \mathbb{R}$ ,  $T \in \{1, \dots, m\}$  (since the averaging is done in a specific fixed point in time  $T$ ), so for simplicity we will refer to the pixel’s single value in time using the pixel’s notation (e.g.  $p$  will denote  $v_T$ ). The averaging methods are

1. Plain average:  $\frac{\sum_{p' \in P'} p'}{|P'|}$
2. Weighted average:  $\frac{\sum_{p^* \in P^*} p^* + \tau \sum_{p^- \in P^-} p^-}{|P^*| + \tau |P^-|}$
3. Square weighted average:  $\frac{\sum_{p^* \in P^*} p^{*2} + \tau^2 \sum_{p^- \in P^-} p^{-2}}{|P^*| + \tau^2 |P^-|}$
4. Group and seeds split weighted average:  $\frac{\frac{|P^-|}{|P^*|} \sum_{p^* \in P^*} p^* + \sum_{p^- \in P^-} p^-}{2|P^-|}$
5. Thresholded group and seeds split weighted average:  $\frac{\frac{|P^-|}{|P^*|} \sum_{p^* \in P^*} p^* + \tau \sum_{p^- \in P^-} p^-}{(1 + \tau)|P^-|}$
6. Seeds only average:  $\frac{\sum_{p^* \in P^*} p^*}{|P^*|}$

Figure 15 compares the average accuracies of the spatial averaging methods above, as well as *PIT*, applied on each of the 7 experimental runs executed on the Gabors data-set. The best method is the plain average method. This was the method used in the experiments.

Due to the high resolution of the signal in the Oriented Gratings, we see that the spatial averaging only worsens the results instead of improving them. This is an expected result—the signal in this case arises from small orientation columns, while averaging over space



**Fig. 15** Comparison of seed-based pixel groups averaging methods: 0 accuracy of applying *PIT*; 1 accuracy of applying *IPCOSA* with plain average; 2 accuracy of applying *IPCOSA* with weighted average; 3 accuracy of applying *IPCOSA* with square weighted average; 4 accuracy of applying *IPCOSA* with group and seeds split weighted average; 5 accuracy of applying *IPCOSA* with thresholded group and seeds split weighted average; 6 accuracy of applying *IPCOSA* with seeds only average

smears them out, causing the loss of signal—hence, the loss of the data’s essential properties. However, with the Gabors and the Contours, we see quite the opposite—spatial averaging provides additional enhancement to the classification abilities. Being much harder to distinguish than with the first data-set case, the types of the visual stimuli of these two data-sets lead to collection of data in which the activation has, at least partially, low spatial frequency characteristics, as opposed to the Oriented Gratings (some of the information in this case has to do with the retinotopic activation). In conclusion, the spatial averaging role depends on the size of the neuronal spatial modules that encode it, leaving space for improvement by the advanced feature extraction technique in data-sets characterized by low spatial frequency.

#### 5.4 Ruling out over-fitting

To disprove the likelihood of high accuracy rates due to over-fitting, we proceeded with additional validation of the results. In VSDI data in particular, the significance of each of the stimuli conditions is realized only after the visual stimulus onset. That is to say, the discrimination between the different stimuli (i.e., the classification of the different classes), is only possible after the stimuli were shown to the monkey, and the appropriate neuronal population responses were provoked. Had we observed the responses of the same neuronal populations, solely before the onset of the stimuli, we would not expect to have the ability to discriminate between them—simply because of the fact that the behavior of these responses is expected to be similar to the ones provoked by the blank control image, where no stimulus is presented (which is exactly the case). Based on this reasoning, we establish the following validation procedure.

We repeat the same experiments as detailed in Sect. 3.1, but in all our data-sets, we used only the first part of each sequence of sample points, limiting the examples to the intervals before the onset of stimuli (see the results tables discussed below for details). We then apply *GIRSS* again. We expect the classification results to be close to the baseline accuracies

**Table 6** Results in the modified ORIENTED GRATINGS data-set (using only the first 10 points in time before the visual stimulus onset)

		PIT	IPCOSA
<b>GIRSS</b>	$u(154), r(20)$	$31.6 \pm 1.8\% (5)$	$34.0 \pm 1.3\% (5)$

Baseline accuracy is 34.13%, chance: 33.33%

**Table 7** Results in the modified CONTOURS data-set (using only the first 3 points in time before the visual stimulus onset)

		PIT	IPCOSA
<b>GIRSS</b>	$u(151), r(100)$	$21.0 \pm 2.3\% (5)$	$22.7 \pm 0.6\% (5)$
	$u(500), r(100)$	$19.0 \pm 2.9\% (5)$	$22.4 \pm 0.6\% (5)$

Baseline accuracy is 23.39%, chance: 20.00%

**Table 8** Results in the modified GABORS data-set (using only the first 10 points in time before the visual stimulus onset)

		PIT	IPCOSA
<b>GIRSS</b>	$u(100), r(150)$	$17.5 \pm 1.6\% (5)$	$15.4 \pm 2.1\% (5)$

Baseline accuracy is 18.95%, chance: 16.67%

of each of the data-sets, as the examples now lack the sample points corresponding to the different stimulus responses.

The results from this evaluation are presented in Tables 6, 7 and 8. The tables present the validation experiment results for each of the VSDI data-sets, using both feature extraction techniques, *PIT* and *IPCOSA*. The numbers in brackets for  $u$  (number of pixels in a random pixel subset) and  $r$  (number of random pixel subsets) are their respective values. The results of the form  $\mu \pm \sigma\%(n)$  have  $\mu$  representing the average accuracy between the trial runs,  $\sigma$  representing the standard deviation and  $n$  representing the number of trial runs.

The results displayed in the tables show that lacking the sample points *after* the exposure to the stimuli, the techniques we develop are no longer able to classify the different examples into the target classes. In fact, the classification rates drop to chance level. Thus, we can conclude that indeed the learning techniques use the information present in the sampled signals, as a response to the stimuli.

### 5.5 Scalability and feasibility

Before basing our pixel selection methods on random subspace [13], we had initially attempted other methods for handling the VSDI data. In particular, we employed techniques that base their feature extraction, selection and classification on the full spatio-temporal range (resembling methods proposed in [15, 16, 19, 28]). However, these attempts ended with impractical running times and memory requirements.

Here's a concrete example for one such early trial. The  $ROI_1$  set of the Gabors data-set, which includes 104 pixels, was turned into a feature-set using the *PIT* approach (Sect. 2.3.1), resulting in 5304 features. The features were ranked with InfoGain scores (as in Algorithm 3, step 2b), and sorted top down (lowest-ranked feature at the bottom of the list). Iteration over the feature set was performed in an RFE-like [8] manner (but with a different feature

weighting mechanism): first, the whole feature set was preserved, the data-set was trained using the SVM classifier, either InfoGain filtered as in Sect. 2.4 or not filtered at all, and 10-fold cross-validated; the evaluation score of this step was written down. In the second iteration, the lowest-ranked feature was removed from the feature set, training and validation was repeated, and a new evaluation score was written down. In this recursive manner, the process was repeated for all the features, until the last and only, top-ranked feature, remained in the feature set. The number of top-ranked features to choose for the model construction was the one with the highest evaluation score collected during the execution of this procedure.

Without judging the quality or the motivation for the above scenario, the magnitude of its running times is roughly the same as of the methods having a resembling nature (such as [16, 19]), where a classifier is trained for each pixel-time combination during the process. The running times of this scenario applied to the Gabors  $ROI_1$  is between 25 and 50 h, depending on whether the InfoGain filtering was applied. Had we managed to run this scenario on the full Gabors pixel set rather than on  $ROI_1$ , this would at first seem like it would have taken between 10 and 20 days; however, this relation is not linear—although these estimates are for classifier training and evaluation based on 10,000 pixels, as opposed to only 104, they are based on a recursion that starts from 104 pixels only. A more correct estimate would be based on running times of  $\sim 30$  min per pixel on average, resulting in an initial estimate of, and easily surpassing, 200 days.

Moreover, we were not even able to run this scenario, neither a few other ones having a resembling pixel-time pairs-based iterative nature, due to the impractical I/O and memory requirements. The basic instances initialization during the initial loading of the Gabor data-set would take tens of minutes due to an intensive I/O, only to crash later on insufficient memory (albeit using a 32 bit architecture OS); this would happen before completing the initialization—not to speak of moving to the next step of basic low profile operations such as InfoGain-based filtering. While the Contours data-set has about the same impractical magnitudes, with Oriented Gratings the experience is slightly different. Here, the loading of a data-set based on only 2 out of 3 available classes, having a 5 times smaller pixels number, but having twice as bigger number of samples, would succeed after less than a few minutes. After additional few minutes of cross-validation though, approximately at the 5th fold, the process would crash—yet again—on insufficient memory.

Finally, the memory obstacle remains relevant even if we remove the run-time challenges of training classifiers for each pixel-time combination. It is enough to see that we cannot load the initial data based on all the available raw values, even before moving to any feature selection or classifier training steps.

However, with *GIRSS* and *IPCOSA*, we were able to build models using a single-threaded Java application on a Core 2 Duo machine with 2GB of RAM, in less than 2 h for the Oriented Gratings, roughly 8 h for the Gabors, and between 8 and 13 h for the Contours data-sets. Using the *PIT* instead of the *IPCOSA* lowers these times by up to an order of magnitude. This demonstrates that *PIT* and *IPCOSA* are not only feasible, but practical.

## 5.6 Sliding windows: averaging along the time course

During our work, we performed various experiments with a time-course reduction using a simple Sliding Window (SW) technique. The motivation for this kind of work comes from two reasons. First, applying SW technique is appropriate with data-sets in which the time dimension poses a dimensionality threat. In such cases, effective reduction in the time dimension makes the classification process more feasible. In addition, this technique can reduce the influence of a noisy data along the time course. However, the first reason was irrelevant,

having the number of time points in our data significantly lower than the number of pixels. In regard to the second reason, one of the experimental techniques is detailed in the next paragraphs.

When using a SW, two parameters are defined: the time window size  $w$  and the overlap factor  $o$ . The  $w$  specifies the number of consecutive points in time along which the averaging is performed. The  $o$  specifies the extent of the overlap between each two consecutive time windows. For instance, having  $w = 3$  and  $o = 1$ , indicates averaging in the following manner:

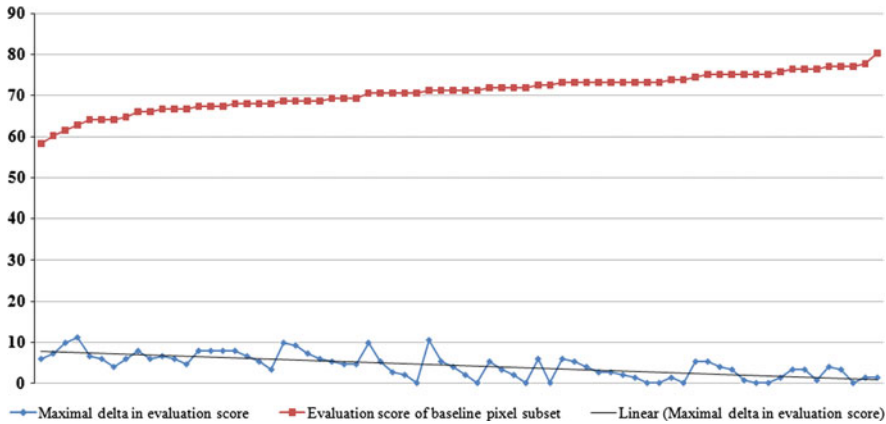
$$\begin{aligned} & \text{average}(\text{timepoint}_1, \text{timepoint}_2, \text{timepoint}_3) \\ & \text{average}(\text{timepoint}_3, \text{timepoint}_4, \text{timepoint}_5) \\ & \quad \vdots \\ & \text{average}(\text{timepoint}_{m-4}, \text{timepoint}_{m-3}, \text{timepoint}_{m-2}) \\ & \text{average}(\text{timepoint}_{m-2}, \text{timepoint}_{m-1}, \text{timepoint}_m) \end{aligned}$$

Note that the following will always hold:  $w \geq 2$ ,  $o \leq w - 1$ ; we have also defined that overlap must exist:  $o \geq 1$ .

Given the 51 time points in the Gabor data-set, we have tried every possible combination conforming to the following definition:  $2 \leq w \leq 10$ ,  $1 \leq o \leq w - 1$  (resulting in 45 different combinations). We then randomly generated 70 pixel subsets, each having a 100 pixels in a subset. For each of these subsets, and for each of its SW variations (resulting in  $(1 + 45) \cdot 70 = 3,220$  of both regular and time-averaged pixel subsets), we have applied a slight variation of our pixel set evaluation method (Algorithm 2), one that handles the values averaged over the time course. The evaluation scores produced by this method were then compared and analyzed, and a comparison between the “regular” (baseline) pixel subsets and the time-averaged pixel subsets was made.

We thus reach the following conclusion: while the sliding window can significantly improve the evaluation score of an arbitrary pixel subset by up to 11.1% (the improvement gain), this improvement will be significant only for the weaker pixel subsets—the ones that in the first place, prior to averaging, were producing low evaluation scores. In fact, the higher the evaluation score of a baseline pixel subset was, the lower was the gain in accuracy (the delta) of applying any of the 45 sliding window variations. The highest-ranked baseline pixel subsets do not benefit from the application of the sliding window. To support this claim, we checked the correlation between the evaluation scores of the baseline pixel subsets, compared to the maximal delta among all 45 possible deltas of the same subset, and revealed a negative correlation coefficient of  $-0.62$ . Figure 16 illustrates these findings. Nevertheless, it is important to mention that the SW had never decreased the evaluation score of the time-averaged pixel subsets—there were no negative deltas.

In a different type of experiment, a wide variety of sliding windows was applied on the Gabors  $ROI_1$  subset (as opposed to applying them on randomly generated pixel subsets). While the evaluation of this subset, along with the evaluation of its SW variations, was different than the techniques introduced in this work (e.g. different usage of InfoGain, application of a feature discretization technique, etc.), the comparison still showed that the maximal gained evaluation score of  $ROI_1$  of  $\sim 57\%$  could not be surpassed by any of the tested SW variations. When a computationally intensive, RFE-like procedure (resembling the one demonstrated in Sect. 5.5) was applied on every SW variation, after much effort and weeks of waiting, a combination of features was found that had produced an evaluation score of 60.8%. While this may show that an apparent benefit can be gained from applying SW, finding this appropriate feature combination using these methods is impractical, for the reasons detailed in Sect. 5.5. Moreover, even if there exists a practical way of extracting the appropriate feature



**Fig. 16** Comparison of the evaluation scores of baseline pixel subsets to the maximal improvement gained (maximal delta) in the evaluation scores of these subsets, after applying the 45 sliding window variations. The chart is ordered by ascending evaluation scores of the baseline pixel subsets. A linear trend-line is displayed for the maximal delta series

combination, the conclusion from the previous type of experiment discussed above shows that the gain in the evaluation score is apparent only for the more inferior pixel subsets, such as  $ROI_1$ .

## 6 Related work

The techniques described in this section are used for feature extraction and feature selection of problems that handle either spatial, temporal or spatio-temporal data. We explore the methods relevant for our work and survey the classification algorithms being used.

*Approaches ignoring temporal aspects.* Machine-learning methods in fMRI-based neuroimaging do not usually take into account the temporal aspect of the domain, and instead focus on spatial properties alone. This is because fMRI measures metabolic changes, and as such its temporal resolution is at least two orders of magnitude slower than neural activity (which measures in tens of ms). Spatial features in this domain included picking the top  $n$  most active voxels based on  $t$ -test [15] or on average between the voxels [28]; picking the top  $n$  most discriminating voxels, based on training a classifier per each voxel [16]; or, picking the  $n$  most active voxels per Region Of Interest (ROI) [16]. The classifiers used in such studies were mostly Gaussian Naïve Bayes (GNB), Support Vector Machine (SVM), and k-Nearest Neighbor (kNN). While these studies managed to produce moderate to high accuracy results, they relied on relatively small resolutions of data (where training a classifier per voxel was admissible), or on expert knowledge (defining an ROI). Given the large-scale characterization of our domains, these methods are not applicable to our problems. Furthermore, as was shown in our experiments, our automated method generates features that yield an accuracy at least as high as those of the ROI features and in some settings even higher.

When facing higher-resolution fMRI neuroimages, one method employed uniform sampling of the data [4] to reduce the number of features. While sampling is indeed a reasonable tool when handling high-resolution data, a uniform sampling is lacking in a sense that it



does not choose the more discriminative data-points, in favor of the less discriminative ones. Hence, while our pixel selection technique is also based on sampling, it adopts elements from the random subspace selection method presented in [13] rather than basing its selection on uniform sampling which does not use domain knowledge to target the sampling.

The multivariate search technique applied on a subspace randomly selected that were presented in [13] chooses randomly  $t$  sets of features. Then the features' weight is a combination of the results of all iterations and the highest-ranked features are selected. Since our domains consist of both spatial and temporal aspects, we extended this method by not using all the randomly selected subsets for evaluating features but by choosing a smaller subset using a greedy approach. This allows us to explore a major part of our large set of pixels while maintaining a reasonable time complexity.

In [6], a classification was carried out of very high-resolution panchromatic images from urban areas. A spatial (area) filter was used to extract information about the inter-pixel dependency. Using a linear composition of kernels, a kernel was defined using both the spectral (i.e. the original gray level of each pixel) and the spatial information, reaching partial success in some types of areas. While this domain resembles the domains in our focus only in its spatial nature, we liked the idea of exploring the inter-pixel dependency and further developed it for our feature extraction technique in the *IPCOSA* algorithm when forming neighborhoods. However, while in [6]'s approach, the neighborhood of one given pixel is defined as the resulting connected zone of the self-complementary area filter, in our *IPCOSA* algorithm neighborhoods are defined to include at least one seed (pixels that were generated in the *GIRSS* algorithm) and pixels that are spatially located near the given seed but also very similar to it.

Another interesting work of [7] studied large-scale classification at multiple spatial resolutions from remote sensing raster data. They proposed methods to reduce computational time significantly using context but do not face the temporal aspect of the spatial data and their method is not applicable to our domains of single resolution.

*Spatio-temporal machine learning.* Classification in large-scale spatio-temporal domains often requires both spatial and temporal dimensionality reduction. With respect to the reduction of the temporal dimension, proven methods from time-series processing include the Discrete Fourier Transform (DFT) and the Discrete Wavelet Transform (DWT).

Vlachos et al. [27] have used DWT for dimensionality reduction of time series. Their objective was to find a data representation at a lower dimensionality that preserves the original information, describing the original shape of the time series data as closely as possible. As a result, they have introduced an improved version of  $k$ -means clustering algorithm that was shown to have results superior to  $k$ -means. DWT and DFT were also successfully employed in [17] for time series data mining, where each time series was compressed with wavelet or Fourier decomposition.

However, our methods consider the spatial dimension much more significantly than the temporal dimension—for which we chose a rather simplistic approach. The reason for taking the spatial-first approach lies in the nature of the data that was used during the development of our methods. The spatial resolution of our data places a much greater scalability challenge than the temporal resolution, which poses a considerably smaller dimensionality threat.

Despite comparatively small efforts to reduce the temporal dimension, we did, however, attempt to apply simple Sliding Window (SW) techniques for temporal reduction while designing our methodology—but without any apparent advantage. These attempts were described in detail in Sect. 5.6.

Palatucci [19] exploited the following heuristic in processing fMRI neuroimages using also temporal information: features were defined as voxel-timepoint pairs, ranked by how well they individually classify a training set, and the top 100 features for the final classifier were chosen. Individual training of classifiers for all time-space combinations is computationally impractical in our large-scale domains and thus we had to carefully select pixels before considering the time aspect. However, in specific places in our algorithms, we could adopt the time-space combination approach, thanks to the reduced pixel set:

- First, when evaluating pixels of a given randomly generated set in the *GIRSS* algorithm all pixel-time pairs are considered.
- Second, the *PIT* method takes all pixel-time pairs of the pixels selected by the *GIRSS* algorithm.
- Finally, in the *IPCOSA* algorithm averaging of neighboring pixels in the generated neighborhoods is done for each time point.

We consider that in addition to the temporal deficiency of fMRI, it samples the space in voxels of one to a few millimeters (only a few thousands of voxels, or well-defined regions of interest). As opposed to it, VSDI is capable of measuring neuronal population responses at high spatial and temporal resolutions. Therefore, it provides a true insight as to the neuronal dynamics from both spatial and temporal aspects. An early work on the decoding of neuronal population responses to visual detection tasks using VSDI is [3]. A specially designed method based on six neuronal population responses pooling rules was used here, with no machine learning use whatsoever. This method relied on the amplitude of the response, and other neuronal characteristics—tailored strictly for VSDI. While this method has produced nearly perfect results, it incorporates a domain-specific knowledge—rendering it as not general enough for use in spatio-temporal domains other than VSDI.

Additional work that has inspired us is [34], which introduced a technique for feature selection by defining voxel-specific time-series analysis, by ranking features by mutual information with respect to the class variable. From the ranked features, the  $n$  highest ranked were selected, and closeness of each pair of voxels' time series was measured. Though yielding high success rates, the techniques in [34] are computationally expensive since they consider a time series for each pixel. This is not applicable in large-scale domains such as ours and therefore we had to introduce the *GIRSS* algorithm which selects a subset of pixels before considering the time aspect.

Yang et al. [32] presented a method for maintaining the correlation information between spatial time-series items by utilizing the correlation coefficient matrix of each such item as features to be employed for classification. Then, Recursive Feature Elimination (RFE) is used for feature subset selection of time-series data-sets. RFE was first proposed for gene selection problem in [8], where the SVM's weight factor was used as ranking criterion for features, and the features with the smallest ranking criterion were recursively eliminated. However, applying RFE or an RFE-like procedure, in a similar manner, on the data discussed in this paper is computationally expensive, as we show experimentally. Nevertheless, we do adopt the approach of correlation between spatial elements in the *IPCOSA* algorithm when we form the neighborhoods around seeds.

A last example from spatio-temporal domains is automated video genre classification [30]. Here, classification was carried out by first computing a spatio-temporal combined audio-visual feature vector (of very high dimensionality). Then, the feature vector was further processed using Principal Component Analysis (PCA) to eliminate redundancy while exploiting the correlations between feature elements. Such PCA-based techniques in multivariate time-series data-sets are known to be difficult to scale. The complexity of the PCA-based

techniques is considered in [31], but the proposed solution is still too time consuming for our domains.

In terms of more general spatio-temporal learning, a few recent examples are related, especially on the issue of scalability. Chan et al. [2] tackle graph-mining, in graphs that evolve over space and time. Our work does not represent images as graphs, and thus differs from this work. Kang et al. [11] face typical graph mining tasks in graphs of several Giga-, Tera- or Peta-bytes. While our spatio-temporal data is much smaller, it still does not fit into main memory and is more complex since it cannot be described using graphs.

## 7 Conclusions

In this paper, we introduced a combination of novel techniques for handling spatio-temporal classification tasks of large scale. We evaluated the techniques in two very different domains: VSDI neuroimaging and hurricane images. In both, we have demonstrated that the techniques work well, at human expert level or above it. This, despite significant conceptual differences that exist between the two domains. We therefore believe that the techniques presented carry great potential.

We plan to continue to investigate techniques for machine learning in large-scale domains, and in particular in the two domains presented above. In addition, we believe that our methods are currently naïve in that they do not take significant steps to reduce the temporal dimensionality of the data. We believe that complementing our methods with other proven methods from time-series processing, such as Discrete Fourier Transform [17] or Discrete Wavelet Transform [27] can be of great benefit in this respect.

**Acknowledgments** We thank anonymous reviewers for their very useful comments and suggestions. This research was supported in part by ISF grants #859/05 (to H. S.) and #1357/07. We thank Elhanan Meirovithz for his help in VSDI data acquisition, and Ofer Garnett for his helpful advice. As always, thanks to K. Ushi and V. Ricki for their support.

## References

1. Boynton GM (2005) Imaging orientation selectivity: decoding conscious perception in V1. *Nat Neurosci* 8(5):541–542
2. Chan J, Bailey J, Leckie C (2008) Discovering correlated spatio-temporal changes in evolving graphs. *Knowl Inform Syst* 16(1):53–96
3. Chen Y, Geisler WS, Seidemann E (2006) Optimal decoding of correlated neural population responses in the primate visual cortex. *Nat Neurosci* 9(11):1412–1420
4. Davatzikos C, Ruparel K, Fan Y, Shen DG, Acharyya M, Loughead JW, Gur RC, Langleben DD (2005) Classifying spatial patterns of brain activity with machine learning methods: application to lie detection. *Neuroimage* 28(3):663–668
5. Fatourehchi M, Birch G, Ward R (2007) Application of a hybrid wavelet feature selection method in the design of a self-paced brain interface system. *J Neuroeng Rehabil* 4(1):11
6. Fauvel M, Chanussot J, Benediktsson JA (2007) A joint spatial and spectral SVM's classification of panchromatic images. In: *IEEE international geoscience and remote sensing symposium (IGARSS'07)* pp 1497–1500
7. Gandhi V, Kang JM, Shekhar S, Ju J, Kolczyk ED, Gopal S (2009) Context inclusive function evaluation: a case study with em-based multi-scale multi-granular image classification. *Knowl Inform Syst* 21(2):231–247
8. Guyon I, Weston J, Barnhill S, Vapnik V (2002) Gene selection for cancer classification using support vector machines. *Mach Learn* 46(1–3):389–422

9. Han J, Kamber M (2000) Data mining: concepts and techniques, chap 3. Morgan Kaufmann, Los Altos, p 121
10. Kamitani Y, Tong F (2006) Decoding seen and attended motion directions from activity in the human visual cortex. *Curr Biol* 16(11):1096–1102
11. Kang U, Tsourakakis CE, Faloutsos C (2010) Pegasus: mining peta-scale graphs. *Knowl Inform Syst* (in press)
12. Knapp KR (2008) Scientific data stewardship of international satellite cloud climatology project B1 global geostationary observations. *J Appl Remote Sens* 2(1):023548
13. Lai C, Reinders MJ, Wessels L (2006) Random subspace method for multivariate feature selection. *Pattern Recogn Lett* 27(10):1067–1076
14. Lee H, Choi S (2003) PCA+HMM+SVM for EEG pattern classification. In: Proceedings of the 7th international symposium on signal processing and its applications, vol 1, pp 541–544
15. Mitchell T, Hutchinson R, Just MA, Niculescu RS, Pereira F, Wang X (2003) Classifying instantaneous cognitive states from fMRI data. In: Proceedings of the 2003 American medical informatics association annual symposium. p 469
16. Mitchell TM, Hutchinson R, Niculescu RS, Pereira F, Wang X, Just M, Newman S (2004) Learning to decode cognitive states from brain images. *Mach Learn* 57(1-2):145–175
17. Mörchen F (2003) Time series feature extraction for data mining using DWT and DFT, technical report 33. Department of Mathematics and Computer Science, University of Marburg, Germany
18. Mourao-Miranda J, Friston KJ, Brammer M (2007) Dynamic discrimination analysis: a spatial-temporal SVM. *NeuroImage* 36(1):88–99
19. Palatucci M (2007) Temporal feature selection for fMRI analysis. Working paper (unpublished)
20. Palatucci M, Mitchell TM (2007) Classification in very high dimensional problems with handfuls of examples. In: Principles and practice of knowledge discovery in databases (ECML/PKDD), vol 4702 of Lecture Notes in Computer Science. Springer, pp 212–223
21. Simpson RH, Riehl H (1981) The hurricane and its impact. Louisiana State University Press, Baton Rouge
22. Singh S (2000) EEG data classification with localized structural information. In: Proceedings of the 15th international conference on pattern recognition (ICPR'00). pp 2271–2274
23. Singh V, Miyapuram KP, Bapi RS (2007) Detection of cognitive states from fMRI data using machine learning techniques. In Proceedings of 20th international joint conference on artificial intelligence (IJCAI'07). pp 587–592
24. Slovín H, Arieli A, Hildesheim R, Grinvald A (2002) Long-term voltage-sensitive dye imaging reveals cortical dynamics in behaving monkeys. *J Neurophysiol* 88(6):3421–3438
25. Song L, Smola A, Gretton A, Borgwardt KM, Bedo J (2007) Supervised feature selection via dependence estimation. In: Proceedings of the 24th international conference on machine learning (ICML'07) ACM. pp 823–830
26. Stopel D, Boger Z, Moskovitch R, Shahar Y, Elovici Y (2006) Improving worm detection with artificial neural networks through feature selection and temporal analysis techniques. *Int J Comput Sci Eng* 15:202–208
27. Vlachos M, Lin J, Keogh E, Gunopulos D (2003) A wavelet-based anytime algorithm for K-Means clustering of time series. In: Workshop on clustering high dimensionality data and its applications at the 3rd SIAM international conference on data mining. pp 23–30
28. Wang X, Hutchinson R, Mitchell TM (2004) Training fMRI classifiers to discriminate cognitive states across multiple subjects. In: Thrun S, Saul L, Schölkopf B (eds) Advances in neural information processing systems 16. MIT Press, Cambridge
29. Witten IH, Frank E (2005) Data mining: practical machine learning tools and techniques, 2nd edn. Morgan Kaufmann, San Francisco
30. Xu L-Q, Li Y (2003) Video classification using spatial-temporal features and PCA. In: Proceedings of the 2003 international conference on multimedia and expo (ICME'03), vol 3. pp III485–III488 vol 3
31. Yang K, Shahabi C (2005) A PCA-based kernel for kernel PCA on multivariate time series. In: Proceedings of ICDM 2005 workshop on temporal data mining: algorithms, theory and applications held in conjunction with the fifth IEEE international conference on data mining (ICDM'05). pp 149–156
32. Yang K, Yoon H, Shahabi C (2005) A supervised feature subset selection technique for multivariate time series. In: Proceedings of international workshop on feature selection for data mining: interfacing machine learning with statistics (FSDM) in conjunction with 2005 SIAM international conference on data mining (SDM'05). pp 92–101
33. Yoon H, Shahabi C (2006) Feature subset selection on multivariate time series with extremely large spatial features. In: Proceedings of the 6th IEEE international conference on data mining—workshops (ICDMW'06). IEEE Computer Society, pp 337–342

34. Zhang L, Samaras D, Tomasi D, Alia-Klein N, Cottone L, Leskovjan A, Volkow ND, Goldstein R (2005) Exploiting temporal information in functional magnetic resonance imaging brain data. In: Proceedings of the 8th international conference on medical image computing and computer-assisted intervention (MICCAI'05), vol 3749 of Lecture Notes in Computer Science. Springer, pp 679–687
35. Zhao Q, Zhang L (2007) Temporal and spatial features of single-trial EEG for brain-computer interface. *Comput Intell Neurosci* 2007(1):4

## Author Biographies



**Igor Vainer** has received his M.Sc. in Computer Science from Bar-Ilan University in 2009. His research interests are in machine learning and data mining fields, and their application to challenging domains. In his master's thesis, he has developed methods for obtaining scalable and accurate classification in large-scale spatio-temporal domains with effective application to real-world data. With 12 years of industry experience behind him, he now works as a lead developer at Aternity Inc., holding responsibility for the optimization of data processing engines and for the development of highly scalable data warehousing solutions.



**Sarit Kraus** (Ph.D. Computer Science, Hebrew University, 1989) is a Professor of Computer Science at Bar-Ilan University (Israel) and Adjunct Professor at the Institute for Advanced Computer Studies, University of Maryland. Her research interests are in multi-agent systems, specially negotiation and cooperation among agents, learning and information agents, personalization and optimization of complex systems. In 1995 Prof. Kraus was awarded the IJCAI Computers and Thought Award (the premier award for a young AI scientist). In 2002 she was elected as AAAI fellow, in 2007 she was awarded the ACM SIGART Agents Research award and her paper with Prof. Grosz was a winner of the IFAAMAS influential paper award. In 2008 she was elected as ECCAI fellow. She has published over 260 papers, she is an author of the book *Strategic Negotiation in Multiagent Environments* (2001) and a co-author of a book on *Heterogeneous Active Agents* (2000); both published in MIT Press.



**Gal A. Kaminka** is an associate professor at the Computer Science department, and the Brain Research Center, at Bar-Ilan University (Israel). His research expertise includes multi-agent and multi-robot systems, teamwork and coordination, behavior and plan recognition, and modeling social behavior. He has received his Ph.D. from the University of Southern California, and spent two years as a postdoctorate fellow at Carnegie Mellon University. Today, Prof. Kaminka leads the MAVERICK research group at Bar-Ilan, supervising over a dozen M.Sc. and Ph.D. students. He was awarded an IBM faculty award and top places at international robotics competitions. He served as the program chair of the 2008 Israeli Conference on Robotics, and the program co-chair of the 2010 Int'l Joint Conference on Autonomous Agents and Multi-Agent Systems. He is currently serving on the international executive bodies of IFAAMAS (International Foundation of Autonomous Agents and Multi-Agent Systems) and AAAI (Association for Advancement of Artificial Intelligence).



**Hamutal Slovin** is a lecturer at the Leslie and Susan Gonda Multidisciplinary Brain Research Center and Faculty of Life Sciences at Bar-Ilan University (Israel). Her research expertise includes optical imaging of intrinsic signals, voltage-sensitive dye imaging and electrophysiological recordings from the cortex of behaving animals. Dr. Slovin has received her Ph.D. from the Hebrew University, and spent three years as a postdoctorate fellow at the Weizmann Institute (Israel). Today Dr. Slovin leads the Laboratory for Neural Imaging at Bar-Ilan. Recently she was awarded the "Alon scholarship" for young excellent scientist and the "Leon and Maria Taubenblatt Prize for Excellence in Medical Research". Her current research focuses on (a) Understanding the neuronal mechanisms underlying visual perception and higher brain functions: visual attention and perceptual learning. (b) Decoding (e.g. "Mind reading") of visual scenes content from brain activity. (c) Brain-Machine interface: Inducing artificial vision using electrical stimulation of the brain.