Signal Processing 174 (2020) 107634

Contents lists available at ScienceDirect

Signal Processing

journal homepage: www.elsevier.com/locate/sigpro

Evolutionary simplicial learning as a generative and compact sparse framework for classification

Yigit Oktar[a], Mehmet Turkan[b,*]

[a] Department of Computer Engineering, Izmir University of Economics, Izmir, Turkey
[b] Department of Electrical and Electronics Engineering, Izmir University of Economics, Izmir, Turkey

ARTICLE INFO

Article history:
Received 3 December 2019
Revised 24 April 2020
Accepted 27 April 2020
Available online 17 May 2020

Keywords:
Sparse representations
Machine learning
Simplex
Simplicial
Dictionary learning
Classification

ABSTRACT

Dictionary learning for sparse representations has been successful in many reconstruction tasks. Simplicial learning is an adaptation of dictionary learning, where subspaces become clipped and acquire arbitrary offsets, taking the form of simplices. Such adaptation is achieved through additional constraints on sparse codes. Furthermore, an evolutionary approach can be chosen to determine the number and the dimensionality of simplices composing the simplicial, in which most generative and compact simplicials are favored. This paper proposes an evolutionary simplicial learning method as a generative and compact sparse framework for classification. The proposed approach is first applied on a one-class classification task and it appears as the most reliable method within the considered benchmark. Most surprising results are observed when evolutionary simplicial learning is considered within a multi-class classification task. Since sparse representations are generative in nature, they bear a fundamental problem of not being capable of distinguishing two classes lying on the same subspace. This claim is validated through synthetic experiments and superiority of simplicial learning even as a generative-only approach is demonstrated. Simplicial learning loses its superiority over discriminative methods in high-dimensional cases but can further be modified with discriminative elements to achieve state-of-the-art performance in classification tasks.

1. Introduction

Sparse representations have been proven to be very successful at restoration and reconstruction tasks such as compression, denoising, deblurring, inpainting and superresolution [1]. In essence, they aim at modeling the data/signal through concise linear combinations attained from an overcomplete basis or set of elements. This overcomplete set of elements is named as the *dictionary* and it can either be carefully fixed (experimentally or analytically) or be adapted to the data at hand through learning [2]. Conventional nonconvex optimization of dictionary learning for sparse representations is given in Eq. (1) as follows,

$$\arg\min_{\mathbf{A},\{\mathbf{x}_i\}} \sum_i \|\mathbf{y}_i - \mathbf{A}\mathbf{x}_i\|_2^2 \quad \text{subject to} \quad \|\mathbf{x}_i\|_0 \leq q, \ \forall i, \tag{1}$$

where the matrix $\mathbf{A}$ is the designated overcomplete dictionary and $\mathbf{x}_i$ is the sparse representation vector of the data point $\mathbf{y}_i$, $\forall i$. While minimizing the reconstruction error of $\mathbf{y}_i$ over the dictionary $\mathbf{A}$, each sparse vector $\mathbf{x}_i$ can have a maximum q number of nonzero components due to the strict ℓ_0-norm constraint. In literature, there exist approximate iterative solutions (namely, *sparse coding* and *dictionary update*) to this highly nonconvex problem and its variants [3].

In addition to reconstructive signal processing tasks, dictionary learning can also be employed in machine learning problems such as classification and clustering [4–6]. At this point, it is proper to introduce one-class classification, as the fundamental form of the general classification problem, to bridge the gap between reconstructive signal processing and machine learning. Supervised machine learning in the form of classification inherently suggests the existence of more than one label. The concept of one-class learning, also known as one-class or unitary classification, emerges when there only exists a single label within the dataset, and one needs to discriminate it against all possible unseen labels [7]. It is actually a special case of binary classification where there is the "in-class" label and also the "out-of-class", but there is not any or enough number of "out-of-class" samples within the training dataset. Therefore, in the absence or weakness of the opposing class samples, conventional binary classification methods will have difficulties as they target the decision boundary in-between.

* Corresponding author.
E-mail address: Mehmet.Turkan@ieu.edu.tr (M. Turkan).
URL: http://people.ieu.edu.tr/en/mehmetturkan (M. Turkan)

https://doi.org/10.1016/j.sigpro.2020.107634
0165-1684/

One-class learning methods can be categorized by the type of the targeted classifier model. There exist decision-boundary approaches which seek enclosing hyperspheres, hyperplanes or hypersurfaces in general [8]. These methods can adjust the level-of-detail through the usage of parametrized kernels to cope with the over- or under-fitting problem. On the other hand, graph-based methods try to fit a skeleton with-in data in a bottom-up manner. As an example, a minimum spanning tree model can be utilized as a one-class classifier [9], in which the classification procedure relies on the distance to the tree. A generalization of graph-based approaches is attained through the concept of hypergraph, in which a hyperedge can now connect more than two data points or vertices. Hypergraph models not only allow custom but also lead the way to heterogeneous dimensionality. Such models are investigated in Wei et al. [10], Silva and Willett [11]. As detailed in Section 2, simplicial learning through an extension of dictionary learning can be thought as the utmost generalization of the graph-based domain, in which vertices of a hypergraph can now move freely in space, taking the form of a simplicial. Note that in the present formulation, the targeted model is not necessarily a simplicial complex which is a much stricter construct that prohibits self-intersections [12]. The term simplicial refers here to an arbitrary union of simplices in the most general sense.

By definition, an inner-skeleton method seeks a low and possibly heterogeneous dimensional piecewise linear model that expresses the data well in a compact manner. Most importantly, the dictionary learning concept can be categorized as an inner-skeleton method. However, the skeleton attained is not bounded in space but rather an infinite one, where each infinite linear bone is connected to all others at the origin. Technically speaking, a bone corresponds to a linear subspace of arbitrary dimensions. This conception will be indeed helpful when dictionary learning is considered within a multi-class classification framework. In its traditional multi-class formulations, the sparse representation based classifier models a separate dictionary for each distinct class through a data fidelity term together with an ℓ_p-norm regularization constraint on sparse codes ($p = 0$ or 1 in general). Later, the test data is encoded sparsely and classified accordingly favoring the most reconstructive or representative dictionary [13]. In the absence of other modifications, this form of sparse representation based classifier is known to be generative-only. The generative type approaches can create natural random instances of a class, in contrast to discriminative-only methods which focus on decision boundaries between classes.

In a simplistic manner, one can draw parallels between inner-skeleton and generative formulations which discard the existence of other classes; on the other hand, also between decision-boundary and discriminative approaches which need the existence of opposing classes. Not surprisingly, a method can be both generative and discriminative at the same time. Discrimination, in this sense, rises from the fact that while learning a dictionary (or a model) for a class, the data points from other classes are also taken into consideration, i.e., distance to those other points are to be maximized. Some examples of discriminative dictionary learning methods can be given as [14,15].

There is a subtle but crucial point that goes unnoticed in sparse representation based classifier applications and this forms the backbone of the proposed study in this paper. Corresponding to this upcoming point, XOr problem of neural networks dictates that a single layer perceptron is not capable of separating XOr inputs as only a single linear decision boundary is at hand. This has paved way to multilayer formulations that can solve linearly non-separable cases. A similar problem haunts dictionary learning methods silently. Consider the case as demonstrated in Fig. 1, in which there are two classes of digit 8. “Pale class” includes pale images, while “Bright class” contains exactly the same images but they are brightened up. In technical terms, there are two opposing

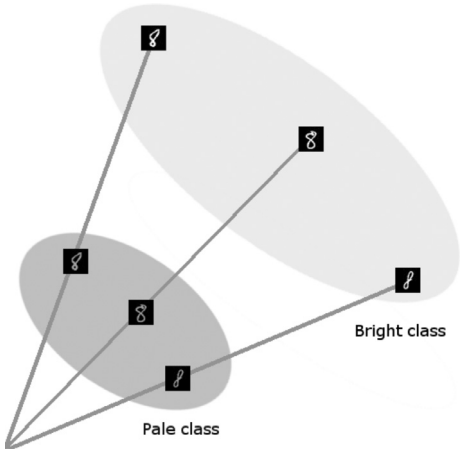

Fig. 1. Conventional dictionary learning is incapable of distinguishing intensity/magnitude, or more technically two classes within the same subspace.

classes lying on the same subspace in the eyes of linear dictionary learning methods. No matter how much discriminative they are, traditional techniques will be incapable of totally distinguishing these two classes. In other words, dictionary learning in its conventional form is insensitive to intensity/magnitude and it will never be able to solve problems requiring intensity/magnitude distinction.

This study proposes a new dictionary learning framework for sparse representations through simplicials. While adapting conventional optimization constraints on sparse codes, the developed evolutionary simplicial learning algorithm leads to a strong generative approach. Experimental validation on different classification tasks demonstrates that this generative-only structure can successfully distinguish two different classes lying on the same subspace as an advantage, while there exist some shortcomings when its discriminative power is under consideration. Achieving state-of-the-art performance in most cases is highly possible through further modifications with discriminative elements. The remaining part of this paper is organized as follows. Section 2 introduces the basic concepts and mathematical foundations of simplicial learning as an extension to classical dictionary learning for sparse representations. Then, Section 3 details the proposed simplicial learning algorithm by adopting an evolutionary approach with the appropriate fitness function to the problem. Section 4 later reports experimental simulations over several datasets and illustrates the obtained results in different classification tasks. Finally, Section 6 briefly concludes this study together with possible considerations which can be adapted to strengthen both theoretical and application aspects of the proposed framework.

2. Simplicial learning: an extension of dictionary learning

2.1. Definitions

Dictionary learning optimization in Eq. (1) basically tries to fit a union of subspaces to the data. Such subspaces are indeed infinite-extent and all crossing the origin without offsets, designated by the dictionary elements usually referred to as *atoms*. Simplicial learning as an adaptation of dictionary learning aims instead at fitting bounded generic piecewise linear objects to the data. Table 1 considers certain bounded generic piecewise linear objects. There are many not-equivalent formal definitions of the first construct, namely a *polytope* to be discussed. This study strictly sticks with the definition that “a polytope is an intact object which admits a simplicial decomposition.” Hence, a polytope is made up of one or

Table 1
Distinctions between the terms for generic objects.

	May not be intact	Piecewise linear	Heterogeneous dimensionality	Arbitrary intersections
Polytope	✗	√	?	√
Simplicial complex	√	√	√	✗
Simplicial	√	√	√	√

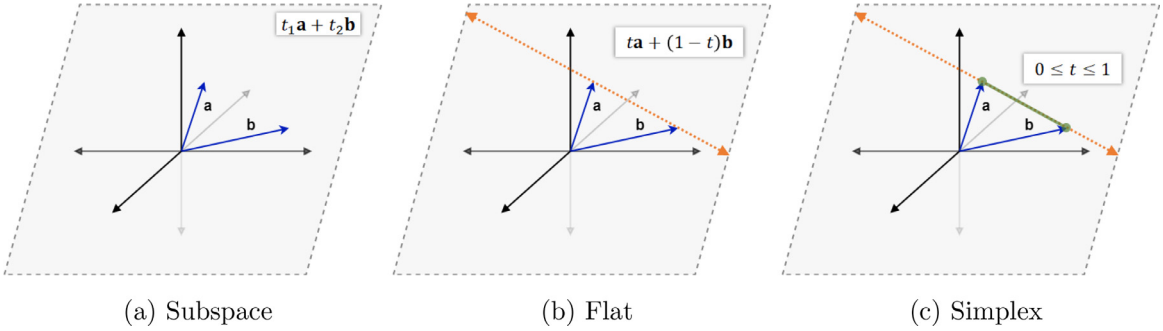

Fig. 2. A simple example of how additional constraints on sparse codes affect the solution of sparse representations. (a) The conventional sparsity constraint together with (b) sum-to-one ($t_1 + t_2 = 1$) and (c) sum-to-one and non-negativity ($t_1 + t_2 = 1$ and $t_1, t_2 \geq 0$) constraints.

more simplices, whereas it is still in question that such simplices can be of different dimensions.

There are two possible ways to generalize the concept of polytope. In the first generalization, connectedness can be discarded leading to the fact that there is not a single object but multiple objects being considered at the same time. The second one allows the building-blocks namely simplices to have different dimensions, thus leading to heterogeneously dimensional objects. A formal name for such union of simplices is a *simplicial complex*, but restricted self-intersections are imposed for a rigorous treatment. By definition, a simplicial complex is a set of simplices satisfying the following two conditions: (*i*) every face of a simplex from this set is also in this set and (*ii*) the non-empty intersection of any two simplices is a face of these two simplices. Losing a bit of formalism, utmost flexibility can be reached by allowing such objects to intersect each other and themselves in arbitrary ways, and such final construct is simply named as a *simplicial* in the remaining part of this paper, to refer to an arbitrary union of simplices in the most general sense. For a more rigorous treatment of these definitions and related concepts, readers might refer to [16].

2.2. Related work

Simplex and simplicial complex based data applications are becoming popular in literature as data analysis receives more and more topological considerations [17–21]. Moreover, utilizing simplices for data applications is not a completely new idea from the perspective of sparse representations [22,23]. Quite similarly, in this study an adaptation of sparse representations framework is chosen that casts a union of subspaces to a union of simplices. A rigorous mathematical formulation is detailed in the following.

2.3. Mathematical formulation

There are three necessary modifications to make a successful transition from the traditional dictionary learning formulation to simplicial learning. First of all, an additional sum-to-one constraint is needed on the sparse codes as noted in Eq. (2) as follows,

$$\arg\min_{\mathbf{A},\{\mathbf{x}_i\}} \sum_i \|\mathbf{y}_i - \mathbf{A}\mathbf{x}_i\|_2^2 \quad \text{subject to} \quad \|\mathbf{x}_i\|_0 \leq q \ \wedge \ \mathbf{1}^{\mathrm{T}}\mathbf{x}_i = 1, \ \forall i, \tag{2}$$

where $\mathbf{1}$ denotes the column vector of ones, of appropriate size with the sparse vectors $\mathbf{x}_i$. Such modification casts q-dimensional subspaces into $(q-1)$-dimensional flats, a flat being a $(q-1)$-subspace with an arbitrary offset. A geometric explanation is illustrated in Fig. 2(a-b) for the case when $q = 2$. In this example, a subspace solution (i.e., an infinite-extent plane) of sparse representations is indeed reduced into a flat (i.e., an infinite-extent line) with an additional sum-to-one constraint on sparse codes.

In addition to above constraint, the second necessary modification is an additional non-negativity on sparse codes as noted in Eq. (3) as follows,

$$\arg\min_{\mathbf{A},\{\mathbf{x}_i\}} \sum_i \|\mathbf{y}_i - \mathbf{A}\mathbf{x}_i\|_2^2 \quad \text{subject to} \quad \|\mathbf{x}_i\|_0 \leq q \ \wedge \ \mathbf{1}^{\mathrm{T}}\mathbf{x}_i = 1 \ \wedge \ \mathbf{0} \leq \mathbf{x}_i, \ \forall i, \tag{3}$$

where $\mathbf{0}$ denotes the column vector of zeros, of appropriate size with the sparse vectors $\mathbf{x}_i$. Together with sum-to-one constraint, sparse codes are now restricted to $[0-1]$ range in magnitude and thus represented flat as an infinite-extent line turns into a simplex (i.e., a bounded line, line segment) as apparent in Fig. 2(b-c) for $q = 2$. In the most generic sense, a simplex can be regarded as a bounded flat.

Note here that there is not any structural constraint on the sparse code patterns for the optimization problems in Eqs. (1)–(3). In other words, all possible q-combinations of dictionary atoms are available for a q-sparse vector solution $\mathbf{x}_i$. Since most of these combinations are unnecessary for a given overcomplete dictionary, keeping a set of possible valid combinations (i.e., forcing certain patterns in sparse codes) will provide a more efficient and more compact representation. This finally leads to the concept of *structured sparsity*, or *group sparsity* in exact terms [24,25], as a last modification on the road to simplicial learning.

While referring back to Section 1, when positional information is removed from a simplicial, the structure left then corresponds to a hypergraph, in which a hyperedge refers to a specific simplex within the simplicial. In relation to group sparsity, a hyperedge exactly corresponds to a group of atoms, hence a valid pattern of sparse codes. As a consequence, a set of groups/hyperedges, or more technically a hypergraph data structure needs to be kept to define the shape of the simplicial. This hypergraph structure will be denoted as $\mathcal{H} = \{h_j\}$ where h_j designates the jth hyperedge referring to jth simplex within the simplicial. In accordance with this

definition, simplicial learning with a structure imposed by $\mathcal{H}$ can be formulated in Eq. (4) as follows,

$$\begin{aligned}&\underset{\mathbf{A},\{\mathbf{x}_i\},\{h_j\}}{\arg\min} \textstyle\sum_i \|\mathbf{y}_i - \mathbf{A}\mathbf{x}_i\|_2^2 \;\; \text{subject to} \\ &\|\mathbf{x}_i\|_0 \le q_j^* \;\wedge\; \mathbf{1}^{\mathrm{T}}\mathbf{x}_i = 1 \;\wedge\; \mathbf{0} \le \mathbf{x}_i \;\wedge\; \left(k \notin h_j^* \rightarrow \mathbf{x}_i^k = 0,\;\; \forall k\right),\;\; \forall i,\end{aligned} \tag{4}$$

where h_j^* is the hyperedge indexing the closest simplex for the data point $\mathbf{y}_i$, $q_j^* = |h_j^*|$ denotes the dimension of that simplex, and the $\left(k \notin h_j^* \rightarrow \mathbf{x}_i^k = 0,\;\; \forall k\right)$ constraint ensures the group sparsity such that only the optimal group (i.e., hyperedge referring to the closest simplex) in $\mathbf{x}_i$ is to be filled and other entries which are represented as $\mathbf{x}_i^k$ shall all be zero. Note here that groups can be not only overlapping but also of different sizes, hence leading to heterogeneous dimensionality. In this final form, $\mathcal{H}$ needs to be learned together with $\mathbf{A}$ but a further careful consideration is needed over the compactness of the simplicial in return.

In summary, as is, the optimization in Eq. (4) is highly ill-posed since there is no restriction on the number of simplices to be used or the dimensions of those simplices. One could even choose a very high-dimensional simplicial construct and zero-out the approximation error easily. Therefore, additional penalty terms need to be investigated based on the number and the dimensionality of simplices for a compact solution. Such a challenge appears to be highly combinatorial in nature and an evolutionary approach can be adopted after a careful consideration of an appropriate fitness function, as described and detailed in Section 3.

3. Evolutionary approach

To obtain an optimal or a suitable simplicial in a heuristic manner, certain number of simplicials are to compete against each other on instances of the same dataset. Basically, an evolutionary approach includes a suitable fitness function to guide this search process, and sub-procedures such as *mutations* and *breeding* to perform the actual search.

3.1. The fitness function

There are certain critical points to be carefully considered before designating the fitness function for the defined problem in this study. First of all, a straightforward optimization procedure for the number and the dimensionality of simplices will not be enough to attain a compact model. For example, consider that the data is distributed in the shape of a triangle with certain area. In this case, a triangle with the most compact area should be preferred as a targeted model. However, one could fit a triangle to this data with correct angles but excessive area. In such a case the dimensionality or the number of simplices indeed do not change. In conclusion, one needs also to take the area (or volume), or more technically the content of the simplices, besides considering the number and the dimensionality of simplices. When the simplex is of dimension 2 (namely a triangle), the content is called the area, when the simplex is 3 dimensional (namely a tetrahedron), the content refers to the volume. Therefore, the term "content" generalizes area and volume concepts to higher dimensions.

The content of an arbitrary simplex can be calculated using Cayley-Menger determinant [26]. Let K be a q-dimensional simplex in $\mathbb{R}^N$, and $\mathbf{B}$ denote $(q+1)\times(q+1)$ distance matrix of vertices $\{v_1, v_2, \ldots, v_{q+1}\}$ such that $\mathbf{B}_{ik} = \|v_i - v_k\|_2^2$. Then the content C_K of K is given in a relation in Eq. (5) as follows,

$$C_K^2 = \frac{(-1)^{q+1}}{2^q (q!)^2} det(\hat{\mathbf{B}}), \tag{5}$$

where $\hat{\mathbf{B}}$ is $(q+2)\times(q+2)$ matrix obtained from $\mathbf{B}$ by bordering it with a top row of $(0, 1, \ldots, 1)$ and a left column of $(0, 1, \ldots, 1)^{\mathrm{T}}$.

Related with the content calculation here, another issue arises because of the allowed heterogeneous dimensionality in the optimization formula. The content of a line-segment (as an object) and a triangle (as an object) are incomparable in a general continuous setting since a triangle contains infinitely-many line-segments itself. To resolve this problem, an exponential term is introduced through *an approximated cumulative discrete content* calculation of a simplicial as given in Eq. (6) as follows,

$$\sum_{j=1}^{|\mathcal{H}|} (1 + C_j)^{q_j}, \tag{6}$$

where $|\mathcal{H}|$ denotes the number of hyperedges or equivalently the number of simplices, C_j is the content of the j^{th} simplex and q_j is the dimension of that simplex. As a content $C_j < 1$ would complicate the exponentiation used, $\left(1 + C_j\right)$ is needed in the discrete approximation.

Having pinned down the above term which will be a component in the fitness function driving the evolutionary process, a fitness function candidate (in a minimization form) is given in Eq. (7) as follows,

$$\sum_i \|\mathbf{y}_i - \mathbf{A}\mathbf{x}_i\|_2^2 + \alpha \sum_j (1 + C_j)^{q_j}, \tag{7}$$

where sum of squared error (SSE) used as the data fidelity term and approximated cumulative discrete content as to regulate the compactness of the representation. α denotes the regularization parameter controlling the contribution of the compactness prior on the solution.

While initially experimenting with the above fitness function, it is observed that the parameter α has a very broad optimality range, which changes drastically from dataset to dataset. This is due to the fact that there is a high dynamic range imbalance between two cumulative terms. Therefore, a variant of the defined fitness function is considered by transforming Eq. (7) into the logarithmic scale in order to compress the dynamic range, leading to a more natural maximization setting formulated in Eq. (8) as follows,

$$\frac{\log_{10}\left(\frac{m}{\sum_i \|\mathbf{y}_i - \mathbf{A}\mathbf{x}_i\|_2^2}\right)}{1 + \beta \log_{10}\left(\gamma + \sum_j (1 + C_j)^{q_j}\right)}, \tag{8}$$

where m denotes the number of data points and the parameter β regulates over- or under-fitting. When $\beta = 0$, the fitness function simply reduces to the data fidelity term favoring only for the reconstruction quality. Instead, a high β value forces the simplicial to be compact. Empirical investigations suggest that a β value around 0.05 could be a global setting as it provides excellent results over all datasets considered in this study. Note that there might be no simplices at certain times of the evolution process. This would erroneously lead the sum of content to be zero, thus logarithm to be infinity. The parameter γ eliminates this possibility by fixing its value to 10. Hence, this parameter forces the lower logarithm to evaluate at least to value 1.

3.2. Mutations and breeding

First of all, it is important to note here that the hypergraph $\mathcal{H}$ is kept in the form of an incidence matrix of zeros and ones, where the row count corresponds to the number of simplices and the column count matches to the number of vertices or rather the number of atoms (columns) in the dictionary $\mathbf{A}$. Mutations can be easily applied on this binary matrix. In detail, there are four main

processes that provide the background for evolution: (*i*) increasing/decreasing the dimension of a simplex, (*ii*) adding/removing a simplex to/from the hypergraph, (*iii*) subdividing a simplex and (*iv*) adding/removing a vertex to/from the dictionary. All of these mutation operations are performed randomly without any optimality consideration.

Algorithm 1: Breeding algorithm.

1: $(\mathcal{H}_1, \mathbf{A}_1) \leftarrow$ get_structure(S_1)
2: $(\mathcal{H}_2, \mathbf{A}_2) \leftarrow$ get_structure(S_2)
3: $\mathcal{H}_a \leftarrow$ a random submatrix of $\mathcal{H}_1$
4: $\mathbf{A}_a \leftarrow$ the submatrix of $\mathbf{A}_1$ corresponding to $\mathcal{H}_a$
5: $\mathcal{H}_b \leftarrow$ a random submatrix of $\mathcal{H}_2$
6: $\mathbf{A}_b \leftarrow$ the submatrix of $\mathbf{A}_2$ corresponding to $\mathcal{H}_b$
7: $\mathbf{A}_{new} \leftarrow \begin{bmatrix} \mathbf{A}_a & \mathbf{A}_b \end{bmatrix}$
8: $\mathcal{H}_{new} \leftarrow \begin{bmatrix} \mathcal{H}_a & 0 \\ 0 & \mathcal{H}_b \end{bmatrix}$
9: $S_{new} \leftarrow (\mathcal{H}_{new}, \mathbf{A}_{new})$

As an additional tool to assist the searching process, breeding of two simplicials is also undertaken in which both dictionary elements and hypergraph structures of those two simplicials are split and then merged appropriately in order to create a new simplicial representative of two parents up to certain extent. Details of the breeding procedure are depicted in Algorithm 1. At first, hypergraph structures and the corresponding dictionary elements are extracted for these two simplicials $S_1 = (\mathcal{H}_1, \mathbf{A}_1)$ and $S_2 = (\mathcal{H}_2, \mathbf{A}_2)$. Then random submatrices $\mathcal{H}_a \in \mathcal{H}_1$ and $\mathcal{H}_b \in \mathcal{H}_2$ from each hypergraph are attained together with the corresponding columns of these dictionaries, contained in matrices $\mathbf{A}_a \in \mathbf{A}_1$ and $\mathbf{A}_b \in \mathbf{A}_2$. While vertices (atoms) are directly concatenated in $\mathbf{A}_{new}$ (line 7), hypergraphs are concatenated in a disjoint manner in $\mathcal{H}_{new}$ (line 8). In short, two subsimplicials are extracted and then grouped together in a disjoint manner to form a new simplicial S_{new}. Such tool can be suitably employed to exploit the underlying dimensionality of the dataset since these splitting and merging processes may lead child simplicials to acquire a properly representative data-dimensionality in a very fast manner, much faster than mutation processes to perform alone. Therefore, as a general observation, breeding determines the core dimensionality of the simplicial and mutations fine-tune the simplicial to the data. However, sufficiently high dimensional simplicials should be employed in the initialization stage for breeding to determine the core dimensionality.

3.3. Implementation details

The algorithm to learn an evolutionary simplicial model on a set of data points $\{\mathbf{y}_i\}_{i=1}^n$ stored in the columns of a data matrix $\mathbf{Y}$ is given in Algorithm 2 . At first, the initial simplicial is to be generated from the given data points (line 1). It is observed that choosing a single point (i.e., centroid of the dataset) as an initial simplicial is sufficient for low-dimensional problems. Through mutations and breeding processes, the initial simplicial takes an appropriate form in a fast manner since the search space is relatively small. However, a procedure involving the *k*-means algorithm [27] as a subroutine is employed to designate the initial simplicial for high-dimensional problems. In such cases, starting from a single point greatly slows down the process of evolution since the search space is quite large. Hence, an initialization based on *k*-means ensures that the starting simplicial is already a relatively fit one. A last point worth mentioning related to initialization here is that the initial simplicial S should satisfy the condition that the numerator of Eq. (8) is positive, i.e., $\sum_i \|\mathbf{y}_i - \mathbf{A}\mathbf{x}_i\|_2^2 < m$ to lead a meaningful evolution.

Algorithm 2: Evolutionary simplicial learning (ESL) algorithm.

1: $pop \leftarrow$ init_pop$(\mathbf{Y})$
2: **while** *not converged* **do**
3: $pop \leftarrow$ mutations(pop)
4: $pop \leftarrow$ breeding(pop)
5: **for all** S in pop **do**
6: $\mathbf{X} \leftarrow$ sparse_coding$(\mathbf{Y}, S)$
7: $\mathbf{A} \leftarrow$ dictionary_update$(\mathbf{Y}, \mathbf{X})$
8: $F \leftarrow$ fitness$(\mathbf{A}, \mathcal{H})$
9: **end for**
10: $pop \leftarrow$ sort and choose based on F values
11: **end while**
12: $S_{best} \leftarrow pop(1)$

On line 6, the algorithm performs the projection of data points $\{\mathbf{y}_i\}$ in $\mathbf{Y}$ onto each simplex of the simplicial S [28,29] which basically corresponds to the sparse coding optimization. The closest simplex for the data point $\mathbf{y}_i, \forall i$, is determined through the minimum approximation error acquired after projecting $\mathbf{y}_i$ onto each simplex. The positive barycentric coordinates of the projection points corresponding to the sparse codes are acquired, and then the necessary spots of the sparse representation matrix $\mathbf{X}$ is filled accordingly.

On line 7, dictionary matrix $\mathbf{A}$ is updated using a direct least-squares solution. To optimize $\arg\min_{\mathbf{A}} \|\mathbf{Y} - \mathbf{A}\mathbf{X}\|_F^2$ by forcing its derivative to zero, the analytic solution is obtained with $\mathbf{A} = \mathbf{Y}\mathbf{X}^+$ where $\mathbf{X}^+$ represents Moore-Penrose pseudo-inverse of $\mathbf{X}$. Note that there is no evolutionary process for learning $\mathbf{A}$, namely the vertices of the simplicial S. Instead, vertices are updated once exactly on this line at each iteration of the algorithm.

Finally, the surviving simplicials are determined based on the fitness scores they attain (line 10). Experimental trials suggest that keeping the population size at 10 is an efficient strategy, while an iteration count of 5 is sufficient instead of a full convergence. Notice here that the parent simplicials are to be kept in the population pool when their fitness scores are higher than their children's.

4. Experimental results

The proposed method is tested in two phases of experiments to evaluate its classification capabilities. In the first experimental setup, the performance is evaluated in a one-class classification task for outlier detection. Datasets contain certain degree of outliers in such outlier detection problems, and methods learn models –agnostic of data labels– in an unsupervised manner. In the second classification task, the performance of the proposed method is evaluated in a multi-class setting. At this stage, seven synthetic multi-class datasets are generated in addition to two handwritten digit recognition datasets. The synthetic datasets are special in that they contain cases which require intensity/magnitude distinction, especially very challenging for conventional dictionary learning methods.

All experiments are performed on an Intel(R) Core(TM) i7 – 6700HQ CPU @ 2.60 GHz 16 GB RAM machine running on Microsoft Windows 10. Benchmark of outlier detection dataset named PyOD [30] is run with Python 3.6 and the proposed ESL algorithm is implemented using Matlab 2014a on the same machine. All multi-class experiments are carried out on Matlab 2014a. DICTOL as the part of LRSDL project [31] is utilized for the implementations of other dictionary learning methods.

Table 2
Information regarding the datasets used in outlier detection experiments.

Dataset	#Samples	#Dimensions	Outlier ratio (%)
arrhythmia	452	274	14.6018
cardio	1831	21	9.6122
glass	214	9	4.2056
ionosphere	351	33	35.8974
letter	1600	32	6.2500
lympho	148	18	4.0541
mnist	7603	100	9.2069
musk	3062	166	3.1679
optdigits	5216	64	2.8758
pendigits	6870	16	2.2707
pima	768	8	34.8958
satellite	6435	36	31.6395
satimage-2	5803	36	1.2235
shuttle	49, 097	9	7.1511
vertebral	240	6	12.5000
vowels	1456	12	3.4341
wbc	378	30	5.5556

4.1. Outlier detection

In total 17 benchmark datasets are taken from ODDS Library [32] for the one-class learning task. Information regarding these datasets in terms of number of samples, sample dimensionality and outlier percentages is summarized in Table 2 and interested readers might refer to [32] for details about each individual dataset. Using these benchmark datasets, a random 60% to 40% train-test set split is repeated for 10 independent simulations and the mean Area Under The Curve (AUC) Receiver Operating Characteristics (ROC) results are reported in Table 3.

The proposed Evolutionary Simplicial Learning (ESL) method is evaluated against an extensive outlier detection benchmark named as PyOD [30]. The competing methods include Angle-based Outlier Detector (ABOD) [33], Clustering-based Local Outlier Factor (CBLOF) [34], Feature Bagging (FB) [35], Histogram-based Outlier Score (HBOS) [36], Isolation Forest (IForest) [37], K Nearest Neighbors (KNN) [38], Local Outlier Factor (LOF) [39], Minimum Covariance Determinant (MCD) [40], One-class Support Vector Machine (OCSVM) [41] and Principal Component Analysis (PCA) [42] and one of the most recent results obtained in Weng et al. [43] on the same benchmark (with an average of 20 runs for each dataset).

Last two rows of Table 3 illustrate the mean AUC ROC results over all datasets and their standard deviations. ESL not only presents the best average AUC ROC performance among all methods in the benchmark but also has the least standard deviation. One can conclude that it is the most reliable methods among considered techniques for this performance measure. Moreover, ESL shows top AUC ROC performance in three datasets. However, additional tests show that it does not have a noticeable advantage in Precision at n (P@n) performance.

4.2. Multi-class classification

For the multi-class classification task, six challenging synthetic datasets are generated by following the procedures in noa [44] and these datasets are depicted in Fig. 3. Four of these datasets (namely, *Cluster-in-Cluster; Two-Spirals; Half-Kernel* and *Crescent&Full-moon*) contain binary classification tasks while the remaining two of them (*Corners* and *Outliers*) consist of four-class classification problems. In addition, a synthetically altered dataset (named as MNIST8) is included in the experimental setup, in which all samples of the digit 8 from the original MNIST [45] are designated as the “Bright class” while a new “Pale class” is generated from all these original samples by dimming with a scale of 0.25 according to the previous discussion related to Fig. 1.

The proposed ESL algorithm in this setup is compared against Sparse Representation-based Classification (SRC) [46], Label Consistent K-SVD (LCKSVD1 and LCKSVD2) [15], Dictionary Learning with Structured Incoherence (DLSI) [47], Fisher Discrimination Dictionary Learning (FDDL) [48], Dictionary Learning for Commonality and Particularity (DLCOPAR) [49] and Low-rank Shared Dictionary Learning (LRSDL) [31,50]. Experimental results in terms of classification success rates are presented in Table 4. It is apparent that ESL easily outperforms all considered dictionary learning methods over all cases. This should not be a surprising result since all utilized synthetic datasets require intensity/magnitude distinction to various extents. On the other hand, some discriminative methods such as LCKSVD2, FDDL and LRSDL undergo meaningful learning (i.e., better than random) over some datasets. This observation leads to an important conclusion that discriminative modifications may alleviate insensitivity to intensity to a certain degree.

Fig. 3 depicts examples of learned simplicial models on six synthetic datasets. As it can be observed clearly, simplicials are bounded and they are composed of simplices (i.e., points and line-segments in these cases) with arbitrary offsets, providing an advantage over unbounded and without-offset dictionary learning models in all these classification tasks.

Digit Classification: In most of the practical pattern recognition applications, the pattern or rather the direction of the feature vector utilized plays an important role on the success rate. For instance, a “star pattern” is a “star pattern” no matter how much bright or pale it is. Therefore, the advantage of simplicial learning over dictionary learning is expected to diminish in some real-world applications. This is observable in digit classification experiments featuring USPS [51] and MNIST datasets as reported in Table 5. In this set of experiments, ESL is compared to classification methods including Supervised Dictionary Learning [14] with generative training (SDL-G) and with discriminative learning (SDL-D), Task-driven Dictionary Learning [52]: unsupervised (TDDL-G) and supervised (TDDL-D), FDDL, KNN, Gaussian SVM, Locality-constrained Linear Coding (LLC) [53] and Locality-sensitive Dictionary Learning (LDL) [13]. LLC and LDL methods have the sum-to-one constraint on sparse codes, therefore they learn spaces with arbitrary offsets but learned models are still not bounded (without the non-negativity constraint).

As apparent from Table 5, ESL appears to be a successful generative-only method which performs nearly at the capacity of Gaussian SVM (i.e., a well-known and widely used discriminative classifier). However, it cannot outperform discriminative dictionary learning methods such as FDDL and TDDL-D in these datasets. A final note is that ESL can also be modified through discriminative elements. Discriminative methods SDL-D and TDDL-D have a 1.5–2% advantage over their generative counterparts SDL-G and TDDL-G. Hence, a successful discriminative version of ESL can then be projected to reach state-of-the-art, an estimation open to discussion or further investigation.

5. Computational complexity

Let us first dissect the loop starting on line 5 and ending on line 9 in Algorithm 2 since this part mainly determines the time complexity. On line 6, each data point $\mathbf{y}_i$ is projected onto each simplex within the simplicial $\mathcal{S}$ and then assigned to the closest one. There are $|\mathcal{H}|$ simplices within a simplicial, namely the number of hyperedges in the corresponding hypergraph. An efficient projection onto a single simplex is claimed to have a time complexity of $\mathcal{O}(n)$, n is the dimension of data space [54]. In a sensible model, there must be at most one simplex for each data point, resulting in a bound $|\mathcal{H}| \leq m$ and m is the number of data points.

Table 3
Results from 10 independent simulations for outlier detection on various datasets. While the top value in each cell is the mean AUC ROC result, the bottom value is the computation time in seconds.

Dataset	ABOD	CBLOF	FB	HBOS	IForest	KNN	LOF	MCD	OCSVM	PCA	[43]	ESL
arrhythmia	0.769	0.784	0.778	0.822	0.801	0.786	0.779	0.779	0.781	0.782	0.801	**0.826**
	0.31s	0.34s	0.68s	0.29s	0.49s	0.11s	0.09s	3.82s	0.05s	0.14s	–	8.51s
cardio	0.569	0.928	0.587	0.835	0.921	0.724	0.574	0.814	0.935	0.950	**0.969**	0.884
	0.46s	0.16s	0.97s	0.01s	0.42s	0.19s	0.12s	1.61s	0.09s	0.01s	–	36.19s
glass	0.795	0.850	0.873	0.739	0.757	0.851	0.864	0.790	0.632	0.675	–	**0.876**
	0.04s	0.05s	0.04s	0.01s	0.31s	0.01s	0.01s	0.06s	0.01s	0.01s	–	4.70s
ionosphere	0.925	0.813	0.873	0.561	0.850	0.927	0.875	**0.956**	0.842	0.796	0.911	0.851
	0.07s	0.07s	0.08s	0.01s	0.33s	0.02s	0.01s	0.35s	0.01s	0.01s	–	7.09s
letter	**0.878**	0.507	0.866	0.593	0.642	0.877	0.859	0.807	0.612	0.528	–	0.776
	0.42s	0.14s	0.88s	0.01s	0.42s	0.16s	0.11s	5.77s	0.08s	0.01s	–	22.63s
lympho	0.911	0.973	0.975	**0.996**	0.994	0.975	0.977	0.900	0.976	0.985	0.987	0.984
	0.03s	0.05s	0.04s	0.01s	0.31s	0.01s	0.01s	0.11s	0.01s	0.01s	–	2.72s
mnist	0.782	0.801	0.721	0.574	0.816	0.848	0.716	0.867	0.853	0.853	**0.929**	0.803
	8.61s	1.50s	55.79s	0.08s	2.24s	7.79s	7.43s	14.14s	4.91s	0.20s	–	171.54s
musk	0.184	0.988	0.526	**1.000**	**1.000**	0.799	0.529	**1.000**	**1.000**	**1.000**	**1.000**	0.972
	2.61s	0.52s	14.49s	0.08s	1.52s	2.05s	1.93s	57.93s	1.27s	0.25s	–	77.48s
optdigits	0.467	0.509	0.443	**0.873**	0.725	0.371	0.450	0.398	0.500	0.509	–	0.746
	2.96s	0.61s	14.60s	0.04s	1.24s	2.11s	1.94s	6.94s	1.45s	0.08s	–	103.72s
pendigits	0.688	0.949	0.460	0.924	0.944	0.749	0.470	0.834	0.930	0.935	0.938	**0.951**
	1.71s	0.37s	4.44s	0.01s	0.72s	0.71s	0.66s	4.82s	0.99s	0.02s	–	91.98s
pima	0.679	**0.735**	0.624	0.700	0.681	0.708	0.627	0.675	0.622	0.648	–	0.626
	0.15s	0.09s	0.12s	0.01s	0.34s	0.04s	0.01s	0.09s	0.01s	0.01s	–	13.19s
satellite	0.571	0.669	0.557	0.758	0.702	0.684	0.557	**0.803**	0.662	0.599	0.750	0.705
	2.11s	0.63s	8.52s	0.03s	1.01s	1.23s	1.14s	9.13s	1.41s	0.04s	–	105.03s
satimage-2	0.819	0.992	0.457	0.980	0.995	0.954	0.458	0.996	**0.998**	0.982	0.976	0.995
	1.91s	0.52s	6.52s	0.02s	0.80s	0.99s	0.87s	8.94s	1.14s	0.04s	–	83.80s
shuttle	0.623	0.627	0.472	0.986	**0.997**	0.654	0.526	0.990	0.992	0.990	0.994	0.992
	17.36s	1.38s	70.16s	0.03s	3.07s	10.12s	13.85s	16.11s	50.88s	0.05s	–	428.03s
vertebral	0.426	0.349	0.417	0.326	0.391	0.382	0.408	0.391	0.443	0.403	**0.580**	0.413
	0.05s	0.06s	0.04s	0.01s	0.30s	0.01s	0.01s	0.06s	0.01s	0.01s	–	4.96s
vowels	0.961	0.586	0.943	0.673	0.759	**0.968**	0.941	0.808	0.780	0.603	–	0.881
	0.31s	0.11s	0.34s	0.01s	0.38s	0.09s	0.04s	1.40s	0.04s	0.01s	–	21.31s
wbc	0.905	0.923	0.933	**0.952**	0.931	0.937	0.935	0.921	0.932	0.916	–	0.924
	0.08s	0.08s	0.09s	0.01s	0.32s	0.02s	0.01s	0.33s	0.01s	0.01s	–	6.03s
MEAN	0.703	0.764	0.677	0.782	0.818	0.776	0.679	0.808	0.793	0.774	n/a	**0.835**
STDEV	0.210	0.195	0.203	0.192	0.164	0.182	0.199	0.179	0.183	0.197	n/a	**0.152**

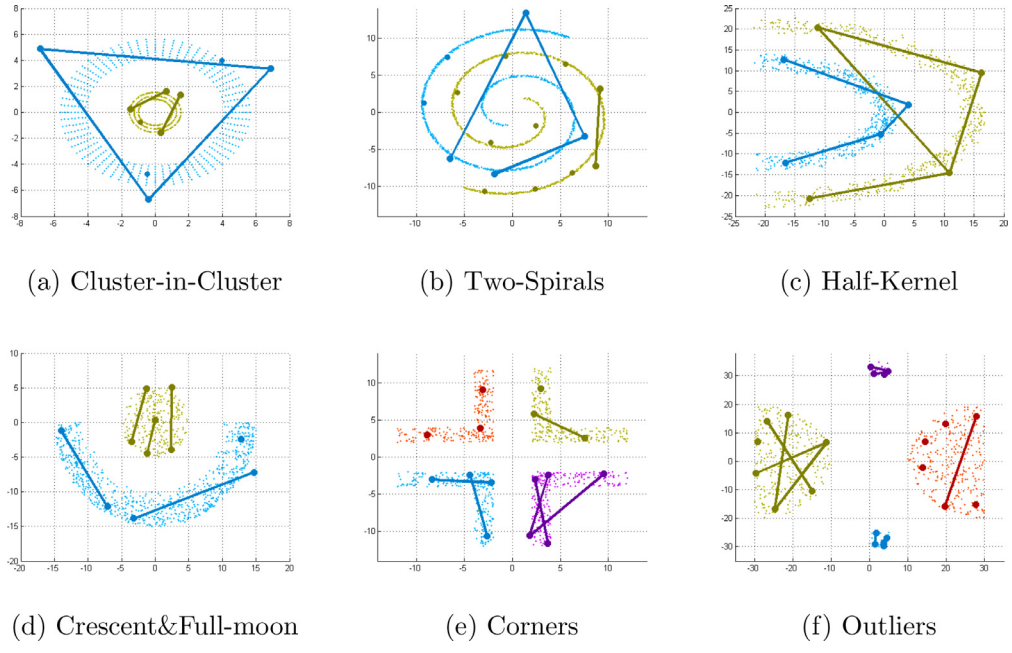

Fig. 3. Examples of learned simplicial models on six synthetic datasets. Best visualized in color.

Therefore, complexity of the sparse coding phase is $\mathcal{O}(m^2 n)$. On line 7, dictionary update is performed by Moore-Penrose pseudo-inverse, having a time complexity of $\mathcal{O}(m^2 v)$ where v denotes the total number of columns (atoms) in the dictionary **A**. Since over-completeness implies $n < v \leq m$, this phase arrives at a complexity of $\mathcal{O}(m^3)$. Lastly, line 8 includes the content calculation for each simplex. Since it involves calculating the determinant of a $(q+2) \times (q+2)$ matrix and q is the dimension of the simplex, the complexity can be given as $\mathcal{O}(\sum_j q_j^3)$ assuming that LU decomposition is employed for the determinant. An important remark here

Table 4
Classification success rates (the top value) and computation time in seconds (the bottom value) of different dictionary learning methods for six synthetic datasets, and for the proposed binary MNIST8 problem (the last row).

Dataset	SRC	LCKSVD1	LCKSVD2	DLSI	FDDL	DLCOPAR	LRSDL	ESL
Cluster-in-Cluster	52.77%	50.99%	54.55%	43.68%	55.53%	67.98%	45.45%	**88.14**%
	2.74s	0.24s	0.24s	3.79s	5.82s	2.07s	33.7s	17.01s
Two-Spirals	49.30%	41.50%	71.30%	52.30%	53.70%	51.10%	59.70%	**80.22**%
	13.26s	0.52s	0.49s	5.97s	6.01s	3.19s	34.84s	33.76s
Half-Kernel	63.80%	64.40%	65.60%	51.60%	58.00%	62.80%	64.80%	**93.65**%
	2.60s	0.25s	0.26s	3.96s	5.98s	2.14s	23.99s	17.32s
Crescent&Full-moon	75.00%	82.60%	78.00%	55.60%	64.40%	64.20%	85.60%	**99.80**%
	2.59s	0.26s	0.25s	3.23s	5.72s	2.04s	24.44s	15.07s
Corners	91.00%	25.00%	44.80%	27.80%	29.60%	29.20%	27.80%	**97.50**%
	2.60s	0.47s	0.39s	8.98s	5.82s	3.84s	37.31s	13.62s
Outliers	51.33%	43.33%	80.00%	52.33%	75.67%	53.33%	99.27%	**100.00**%
	0.59s	0.38s	0.33s	9.08s	5.84s	5.40s	30.78s	8.26s
MNIST8	50.00	50.00%	50.00%	50.00%	75.45%	50.05%	63.24%	**99.05**%
	1746.49s	81.25s	82.14s	61.92s	16.37s	42.31s	661.54s	604.91s

Table 5
Classification error rates of various methods on handwritten digit datasets, USPS and MNIST. ESL appears as a superior generative method, nearly performing at the capacity of discriminative Gaussian SVM on both datasets.

Generative-only						Discriminative				
Dataset	SDL-G	TDDL-G	LLC	LDL	ESL	KNN	SVM-Gauss	SDL-D	FDDL	TDDL-D
USPS	6.67	4.58	4.48	**3.79**	4.31	5.2	4.2	3.54	3.69	**2.84**
MNIST	3.56	2.36	–	–	**1.85**	5.0	1.4	1.05	–	**0.54**

is that $\sum_j q_j^3$ is negligible compared to m^3 and dictionary update is still the most expensive step within the loop. However, having very high dimensional simplices will slow down the algorithm. Note also that sorting, applying mutations and breeding are not computationally expensive when compared against operations within the loop.

Let us now discuss the sensitivity of the algorithm to the ratio between actual dimension of the ambient space and actual inner size of the data. As noted before, mutations alone increase or decrease the dimensions of the model in a relatively slow manner. This is the main reason of breeding which may speed up the algorithm by creating children that are much less dimensional than their parents. Assuming that the starting simplicials have high enough dimensions, the breeding process uncovers the core dimensionality and then mutations will uncover local varieties. In short, it would take a long amount of time to recover the actual inner size of the data if only mutations were being used, but the algorithm can cope with this issue via the breeding process in a more effective way.

As a final note, the complexity of the proposed evolutionary approach is highly related to the population size. Therefore, the population size can be adjusted accordingly to satisfy the computational requirements versus the performance criteria. Moreover, the implemented Matlab code in this study is experimental, hence even larger population sizes can be manageable with more optimized implementations.

6. Discussion and conclusion

Dictionary learning through simplicials is more flexible than classical dictionary learning models since simplices are bounded and freely positioned in space. The proposed sparsity based evolutionary structure, called ESL is highly applicable if the characteristics of the problem at hand requires such successful localized models. In this study, a global fitness function is employed and there is no restriction on the local fitness of each individual simplex within the simplicial. If the local fitness of each simplex is considered and optimized individually, the resulting simplicial model might be in a more compact form. For example, the unnecessary simplex of the green simplicial in Fig. 3(c) would most probably be eliminated as it does not have any local fitness, thus lead to an increased accuracy of classification. Another point worth mentioning here is that the employed fitness function in Eq. (8) is reminiscent of Poisson distribution, in a multidimensional form [55,56]. Hence, other probabilistic considerations and also discriminative elements can be adapted to strengthen both theoretical and application aspects of the proposed framework.

As exemplified in this paper, simplicial learning can successfully address some weak points of conventional dictionary learning for the considered machine learning problems; it is a promising approach inherently capable of performing signal processing tasks and can become a general machine learning tool with many application domains.

Declaration of Competing Interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

CRediT authorship contribution statement

Yigit Oktar: Conceptualization, Methodology, Software, Validation, Formal analysis, Investigation, Writing - original draft, Visualization. **Mehmet Turkan:** Conceptualization, Methodology, Formal analysis, Writing - review & editing, Supervision.

References

[1] M. Elad, M.A.T. Figueiredo, Y. Ma, On the role of sparse and redundant representations in image processing, Proc. IEEE 98 (6) (2010) 972–982.
[2] I. Tosic, P. Frossard, Dictionary learning: what is the right representation for my signal? IEEE Signal Process. Mag. 28 (2011) 27–38.
[3] R. Gribonval, R. Jenatton, F. Bach, M. Kleinsteuber, M. Seibert, Sample complexity of dictionary learning and other matrix factorizations, IEEE Trans. Inf. Theory 61 (6) (2015) 3469–3486.
[4] N. Akhtar, F. Shafait, A. Mian, Discriminative Bayesian dictionary learning for classification, IEEE Trans. Pattern Anal. Mach. Intell. 38 (12) (2016) 2374–2388.
[5] Y. Oktar, M. Turkan, A review of sparsity-based clustering methods, Signal Process. 148 (2018) 20–30.
[6] Y. Oktar, M. Turkan, K-polytopes: a superproblem of k-means, Signal Image Video Process. 13 (6) (2019) 1207–1214.

[7] M.M. Moya, D.R. Hush, Network constraints and multi-objective optimization for one-class classification, Neural Netw. 9 (3) (1996) 463–474.
[8] S.S. Khan, M.G. Madden, One-class classification: taxonomy of study and review of techniques, Knowl. Eng. Rev. 29 (3) (2014) 345–374.
[9] P. Juszczak, D.M.J. Tax, E. Pe-kalska, R.P.W. Duin, Minimum spanning tree based one-class classifier, Neurocomput. 72 (7–9) (2009) 1859–1869.
[10] L. Wei, W. Qian, A. Zhou, W. Jin, X.Y. Jeffrey, Hot: hypergraph-based outlier test for categorical data, in: Pacific-Asia Conf. Know. Discov. Data Mining, 2003, pp. 399–410.
[11] J. Silva, R. Willett, Hypergraph-based anomaly detection of high-dimensional co-occurrences, IEEE Trans. Pattern Anal. Mach. Intell. (3) (2008) 563–569.
[12] S. Barbarossa, S. Sardellitti, E. Ceci, Learning from signals defined over simplicial complexes, in: IEEE Data Sci. W., 2018, pp. 51–55.
[13] C.-P. Wei, Y.-W. Chao, Y.-R. Yeh, Y.C.F. Wang, Locality-sensitive dictionary learning for sparse representation based classification, Pattern Recognit. 46 (5) (2013) 1277–1287.
[14] J. Mairal, J. Ponce, G. Sapiro, A. Zisserman, F.R. Bach, Supervised dictionary learning, in: Adv. Neural Inf. Process. Syst., 2009, pp. 1033–1040.
[15] Z. Jiang, Z. Lin, L.S. Davis, Label consistent K-SVD: learning a discriminative dictionary for recognition, IEEE Trans. on Pattern Anal. Mach. Intell. 35 (11) (2013) 2651–2664.
[16] J.R. Munkres, Analysis on Manifolds, CRC Press, 2018.
[17] C. Luo, C. Ma, C. Wang, Y. Wang, Learning discriminative activated simplices for action recognition, in: AAAI Conf. Artif. Intell., 2017, pp. 4211–4217.
[18] J. Huang, F. Nie, H. Huang, A new simplex sparse learning model to measure data similarity for clustering, in: Int. Joint Conf. Artif. Intell., 2015, pp. 3569–3575.
[19] R.L. Belton, B.T. Fasy, R. Mertz, S. Micka, D.L. Millman, D. Salinas, A. Schenfisch, J. Schupbach, L. Williams, Learning simplicial complexes from persistence diagrams, in: Conf. Comput. Geometry, 2018, p. 18.
[20] H. Tasaki, R. Lenz, J. Chao, Simplex-based dimension estimation of topological manifolds, in: Int. Conf. Patt. Recog., 2016, pp. 3609–3614.
[21] A. Patania, F. Vaccarino, G. Petri, Topological analysis of data, EPJ Data Sci. 6 (1) (2017) 7.
[22] C. Wang, J. Flynn, Y. Wang, A. Yuille, Recognizing actions in 3D using action-snippets and activated simplices, in: AAAI Conf. Artif. Intell., 2016, pp. 3604–3610.
[23] D.K. Nguyen, K. Than, T.B. Ho, Simplicial nonnegative matrix factorization, in: Int. Conf. Comput. Commun. Tech.-Res. Innov. Vis. Fut., 2013, pp. 47–52.
[24] M. Yuan, Y. Lin, Model selection and estimation in regression with grouped variables, J. R. Stat. Soc. B 68 (1) (2006) 49–67.
[25] L. Jacob, G. Obozinski, J.P. Vert, Group lasso with overlap and graph lasso, in: Int. Conf. Mach. Learn., 2009, pp. 433–440.
[26] H.-C. Li, M. Song, C.I. Chang, Simplex volume analysis for finding endmembers in hyperspectral imagery, in: Satellite Data Comp. Commun. Process. XI, volume 9501, 2015, p. 950107.
[27] A.K. Jain, Data clustering: 50 years beyond k-means, Pattern Recognit. Lett. 31 (8) (2010) 651–666.
[28] J. Duchi, S. Shalev-Shwartz, Y. Singer, T. Chandra, Efficient projections onto the l 1-ball for learning in high dimensions, in: Int. Conf. Mach. Learn., 2008, pp. 272–279.
[29] O. Golubitsky, V. Mazalov, S.M. Watt, An algorithm to compute the distance from a point to a simplex, Commun. Comput. Algebra 46 (2012). 57–57
[30] Y. Zhao, Z. Nasrullah, Z. Li, PyOD: a python toolbox for scalable outlier detection, J. Mach. Learn. Res. 20 (2019) 1–7.
[31] T.H. Vu, V. Monga, Fast low-rank shared dictionary learning for image classification, IEEE Trans. Image Process. 26 (11) (2017) 5160–5175.
[32] S. Rayana, ODDS library, 2016, http://odds.cs.stonybrook.edu.
[33] H.-P. Kriegel, M. Schubert, A. Zimek, Angle-based outlier detection in high-dimensional data, in: Int. Conf. Knowledge Discovery Data Mining, 2008, pp. 444–452.
[34] Z. He, X. Xu, S. Deng, Discovering cluster-based local outliers, Pattern Recognit. Lett. 24 (9–10) (2003) 1641–1650.
[35] A. Lazarevic, V. Kumar, Feature bagging for outlier detection, in: Int. Conf. Knowledge Discovery Data Mining, 2005, pp. 157–166.
[36] M. Goldstein, A. Dengel, Histogram-based outlier score (HBOS): a fast unsupervised anomaly detection algorithm, in: KI-2012: Poster and Demo Track, 2012, pp. 59–63.
[37] F.T. Liu, K.M. Ting, Z.H. Zhou, Isolation forest, in: IEEE Int. Conf. Data Mining, 2008, pp. 413–422.
[38] S. Ramaswamy, R. Rastogi, K. Shim, Efficient algorithms for mining outliers from large data sets, in: ACM SIGMOD Record, volume 29, 2000, pp. 427–438.
[39] M.M. Breunig, H.-P. Kriegel, R.T. Ng, J. Sander, LOF: identifying density-based local outliers, in: ACM SIGMOD Record, volume 29, 2000, pp. 93–104.
[40] J. Hardin, D.M. Rocke, Outlier detection in the multiple cluster setting using the minimum covariance determinant estimator, Comput. Stat. Data Anal. 44 (4) (2004) 625–638.
[41] B. Scholkopf, J.C. Platt, J. Shawe-Taylor, A.J. Smola, R.C. Williamson, Estimating the support of a high-dimensional distribution, Neural Comput. 13 (7) (2001) 1443–1471.
[42] M.-L. Shyu, S.-C. Chen, K. Sarinnapakorn, L. Chang, A novel anomaly detection scheme based on principal component classifier, in: Int. Conf. Data Mining, 2003.
[43] Y. Weng, N. Zhang, C. Xia, Multi-agent-based unsupervised detection of energy consumption anomalies on smart campus, IEEE Access 7 (2018) 2169–2178.
[44] 6 functions for generating artificial datasets - File Exchange - MATLAB Central, Accessed 2019-10-09, https://www.mathworks.com/matlabcentral/fileexchange/41459.
[45] Y. LeCun, C. Cortes, C.J.C. Burges, MNIST Handwritten Digit Database, 2010,
[46] J. Wright, A.Y. Yang, A. Ganesh, S.S. Sastry, Y. Ma, Robust face recognition via sparse representation, IEEE Trans. Pattern Anal. Mach. Intell. 31 (2) (2008) 210–227.
[47] I. Ramirez, P. Sprechmann, G. Sapiro, Classification and clustering via dictionary learning with structured incoherence and shared features, in: IEEE Conf. Comp. Vis. Patt. Recog., 2010, pp. 3501–3508.
[48] M. Yang, L. Zhang, X. Feng, D. Zhang, Fisher discrimination dictionary learning for sparse representation, in: Int. Conf. Comp. Vis., 2011, pp. 543–550.
[49] S. Kong, D. Wang, A dictionary learning approach for classification: separating the particularity and the commonality, in: European Conf. Comp. Vis., 2012, pp. 186–199.
[50] T.H. Vu, V. Monga, Learning a low-rank shared dictionary for object classification, in: IEEE Int. Conf. Image Process., 2016, pp. 4428–4432.
[51] J.J. Hull, A database for handwritten text recognition research, IEEE Trans. Pattern Anal. Mach. Intell. 16 (5) (1994) 550–554.
[52] J. Mairal, F. Bach, J. Ponce, Task-driven dictionary learning, IEEE Trans. Pattern Anal. Mach. Intell. 34 (4) (2011) 791–804.
[53] J. Wang, J. Yang, K. Yu, F. Lv, T. Huang, Y. Gong, Locality-constrained linear coding for image classification, in: IEEE Conf. Comp. Vis. Patt. Recog., 2010, pp. 3360–3367.
[54] L. Condat, Fast projection onto the simplex and the l1 ball, Math. Program. 158 (1–2) (2016) 575–585.
[55] D.I. Inouye, E. Yang, G.I. Allen, P. Ravikumar, A review of multivariate distributions for count data derived from the poisson distribution, Wiley Interdiscip. Rev. 9 (3) (2017) e1398.
[56] Y.K. Belyaev, Y.P. Lumen'skii, Multidimensional poisson walks, J. Soviet Math. 40 (2) (1988) 162–165.