The pyramid quantized Weisfeiler-Lehman graph representation

Katerina Gkirtzou[a,b,c], Matthew B. Blaschko[b,c,*]

[a]*Research Center Athena, Artemidos 6 & Epidavrou, Marousi 15125, Greece*
[b]*Inria Saclay, Campus de l'École Polytechnique, 91120 Palaiseau, France*
[c]*CentraleSupélec, Grande Voie des Vignes, 92295 Châtenay-Malabry, France*

Abstract

Graphs are flexible and powerful representations for non-vectorial structured data. Graph kernels have been shown to enable efficient and accurate statistical learning on this important domain, but many graph kernel algorithms have high order polynomial time complexity. Efficient graph kernels rely on a discrete node labeling as a central assumption. However, many real world domains are naturally described by continuous or vector valued node labels. In this article, we propose an efficient graph representation and comparison scheme for large graphs with continuous vector labels, the *pyramid quantized Weisfeiler-Lehman graph representation*. Our algorithm considers statistics of subtree patterns with discrete labels based on the Weisfeiler-Lehman algorithm and uses a pyramid quantization strategy to determine a logarithmic number of discrete labelings that results in a representation that guarantees a multiplicative error bound on an approximation to the optimal partial matching. As a result, we approximate a graph representation with continuous vector labels as a sequence of graphs with increasingly granular discrete labels. We evaluate our proposed algorithm on two different tasks with real datasets, on a fMRI analysis task and on the generic problem of 3D shape classification. Source code of the implementation can be downloaded from `https://web.imis.athena-innovation.gr/~kgkirtzou/Projects/WLpyramid.html`.

Keywords:
Weisfeiler-Lehman algorithm, graph kernels, regularization, vector labels

1. Introduction

Graphs are a powerful and natural way to represent complex data with integrated structure. Graphs have been used in numerous applications in a number of different fields, such as (i) computer vision [1, 2] and biomedical imaging analysis [3, 4, 5], (ii) bioinformatics [6, 7], (iii) social networks analysis [8] and (iv) chemoinformatics [9]. In many applications, the exploration of the data requires the ability to efficiently compare graphs and to provide a similarity measurement, a problem known as *graph comparison*.

Graph kernels are the most recent approaches that tackle both the problem of graph representation and graph comparison and have been found to be useful across a wide range of applications. They exploit the graph topology by decomposing the graph into substructures and aggregating statistics over these substructures. This strategy considers a measure of similarity between the graphs as a form of inner product. Graph kernels are commonly derived as an instance of the family of the R-convolution kernels [10], which are a generic way of constructing kernels of complex objects by decomposing them into discrete structures and comparing all pairs of decompositions. Every new decomposition would yield a new kernel. A first approach would be to decompose the graphs into all possible subgraphs. However, calculating all subgraphs is at least as hard as deciding whether two graphs are isomorphic [11]. So it is necessary to limit the decomposition of the graphs only into specific types of subgraphs that are computable in polynomial time [12, 13]. There are three main categories of graph kernels, graph kernels based on (i) walks and paths, (ii) subgraphs of limited size, and (iii) subtree patterns. Table 1 shows a summary of the state of the art of graph kernels grouped per category, while we review extensively each of these categories in the following Section 1.2.

*Corresponding author. Phone: +33 141131098. Fax: +33 141131006.
Email address: `matthew.blaschko@inria.fr` (Matthew B. Blaschko)

Although efficient graph kernels have been developed that have good performance on discretely labeled graphs, the literature on continuously or vector labeled graphs is still relatively undeveloped, as is shown in Table 1. As a result, existing graph kernels are poorly suited to applications that are naturally represented by continuous node labeled graphs, such as in graph representations for fMRI analysis and shape classification based on 3D mesh representations. We help to address this shortcoming of existing methods by proposing a novel framework for kernel construction, *the pyramid quantized Weisfeiler-Lehman graph representation*. Our method makes use of the efficiency of the Weisfeiler-Lehman kernel [13] for discrete labels by converting continuous vector labeled graphs into a sequence of discretely labeled graphs using a pyramid quantization strategy. This results in a kernel with a multiplicative error bound on an optimal matching measure.

Firstly, we exploit the key concepts of the Weisfeiler-Lehman test of isomorphism and then see how it is used to produce subtree patterns for comparing discrete labeled graphs in Section 2. In Section 3 we present our pyramid quantization scheme and we then learn different strategies for combing the pyramid levels in Section 4. Secondly, we perform an experimental comparison with a graph matching technique in Section 5 and we evaluate our proposed method in two general tasks, an fMRI analysis task in Section 6 and 3D mesh shape classification in Section 7. Finally, we conclude with a discussion in Section 8.

1.1. Terminology

Before reviewing the graph kernels literature, we clarify our terminology to provide a unified presentation of previous work. We define a graph as a triplet $G = (V, E, \mathcal{L})$, where V is the vertex set, E is the edge set and $\mathcal{L} : V \mapsto \Sigma$ is a function assigning a label from an alphabet Σ to each vertex in the graph. The neighborhood $N(v) = \{v' | (v, v') \in E\}$ of a vertex v is the set of all vertices adjacent to v, *i.e.* all vertices connected with a single edge. The degree $d(v)$ of a vertex v is the number of edges incident with v. Every graph has at most v vertices, e edges and a maximum degree of d. A walk in a graph is a sequence of adjacent vertices. A path is a walk that contains only distinct vertices, while a cycle is a closed walk. A rooted tree is an acyclic graph with a specified root vertex. The height of a tree is the maximum distance between the root vertex and any other vertex in the tree. Subtree patterns are labeled trees extracted from a labeled graph G for a given depth h and a given vertex v. Repetition of the same vertex is allowed in subtree pattern, but it is treated as distinct vertices, allowing a cycle-free pattern.

1.2. Related Work

The first class of kernels are based on walks (paths) and compute the number of matching pairs of random walks (paths). The standard formulation of random walk kernel counts the number of nodes in the walk which have the same label and requires $O(\mathsf{v}^6)$ computation for a pair of graphs [11]. Restating the same problem in terms of Kronecker products reduces the runtime complexity to $O(\mathsf{v}^3)$ for a pair of graphs [12]. Another approach based on dynamic programming speeds up the computations of the random walk kernel [14] at the cost of considering only walks of fixed size. The marginalized graph kernel [15] modifies the label of each vertex with the use of the Morgan index [16], which increases the specificity of labels by adding information on the number of walks starting from that node, and therefore reduces the run-time as fewer vertices will match. A graph kernel that compares the length of shortest path between pairs of nodes with matching source and sink labels in two graphs requires $O(\mathsf{v}^4)$ [17]. A specialized graph kernel for chemoinformatics [18] uses molecular fingerprinting techniques and counts labeled paths of length p that can be retrieved by depth-first search from each vertex, an efficient approach for graphs with an average node degree of 2 or 3.

The second class of graph kernels are based on limited-size subgraph structures called *graphlets*. A naive computation of all graphlets of a graph, without considering labels, requires $O(\mathsf{v}^k)$ computations, where $k \in \{3, 4, 5\}$ is the size of subgraphs. Since enumerating all graphlets is prohibitively expensive, even for small values of k, [19] showed that sampling a fixed number of graphlets suffices to bound the deviation of the empirical estimates of the graphlet distribution from the true distribution and for graphs of degree bounded by d, the exact number of all graphlets of size k can be determined in time $O(\mathsf{v}\mathsf{d}^{k-1})$. Another kernel is the cyclic pattern kernels [20], which count pairs of matching cyclic and tree patterns in two graphs. In the general case, the cyclic pattern kernel is NP-hard, but in specific cases the kernel can be computed efficiently. Finally, the neighborhood subgraph pairwise distance kernel [21] decomposes a graph into all pairs of neighborhood subgraphs of small radius r at increasing distances.

The third class of graph kernels is based on subtree patterns. The Ramon-Gärtner subtree kernel [22] compares all pairs of nodes from two labeled graphs by iteratively comparing their subtree patterns of height h. Although the subtree kernel is more expressive than kernels based on walks, unfortunately it is computationally expensive, since for a set of n graphs it requires $O(n^2\mathsf{v}^2 h 4^{\mathsf{d}})$. Mahé and Vert [23] and Bach [24] refine the Ramon-Gärtner subtree kernel for applications in chemoinformatics with a parameter to control the complexity of the subtree features and hand-written digit recognition considering α-ary subtrees with at most α children per node. Unfortunately, the complexity of both kernels is still exponential in the smoothing and α parameter, and both kernels are feasible on small sized graphs only. More recently an efficient kernel, the Weisfeiler-Lehman subtree kernel, was introduced in [13]. It uses the Weisfeiler-Lehman test of isomorphism [25] to efficiently compute subtree patterns up to height h for discretely labeled graphs. For n pairs of discretely labeled graphs, the Weisfeiler-Lehman subtree kernel requires $O(nh\mathsf{e} + n^2 h\mathsf{v})$. At its core, the Weisfeiler-Lehman kernel achieves linear complexity in the graph size due to an efficient hashing scheme for discrete labeled graphs, a remarkable improvement over previous approaches.

The majority of these methods focus on either unlabeled or discretely labeled graphs, while an efficient and expressive representation and comparison of graphs with continuous high-dimensional vectors remains an open research problem. Initial results in this direction have been developed by [26], which address the special case of scalar valued node labels with missing data, and by [27] based on sub-path similarities. We are unaware of previous work that has achieved computational complexity linear in the size of the graph, as in our method.

2. The Weisfeiler-Lehman kernel

2.1. The Weisfeiler-Lehman test of isomorphism

Our proposed algorithm uses the discretely labeled subtree pattern features introduced by the Weisfeiler-Lehman kernel [13], which exploits the key concepts from the Weisfeiler-Lehman test of isomorphism [25]. The key idea, given two graphs, is the construction of augmented node labels from the labels of all the neighbor nodes and their compression into new short labels. This process is repeated until either the label sets of the two graphs differ or the maximum number of iterations has been reached. In the former case, the two graphs are not isomorphic, while in the latter the test was not able to determine whether they are not isomorphic. Algorithm 1 provides pseudocode for the Weisfeiler-Lehman algorithm, while Figure 1 shows its first iteration. The algorithm has fast run-time $O(h\mathsf{e})$, where h is the maximum number of iterations of the test and e the maximum number of edges [13].

2.2. The linear Weisfeiler-Lehman subtree kernel

For each iteration i of the Weisfeiler-Lehman algorithm, we obtain new compressed labels $l_i(v)$ for all nodes v in all graphs (see Line 12 in Algorithm 1). Each compressed label $l_i(v)$ is a subtree pattern of height i rooted at node v and histograms of their occurrences have been recently employed in a kernel for comparing graphs with discrete labels, the linear Weisfeiler-Lehman subtree kernel [13, Definition 4]. Table 2 shows the subtree patterns up to depth $h = 1$ produced by the Weisfeiler-Lehman algorithm and their respective occurrences between the two graphs G, G' shown in Figure 1(a).

The linear Weisfeiler-Lehman subtree kernel is computed as an inner product of a vector of statistics $\phi_{(h)}(G)$ explicitly computed for each graph (Table 2). A key advantage of the kernel is its complexity being linear in the graph size $O(nh\mathsf{e} + n^2 h\mathsf{v})$ for n graphs, where e is the maximal number of edges, v the maximal number of vertices and h the depth of subtree patterns used [13, Theorem 4]. In addition to these computational benefits, linear Weisfeiler-Lehman graph kernels have been shown to perform comparably to or better than a number of more computationally complex kernels [13]. Finally, we note that the linear Weisfeiler-Lehman algorithm at depth 0 computes exactly the bag of words representation commonly used in natural language processing [28, 29] and computer vision [30, 31].

3. The pyramid quantization strategy for continuous vector labels

The Weisfeiler-Lehman algorithm is efficient precisely because it makes use of a discrete labeling over nodes, which enables an efficient hashing scheme. A problem occurs when extending this method to continuous labeled graphs: we no longer have a notion of an exact match of a discrete label, and a hash function that counts approximate matches would implicitly define a single quantization of the vector space to a discrete set of labels. A single quantization is inexact, and gives only a weak relationship to the potentially rich geometry of the original label space. It is also not clear what the resolution of the quantization should be to maximize performance. To overcome this, we propose a pyramid quantization strategy similar to the one used by [32, 33] to determine a logarithmic number of quantizations, ending with a sequence of graphs with discrete labels of increasing granularity, for which we run the Weisfeiler-Lehman algorithm.

Table 1: An overview of graph kernel methods. From left to right we show the type of subgraphs used, the algorithm, its complexity when that is known and whether the kernel works on unlabeled, discretely, continuous or vector labeled graphs. Note that n is the number of graphs under comparison, v is the maximal number of nodes, e is the maximal number of edges, h is the height of subtree patterns, d is the maximum degree, k is the size of graphlets, δ is the graph diameter and f is the dimension of the node labels.

	Algorithm	**Complexity**	**Unlabeled**	**Discrete**	**Continuous**	**Vector**
Walks	Gärtner et al. [11]	$O(n^2v^6)$	✓	✓	✓	✓
	Mahe et al. [15]		✓	✓		
	Vishwanathan et al [12]	$O(n^2v^3)$	✓	✓	✓	✓
Paths	Borgwardt et al. [17]	$O(n^2v^4)$	✓	✓	✓	✓
	Ralaivola et al. [18]		✓	✓		
	Feragen et al. [27]	$O(n^2v^2(m + \log v + \delta^2 + f))$	✓	✓	✓	✓
Graphlets	Horvath et al. [20]		✓	✓		
	Shervashidze et al. [19]	$O(vd^{k-1})$	✓			
	Costa et al. [21]		✓	✓		
Subtree Patterns	Ramon et al. [22]	$O(n^2v^2h4^d)$	✓	✓		
	Bach et al. [24]		✓	✓		
	Mahe et al [23]		✓	✓		
	Shervashidze et al. [13]	$O(nhe + n^2hv)$	✓	✓		
	Neumann et al. [26]		✓	✓	✓	✓

Table 2: An example of the subtree patterns up to depth $h = 1$ for the graphs from Figure 1(a). The first row are the labels $\Sigma_0 \cup \Sigma_1$ encountered up to depth $h = 1$, while the second and the third row the occurrences of subtrees for graph G and G' respectively

		Labels Σ_0			Labels Σ_1							
Labels $\Sigma_0 \cup \Sigma_1$	=	{1,	2,	3,	4,	5,	6,	7,	8,	9,	10,	11}
$\phi_{(1)}(G)$	=	(3,	1,	3,	0,	3,	0,	1,	2,	1,	0,	0)
$\phi_{(1)}(G')$	=	(2,	2,	3,	1,	1,	1,	1,	1,	0,	1,	1)

3.1. *The pyramid quantization strategy*

Given a vector labeled graph $G = (V, E, \mathcal{L})$, where $\mathcal{L} : V \mapsto \mathbb{R}^d$ is the function assigning a d-dimensional vector label to each vertex, we want to derive a hierarchical decomposition of $\mathbb{R}^d$ as multi-resolution quantizations that will then be used to determine the discrete labeling of increasing granularity. First we construct a set of quantization functions $Q^{(l)} : \mathbb{R}^d \mapsto \Sigma_0^{(l)}$ that will encode the continuous vector labels into a quantization of a given resolution $|\Sigma_0^{(l)}| = 2^l$. The quantization function $Q^{(l)}$ is generated for multiple resolutions $l \in \{0, \ldots, L\}$, where $L = \lceil \log_2 D \rceil$, D is the number of unique values in the image of the vertex set V under the label function $\mathcal{L}$. Note that the single quantization bin for $Q^{(0)}$ is big enough so that all vertices receive the same discrete label, while the coarser quantization $Q^{(L)}$ contains quantization bins that are small enough so each unique continuous vector label falls into its own quantization bin. To achieve this hierarchical quantization in the experiments performed here, we have used an agglomerative hierarchical clustering with Ward's minimum variance method [34].

Secondly, we compose the quantization function $Q^{(l)}$ with the labeling function $\mathcal{L}$, $\forall l \in \{0, \ldots, L\}$, and we approximate our initial vector labeled graph G as a sequence of graphs with discrete labels of increasing granularity:

$$G = (V, E, \mathcal{L}) \overset{Q^{(l)} \circ \mathcal{L}}{\mapsto} \left(G^{(0)}, \ldots, G^{(L)}\right) \quad (1)$$
$$= \left((V, E, \mathcal{L}^{(0)}), \ldots, (V, E, \mathcal{L}^{(L)})\right),$$

where $\mathcal{L}^{(l)} : V \mapsto \Sigma_0^{(l)}$ is defined to be $Q^{(l)} \circ \mathcal{L}$, and $\Sigma_0^{(l)}$ is the discrete label alphabet for a given level l of quantization. Note that the topology of the graph does not change in the sequence of graphs, only the continuous vector labels are discretized.

This type of quantization scheme, when paired with a histogram intersection kernel and an appropriately weighted linear combination of kernel values across

Algorithm 1 The one dimensional Weisfeiler-Lehman test of graph isomorphism

```
Require: Two graphs, $G = (V, E, \mathcal{L})$, $G' = (V', E', \mathcal{L}')$, with discrete labelings $\mathcal{L} : V \mapsto \Sigma$ and $\mathcal{L}' : V' \mapsto \Sigma$ over
    vertices, where $\Sigma$ is a vertex label set and the maximum number of iterations $h$.
 1: $i \leftarrow 0$
 2: repeat
 3:   if $i = 0$ then
        {———— Multiset-label initialization ————}
 4:     $M_i(v) := l_0(v) = \mathcal{L}(v)$.
 5:   else if $i \geq 1$ then
        {———— Multiset-label determination ————}
 6:     Assign a multiset-label $M_i(v)$ to each node $v$ in $G$ and $G'$ which consists of the multiset $\{l_{i-1}(u) | u \in \mathcal{N}(v)\}$,
        where $\mathcal{N}(v)$ denotes the neighbor set of $v$.
        {———— Sorting each multiset ————}
 7:     Sort the elements in $M_i(v)$ in ascending order.
 8:     Concatenate the elements in $M_i(v)$ into a string $s_i(v)$.
 9:     Add $l_{i-1}(v)$ as a prefix to $s_i(v)$.
        {———— Sorting the set of multisets ————}
10:     Sort all of the strings $s_i(v)$ for all $v$ from $G$ and $G'$ in ascending order.
        {———— Label compression via hashing ————}
11:     Map each string $s_i(v)$ to a new compressed label using a function $f : \Sigma^* \mapsto \Sigma$ such that $f(s_i(v)) =$
        $f(s_i(w)) \iff s_i(v) = s_i(w)$.
        {———— Relabeling ————}
12:     Set $l_i(v) := f(s_i(v))$ for all nodes in $G$ and $G'$.
13:   end if
14:   $i \leftarrow i + 1$
15: until $\{l_i(v) | v \in V\} \neq \{l_i(v') | v' \in V'\}$ or $i > h$
```

quantization levels, results in a multiplicative error bound on the optimal graph matching [32, Proposition 3]. We may therefore interpret the pyramid quantized Weisfeiler-Lehman graph representation as a function space that enables tight approximations to cost of the optimal matching over vector representations of subtree patterns.[1]

3.2. The intersection Weisfeiler-Lehman subtree kernel

Each graph with vector valued labels after the pyramid quantization step described in Section 3.1 is represented as a sequence of graphs with nested quantizations of increasing granularity as described in Equation 1. We construct a kernel, using the following definition:

Definition 3.1 (The intersection Weisfeiler-Lehman subtree kernel)**.** Let $G^{(l)} = (V, E, \mathcal{L}^{(l)})$ and $G'^{(l)} = (V', E', \mathcal{L}'^{(l)})$ be two graphs of the same quantization level l, where $\mathcal{L}^{(l)} : V \mapsto \Sigma_0^{(l)}$ and $\mathcal{L}'^{(l)} : V' \mapsto \Sigma_0^{(l)}$, of two vector labeled graphs $G = (V, E, \mathcal{L})$ and $G' = (V', E', \mathcal{L}')$, where $\mathcal{L} : V \mapsto \mathbb{R}^d$ and $\mathcal{L}' : V' \mapsto \mathbb{R}^d$. Define $\Sigma_i^{(l)} \subseteq \Sigma^{(l)}$ as the set of symbols that occur as node labels at least once in $G^{(l)}$ or $G'^{(l)}$ at the end of the i-th iteration of the Weisfeiler-Lehman algorithm. Let $\Sigma_0^{(l)}$ be the set of original node labels of $G^{(l)}$ and $G'^{(l)}$. Assume all $\Sigma_i^{(l)}$ are pairwise disjoint. Without loss of generality, assume that every $\Sigma_i^{(l)} = \{\sigma_{i1}^{(l)}, \ldots, \sigma_{i|\Sigma_i^{(l)}|}^{(l)}\}$ is ordered. Define a map $\eta_i : \{G^{(l)}, G'^{(l)}\} \times \Sigma_i^{(l)} \mapsto \mathbb{N}$ such that $\eta_i(G^{(l)}, \sigma_{ij}^{(l)})$ is the number of occurrences of the letter $\sigma_{ij}^{(l)}$ in the graph $G^{(l)}$. The intersection Weisfeiler-Lehman subtree kernel on two graphs G and G' with h iterations is defined as

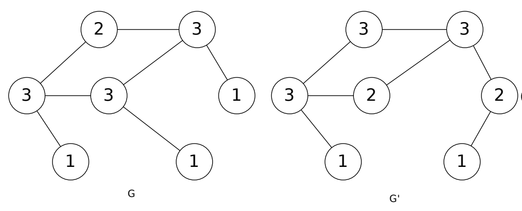
$$K_{i-WLsubtree}^{(h)}(G^{(l)}, G'^{(l)}) := \mathcal{I}\left(\phi_{(h)}^{(l)}(G^{(l)}), \phi_{(h)}^{(l)}(G'^{(l)})\right) \quad (2)$$

where

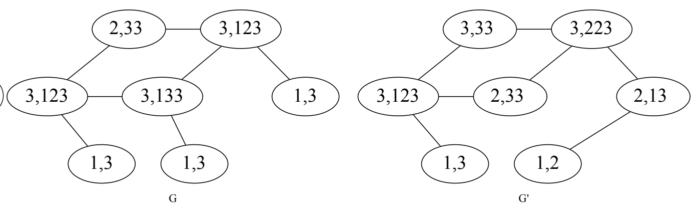
$$\phi_{(h)}^{(l)}(G^{(l)}) = \left(\eta_0(G^{(l)}, \sigma_{01}^{(l)}), \ldots, \eta_h(G^{(l)}, \sigma_{h|\Sigma_h^{(l)}|}^{(l)})\right) \quad (3)$$

$$\phi_{(h)}^{(l)}(G'^{(l)}) = \left(\eta_0(G'^{(l)}, \sigma_{01}^{(l)}), \ldots, \eta_h(G'^{(l)}, \sigma_{h|\Sigma_h^{(l)}|}^{(l)})\right) \quad (4)$$

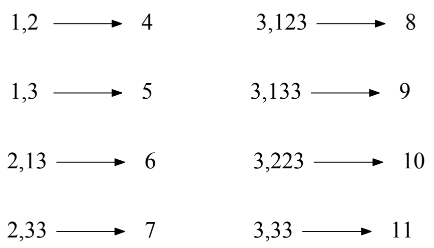
[1] In practice, we do not use the fixed weighting across quantization levels proposed by Grauman and Darrell [32], but instead discriminatively optimize over the function space (see Section 4).



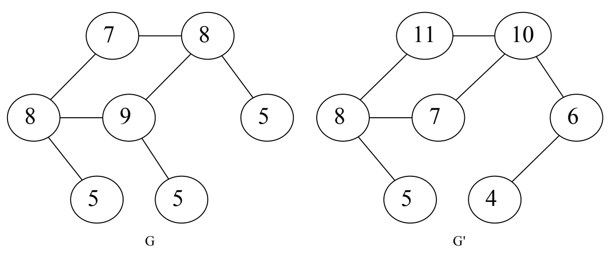
(a) Given labeled graphs G and G'.



(b) Multilabel determination and sorting.



(c) Label compression.



(d) Relabeling graphs.

Figure 1: An illustration of the Weisfeiler-Lehman algorithm for the iteration $i = 1$. Note that there is a one-to-one correspondence between the compressed labels and subtree patterns. For example, the compressed label 9 corresponds to a subtree pattern of height 1 where the root node has label 3 and its children nodes have labels 1, 2 and 3 respectively.

and

$$\mathcal{I}\left(\phi^{(l)}_{(h)}(G^{(l)}), \phi^{(l)}_{(h)}(G'^{(l)})\right) = \sum_{i=0}^{h}\sum_{j=1}^{|\Sigma_i^{(l)}|} \min\left(\eta_i(G^{(l)}, \sigma^{(l)}_{ij}), \eta_i(G'^{(l)}, \sigma^{(l)}_{ij})\right) \quad (5)$$

Note that the intersection Weisfeiler-Lehman subtree kernel counts the overlap of subtree features $\phi^{(l)}_{(h)}$ between two graphs at the same quantization level l and for a given binning resolution l the intersection kernel is a positive-definite similarity function [35]. We define and propose the use of the intersection kernel instead of the dot-product kernel, as it can give us better control over feature matches per level due to the overlapping label representation. Moreover, the histogram intersection kernel has been shown to give good results in a number of applications [35].

For the pyramid quantized Weisfeiler-Lehman kernel described above the following monotonicity property holds:

Theorem 3.2 (Monotonicity property of the pyramid quantized Weisfeiler-Lehman kernel with the granularity of the node labeling)**.** *The Weisfeiler-Lehman algorithm for a given height h of subtree patterns produces histograms whose intersection are monotonically decreasing in the granularity of the graph node labeling:*

$$\mathcal{I}\left(\phi^{l}_{(h)}(G^{(l)}), \phi^{l}_{(h)}(G'^{(l)})\right) \geq \mathcal{I}\left(\phi^{l+1}_{(h)}(G^{(l+1)}), \phi^{l+1}_{(h)}(G'^{(l+1)})\right) \quad \forall l, G^{(l)}, G'^{(l)}, \quad (6)$$

where $\phi^{l}_{(h)}(G^{(l)})$ is the histogram of subtree patterns of height h computed at pyramid level l, and level $l+1$ is more granular.

Proof. We first note that the number of subtree patterns of a given depth, h, of the Weisfeiler-Lehman algorithm is dependent only on the topology of the graph, and not on the graph labeling:

$$||\phi^{l}_{(h)}(G^{(l)})||_1 = \mathrm{v} \cdot (h+1)\ \forall l \quad (7)$$

where v is the number of vertices in the graphs. We next note that the number of vertex labels is strictly monotonic in the pyramid level, $|\Sigma^l| < |\Sigma^{l+1}|$, and that for each label $\sigma \in \Sigma^l$ at level l, there exist a non-empty set of labels $\mathcal{S} \subset \Sigma^{l+1}$ at level $l+1$ such that $\mathcal{L}^{l+1}(u) \in \mathcal{S} \iff \mathcal{L}^l(u) = \sigma$. To complete the proof, we observe that

$$\begin{aligned}\forall G^{(l)}\ \forall l\Big[&\left(\|\phi^{l}_{(h)}(G^{(l)})\|_1 = \|\phi^{l+1}_{(h)}(G^{(l+1)})\|_1\right) \wedge \\ &\left(\|\phi^{l}_{(h)}(G^{(l)})\|_0 < \|\phi^{l+1}_{(h)}(G^{(l+1)})\|_0\right) \Big] \Longrightarrow \\ \forall G^{(l)}, G'^{(l)}\ \forall l\Big[&\mathcal{I}\left(\phi^{l}_{(h)}(G^{(l)}), \phi^{l}_{(h)}(G'^{(l)})\right) \geq \\ &\mathcal{I}\left(\phi^{l+1}_{(h)}(G^{(l+1)}), \phi^{l+1}_{(h)}(G'^{(l+1)})\right)\Big], \end{aligned} \tag{8}$$

where $\| \cdot \|_0$ is the ℓ_0 pseudo norm. □

Proposition 3.3. *The intersection Weisfeiler-Lehman subtree kernel (Definition 3.1) specializes to a bag-of-words intersection kernel for $h = 0$.*

Proof. For $h = 0$, η^l maps to Σ^l_0, a single quantization. The outer summation in Equation (5) is over one term, and the inner summation computes a histogram intersection over a single fixed quantization. □

As the Weisfeiler-Lehman algorithm with $h = 0$ specializes to the bag of words model as a result of Proposition 3.3, we have as a result that our graph kernel for continuous vector valued node labels strictly generalizes the pyramid match kernel [32]. Theorem 3.2 generalizes the monotonicity exploited in the construction of the pyramid match kernel to guarantee that Mercer's condition holds [32, Equation (4) & Proposition 1].

3.3. The pyramid quantized Weisfeiler-Lehman kernel complexity

The main steps of the pyramid quantized Weisfeiler-Lehman kernel are the calculation of the multi-resolution hierarchical quantization of the continuous vector labels of the vertices, the assignment of the initial continuous vector label of each vertex to the appropriate path of the hierarchical quantization and the calculations of the subtree pattern features. Calculating a multi-resolution hierarchical quantization of m unique values requires $O(m^3)$ calculations. In the ideal case, we would like to use all unique values in the image of the vertex set V under the label function $\mathcal{L} : V \to R^d$, but since in some applications this number can be prohibitively large, a representative subset of size $m \ll D$ – where D is the number of unique values in the image of the vertex set V under the label function $\mathcal{L}$ – can be sufficient. Independent of whether all unique label values or a subset is used to calculate the multi-resolution hierarchical tree, this process should be distinguished as a separate step that can be done once per application domain, and it is not required during test time.

During test time, the steps of the *pyramid quantized Weisfeiler-Lehman graph representation* are the assignment of the continuous vector label of each vertex to the appropriate path of the multi-resolution hierarchical quantization and the calculation of the subtree pattern features. Note that since we want to approximate our initial vector labeled graph $G(V, E, \mathcal{L})$ as a sequence of $L = \lceil \log_2 D \rceil$ graphs with discrete labels of increasing granularity, that means we need to calculate L Weisfeiler-Lehman kernels. For the assignment of a given vector label to the hierarchical quantization, it should be emphasized that the multi-resolution hierarchical quantization can be seen as a complete binary tree of height L. So the label's placement in the hierarchical quantization tree is determined by comparing it to the appropriate two bin centers at each of the L levels. So we start at the second coarser quantization level $Q^{(1)}$ where all labels receive the only two discrete labels, (the coarser level $Q^{(0)}$ is a trivial one where all labels get the same discrete label) and we compare it with the two bin centers. The label gets the assigned discrete label of the bin center that is nearest to it in terms of the distance used to create the hierarchical quantization tree. Then, we push the label down to hierarchical quantization tree and continue to finer levels only along the branch that is chosen at each level. So a label is first assigned to one of the coarser levels and then it is assigned to one of its children, and so on recursively. This amounts to a total of $2L$ comparisons in order to find all multiresolution discrete labels for a given vector label and $O(\mathsf{v}L)$ for a given graph. The complexity of label assignment is dominated by the complexity of the kernel computation for L quantization levels, and consequently the overall complexity for n graphs is $O(L(nh\mathsf{e} + n^2h\mathsf{v}))$.

4. Learning to combine pyramid quantized Weisfeiler-Lehman features

Given a set of vector labeled graphs $\mathbf{G} = \{G_i = (V_i, E_i, \mathcal{L}_i)\}_{1 \le i \le n}$ where $\mathcal{L} : V \mapsto \mathbb{R}^d$ and a classification label $y_i \in \mathcal{Y}$ for each graph G_i, we classify graphs using the representation created by the Weisfeiler-Lehman algorithm after their quantization as described in Section 3. In order to maximize their classification performance we examine two different approaches, one through multiple kernel learning (Section 4.1) and another through direct sparse regularization of individual subtree patterns (Section 4.2).

4.1. Multiple kernel learning

Applying the intersection Weisfeiler-Lehman subtree kernel for each pair of graphs $G^{(l)}, G'^{(l)}$ for all the pyramid levels from Equation 1, we end up with a sequence of intersection Weisfeiler-Lehman kernels for a given height h of subtree patterns : $\left(K_{(h)}^{(0)}(G^{(0)}, G'^{(0)}), \ldots, K_{(h)}^{(L)}(G^{(l)}, G'^{(L)})\right)$. Taking also into consideration the observation by [36] that using multiple kernels instead of a single one can improve performance, we would like to combine our sequence of kernels. A common approach considers that the kernel $K(G, G')$ is actually a convex combination of *basis* kernels:

$$K_{(h)}(G, G') = \sum_{l=0}^{L} d_l K_{(h)}^{(l)}(G^{(l)}, G'^{(l)}), \;\; d_l \geq 0, \;\; \sum_{l=0}^{L} d_l = 1. \tag{9}$$

For determining the weights d_l we consider two different approaches: an automatic way through the framework of *multiple kernel learning* and another through *a fixed weight scheme*.

The automatic determination of the weights d_l of the linear combination of our multiple kernels $K_{(h)}^{(l)}(G^{(l)}, G'^{(l)})$ as well as the coefficients α_i,β in a single optimization problem is known as the multiple kernel learning (MKL) problem [37, 38, 36, 39]. The multiple kernel learning approach addresses the problem through a weighted ℓ_2 norm regularization formula. In addition, a ℓ_1 norm is posed as a constraint on the kernel weights a_i. This additional constraint encourages sparse set of basis kernels as an inherited property from the ℓ_1 norm. The primal MKL problem is defined as

$$\begin{aligned} &\min_{f,\beta,\xi,d} \frac{1}{2} \sum_{l=0}^{L} \frac{1}{d_l} \|f_l\|_{\mathcal{H}_l}^2 + C \sum_{i=1}^{n} \xi_i \\ &\text{s.t. } y_i \sum_{l=0}^{L} f_l(G_i^{(l)}) + y_i\beta \geq 1 - \xi_i \forall i \qquad (10) \\ &\xi_i \geq 0 \;\forall i, \sum_{l=0}^{L} d_l = 1, \; d_l \geq 0 \;\forall l \end{aligned}$$

where each function f_l belongs to a different RKHS $\mathcal{H}_l$ associated with a kernel $K_{(k)}^{(l)}$. The constraint of the ℓ_1 norm on the weighted leads to sparse solutions, which may be undesirable if all pyramid levels are informative. We therefore additionally consider taking their empirical mean:

$$K_{(h)}(G, G') = \sum_{l=0}^{L} d_l K_{(h)}^{(l)}(G^{(l)}, G'^{(l)}), \text{ where } d_l = \frac{1}{L+1}. \tag{11}$$

Visualization. When we consider a linear combination of our kernels and since the intersection kernel can be considered as a "quasi-linear" kernel [40], we can develop visualizations that approximate the learned discriminant functions. The discriminant function for class c has the form:

$$\begin{aligned} f_c(G) &= b + \sum_{l=0}^{L} d_l \sum_{j} \alpha_{lj} \mathcal{I}\left(\phi_{(h)}^{(l)}(G_j^{(l)}), \phi_{(h)}^{(l)}(G^{(l)})\right) && (12) \\ &\approx b + \sum_{l=0}^{L} d_l \langle w_l, \phi_{(h)}^{(l)}(G^{(l)})\rangle && (13) \end{aligned}$$

where l indexes the levels of the pyramid and j indexes over the samples in the training set. At each pyramid level l there is exactly one subtree pattern of a given height h rooted at each vertex of the graph. We may generate for each vertex v a visualization by coloring each vertex by the weight in w_i corresponding to the subgraph pattern rooted at that node.

4.2. Elastic Net on the pyramid quantized Weisfeiler-Lehman subtree features

We also consider a direct sparse regularization on the subtree patterns using the Elastic Net [41]. The Elastic Net linearly combines ℓ_1 with ℓ_2 regularization in order to appropriately trade off sparsity with a low variance estimator in the case of correlated signals. Formally, if $\phi_{(h)}^{(l)}(G_i^{(l)})$ is a feature vector of subtree pattern up to height h for a given quantization level l for a graph G_i, the Elastic Net computes

$$\hat{\beta} = \arg\min_{\beta \in \mathbb{R}^d} \lambda_2 \|\beta\|_2^2 + \lambda_1 \|\beta\|_1 + \frac{1}{n} \sum_{i=1}^{n} \left(\langle \beta, \phi_{(h)}^{(l)}(G_i^{(l)})\rangle - y_i\right)^2, \tag{14}$$

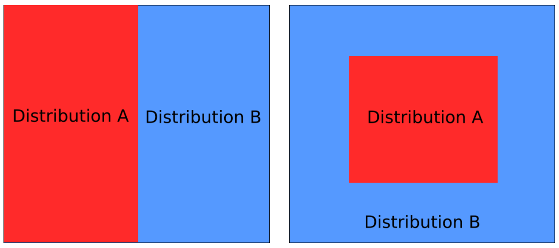
where λ_1, $\lambda_2 \geq 0$ are scalar regularization parameters.

5. Experimental Comparison with Graph Matching

Graph matching has been proposed to construct graph kernels. The matching energy is used as a kernel value [42, 43, 44]. Although this does not lead to a strictly positive definite function, it can nevertheless yield reasonable performance in some domains. In this section, we construct synthetic data sets that show the relative advantages of the pyramid quantized Weisfeiler-Lehman kernel vs. graph matching based approaches.

We perform an experimental comparison using the factorized graph matching technique [45]. We perform a classification experiment with synthetic node labeled graphs from two classes.

In order to create the graphs, we consider a 2D unit square plane, two labeling schemes and two Gaussian distribution for labeling the nodes. Each labeling scheme represents a different graph class. In the first labeling scheme, i.e. for the graphs from class 0, we split



(a) Labeling schema for class 0. (b) Labeling schema for class 1.

Figure 2: The spatial distribution of node labels for the synthetic experiments in Section 5.

the 2D plane in half on the x-axis and for the nodes that are placed on the left side are labeled from Distribution A, otherwise they are labeled from Distribution B (see Figure 2(a)). In the second labeling scheme, i.e. for the graphs from class 1, the 2D plane is split by a square in the middle of the plane occupying the half area of the 2D plane and for the nodes that are placed within the square are labeled from Distribution A, otherwise they are labeled from Distribution B (Figure 2(b)).

We use two different edge construction methods. In the first variant, we ensure that nodes lie on a regular grid, and use a four-connected graph topology. In the second variant, we randomly sample the locations of the nodes, and construct the topology using a k-nearest neighbor graph construction. We performed 10-fold cross validation experiments with a fixed c value ($c = 0.01$) and accuracies are shown in Table 3.

6. Experiments on an fMRI analysis problem

We evaluate the *pyramid quantized Weisfeiler-Lehman graph representation* using real data from two different domains. The first evaluation described here comes from the fMRI analysis area, while the second described in Section 7 focus on a 3D shape mesh classification problem. We approach the fMRI analysis by representing fMRI recordings as graphs, and we use our proposed method to learn from the interconnections between voxels. Our approach has an enriched capacity to model such dependencies by considering interconnections between voxels which may be functionally important. [2]

6.1. Cocaine addiction dataset

We evaluate the approach on a dataset [47, 48, 49, 46, 50] that contains an approximately equal number of cocaine addicted individuals and control subjects performing a neuropsychological experiment of block design, called a drug Stroop experiment. The classification task is to discriminate cocaine from control subjects. The data were preprocessed using SPM2 [51] and a contrast map for each subject was produced. Only the subjects that complied to motion $<$ 2mm translation, $< 2^\circ$ rotation and at least 50% performance of the subject in an unrelated task [47] were kept.

6.2. Graph construction

We build a graph representation from the contrast maps by determining a subset of voxels with the use of the Elastic Net [41] on the raw voxel values. This method is particularly appropriate in fMRI where nearby voxels are likely to be correlated, and regions responsible for a given function or behavior distributes across multiple voxels. We have made use of k-nearest neighbor graphs on the voxels that were selected by an initial training of the Elastic Net. We symmetrize the k-nn relationship by considering the edges to indicate an undirected graph structure. While other models of connectivity are of interest [52, 53], we have found that the use of k-nearest neighbors to determine the graph topology yields good performance in general and illustrates the advantages of the pyramid Weisfeiler-Lehman approach. Furthermore, the subtree statistics considered here implicitly account for longer distance connections for sufficiently deep subtree patterns. We set the number of neighbors $k = 5$ in all experiments.

To enrich our graph representations of the fMRI contrast maps, we take advantage of the activation information. At each voxel selected by the Elastic Net for the construction of the graph, we label it with its activation. Since the activation has continuous values, our graph representation is transformed to a continuous labeled graph. We subsequently compute subgraph statistics over this graph to generate a feature vector, $\phi^{(l)}_{(h)}(G^{(l)})$ for a given height h of subtree patterns and a given quantization level l for a graph G. Finally, we use the Elastic Net on these subgraph statistics over all quantization levels, in order to determine our final prediction function, with a model selection step to determine appropriate values for λ_1 and λ_2.

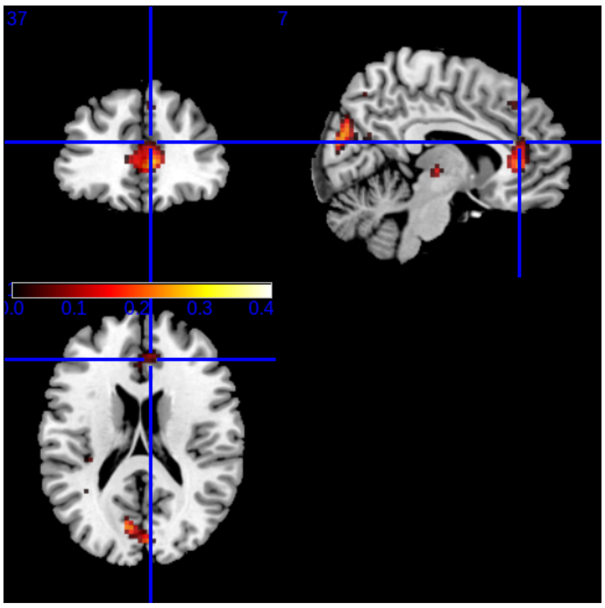
6.3. Results

We use the same experimental setup, a random splitting scheme with 50 trials, to estimate the classification performance of *pyramid quantized Weisfeiler-Lehman graph representation* and the baseline methods. In each trial, a random selection of 80% of the data are used for

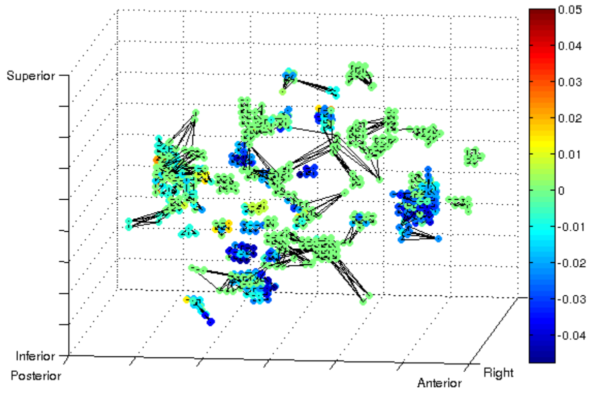
[2]This section is based in part on results originally published in [46].

Table 3: A comparison of the Weisfeiler-Lehman pyramid kernel with a graph matching based approach. The Weisfeiler-Lehman kernel is adaptable to different graph topologies, while graph matching can fail when the topology has significant local random variations as in the *k*-NN graph construction.

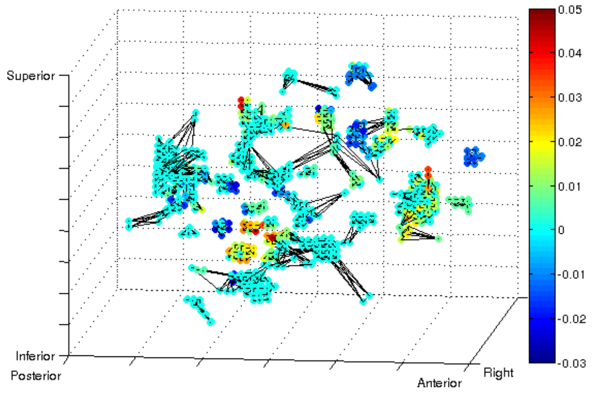
Experiment	**Graph Matching**	**Bag of Words**	**WLpyramid**				
			k=1	**k=2**	**k=3**	**k=4**	**k=5**
Grid	72.42 ± 1.35	48.08 ± 2.12	48.08 ± 2.24	53.33 ± 2.51	64.00 ± 2.55	63.67 ± 2.15	64.33 ± 2.26
***k*-NN**	51.32 ± 1.57	44.80 ± 0.68	82.22 ± 0.51	84.15 ± 0.31	85.77 ± 0.57	84.88 ± 0.58	84.60 ± 0.6



(a) A visualization of the areas of the brain selected by Elastic Net.



(b) Weisfeiler-Lehman - Control



(c) Weisfeiler-Lehman - Cocaine

Figure 3: A visualization of the areas of the brain selected by Elastic Net as well as a visualization of the learned functions on the quantized Weisfeiler-Lehman representation.

training, while the remaining 20% are used to estimate the performance. In Table 4 we show the performance of the *pyramid quantized Weisfeiler-Lehman graph representation* for four different depths of subtree patterns. Our approach achieves a mean accuracy of 64.28% for subtree patterns up to depth two, a significant improvement over the bag of words kernel ($h = 0$). We also compare our proposed technique with three other methods on the same dataset: (i) Gaussian kernel ridge regression, (ii) the Elastic Net with raw voxels as features, and (iii) the Elastic Net with raw voxels and *pyramid quantized Weisfeiler-Lehman* subtree features concatenated in a joint feature vector. In Figure 4 we show the mean accuracy of the final system and the standard error. The *pyramid quantized Weisfeiler-Lehman graph representation* outperforms the rest of the methods and with a Wilcoxon signed rank test we determine that our proposed method is statistically significantly better

Table 4: Mean accuracy on the cocaine addiction dataset of the *pyramid quantized Weisfeiler-Lehman graph representation* for four different subtree pattern depths, $h \in \{0, 1, 2, 3\}$. Maximum performance is achieved with subtree patterns up to depth two.

Pyramid Quantized Weisfeiler-Lehman				
h	**0**	**1**	**2**	**3**
Accuracy	54.00%	57.14%	**64.28%**	63.42%

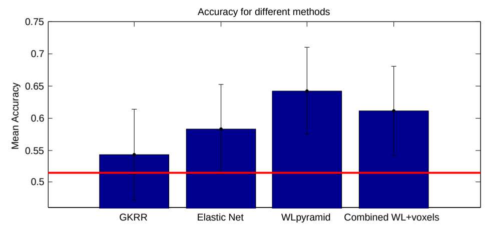


Figure 4: Mean accuracy and standard error on the cocaine addiction dataset. The compared methods are (left to right) Gaussian kernel ridge regression (GKRR), the Elastic Net on raw voxels, *pyramid quantized Weisfeiler-Lehman* (WLpyramid), and the Elastic Net with a concatenation of the raw voxels and the *pyramid quantized Weisfeiler-Lehman* features (Combined EN+WL). The horizontal red line indicates chance performance.

(p = 0.02). Additionally, a reduction of over 14% in classification error is recorded between the Elastic Net on the raw voxels and our method. Figure 3(a) shows the areas selected by the Elastic Net, while Figure 3(b) and Figure 3(c) show the visualizations of the learned functions for control and cocaine addicted subjects, respectively. Note that Elastic Net on the raw voxels selected the rostral anterior cingulate cortex, an important region for addictive behavior [49, 46].

Although our method works in an implicitly high dimensional space, we empirically observe that Elastic Net controls the complexity at each stage of the pipeline. The first learning step selects approximately 1100 voxels. Using our method, we generate a feature vector of length 6×10^5, but with a sparsity of $\sim 2\%$. The second application of Elastic Net selects only 2K dimensions. In each step, the method retains complexity much lower than a “simple” linear function over tens of thousands of voxels as in previous works.

Several broad observations are apparent from our quantitative results. From Table 4, we note that subtree patterns up to depth two performs best, and that deeper subtree patterns begin to reduce average performance. This indicates that the big-O complexity of the graph representation is only slightly higher than using a simple linear function as we use very small values of h in practice (see Section 3.3). The proposed method performs significantly better than the baselines (see Table 4 and Figure 4). In our final experiment of combining the raw voxel values with the subtree pattern features, we found that performance decreased slightly from that of only considering subtree pattern features.

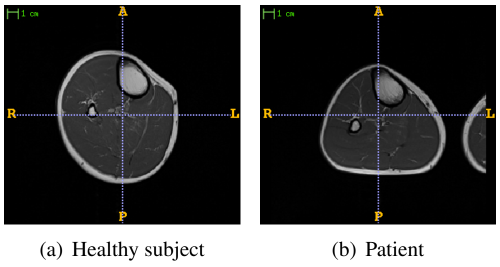


Figure 5: On the left a slice of the MR image from a healthy subject, and on the right a slice of the MR image from a patient with a neuromuscular disease.

7. Experiments on 3D shape classification

In this section, we evaluate the *pyramid quantized Weisfeiler-Lehman kernel* on two 3D shape categorization tasks. In our approach, we denote that the 3D shape can be viewed as graphs $G(V, E)$, where the finite set of points in the 3D space will represent the vertices V and the connection between two points in order to form triangles or curved surfaces will represent the edges E. This perspective specifies the topology of a graph, but does not explicitly encode relative vertex positions or other geometric properties. Therefore, we extend the notion of the graph to incorporate node labels that encode properties such as local curvature of the surface.

7.1. 3D shapes datasets

The evaluation is performed on two 3D shape datasets. The first dataset is the neuromuscular dystrophy dataset, which consists of 41 subjects: 27 are affected by a neuromuscular dystrophy, while the remaining 14 subjects are healthy [54, 55]. The subjects were imaged in the calf using a 1.5 T MRI scanner. An example of the T1-weighted MR images of the calf from a healthy and patient subject can be seen in Figure 5.

The T1 weighted MR images were manually segmented by an expert separating 7 important calf muscle groups and each segmented muscle is then transformed into a 3D surface mesh using the itk-snp program.[3] The

[3]`http://www.itksnap.org/pmwiki/pmwiki.php`

SHREC 2013 dataset consists of 20 classes of generic objects.[4] Each class contains 18 different large-scale models, resulting in a total of 360 objects.

7.2. Node label description

We select as labels on the vertices of the 3D surface mesh a weighted average over the principal curvature features of the immediately adjacent triangulated faces [56]. In the SHREC 2013 dataset, we also used as labels multi-viewpoint rendering features, a successful method for 3D shape classification [57], but only for the 3D mesh enclosed within a fixed radius. As we cannot assume a canonical basis for specifying the 3D coordinates of the surface control points, we use PCA to determine one. For comparison, we also develop a multi-viewpoint rendering baseline. Similarly, we use PCA to determine a basis. We then render images in these canonical bases and linear, polynomials of 2nd and 3rd degree and Gaussian kernels are computed.

7.3. Results

For both datasets we use the same experimental setup, a nested cross-validation procedure where the inner loop is used for automatic model selection. We only report results from the outer cross-validation procedure ensuring the statistical validity of our model selection. The performance is evaluated as the accuracy and the area under the ROC curve (AUC). We also compare our method with two other baselines on both datasets (*a*) a pyramid bag of words model and (*b*) a multi-viewpoint rendering procedure following the same experimental setup. We also present the results obtained from the combination of the multi-viewpoint rendering representation with the *pyramid quantized Weisfeiler-Lehman kernel*.

The performance on the neuromuscular dystrophy dataset is shown in Table 5. The *pyramid quantized Weisfeiler-Lehman kernel* outperforms both baseline methods. The overall best performance is achieved when we combined our method with the multi-viewpoint rendering approach with a accuracy of $\sim$ 83% and an AUC of 0.6648. The performance for the SHREC 2013 dataset is shown in Table 6. The overall best performance is achieved when we combined the *pyramid quantized Weisfeiler-Lehman kernel* with the multi-viewpoint rendering images with an AUC of $\sim$ 0.85 across all 20 classes. A Wilcoxon signed-rank test showed that the combined method performed better than all other methods with high statistical significance ($p < 10^{-3}$). We further show the learned weight of the *pyramid quantized Weisfeiler-Lehman kernel* for the SHREC 2013 dataset in Figure 6 and in Figure 7. Figure 6 shows the learned weights of the bird class for three different subtree depths ($h \in \{0, 1, 2\}$), while Figure 7 shows the learned weights for depth $h = 1$ for all classes. Note that the values of the learned weights increase as the color changes from blue to red.

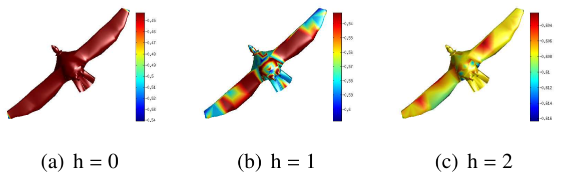

(a) h = 0 (b) h = 1 (c) h = 2

Figure 6: An example of the learned weights of the *pyramid quantized Weisfeiler-Lehman Kernel* of the class bird from the SHREC 2013 dataset for three different subtree depths $h \in \{0, 1, 2\}$. (Figure best viewed in color).

In the neuromuscular dystrophy dataset, the *pyramid quantized Weisfeiler-Lehman kernel* performs substantially better than both baselines. These techniques clearly contain complementary information as the combined method performs best. We have further confirmation from the SHREC 2013 dataset since the combined approach gives the best average performance with statistical significance.

8. Discussion

In this article, we have presented a fully automated, efficient and statistically sound framework for graph classification with continuous vector node labels using the *pyramid quantized Weisfeiler-Lehman graph representation*. It combines the Weisfeiler-Lehman graph kernel [13] with the pyramid match kernel [32], in order to learn from graphs with continuous labels through a pyramid quantization scheme. The method was evaluated on a wide variety of datasets and outperformed other machine learning techniques with statistical significance. We have demonstrated the utility of such graph representations in two main areas: fMRI analysis and 3D shape classification.

The fMRI application was evaluated on a real world cocaine addiction dataset and outperformed other methods with statistical significance, including kernel ridge regression and the Elastic Net. This validates our hypothesis for fMRI analysis: that the interconnections between voxels can contain additional information about brain structure that is not apparent in a linear function on the raw voxel values. The 3D shape classification

[4]`http://dataset.dcc.uchile.cl/`

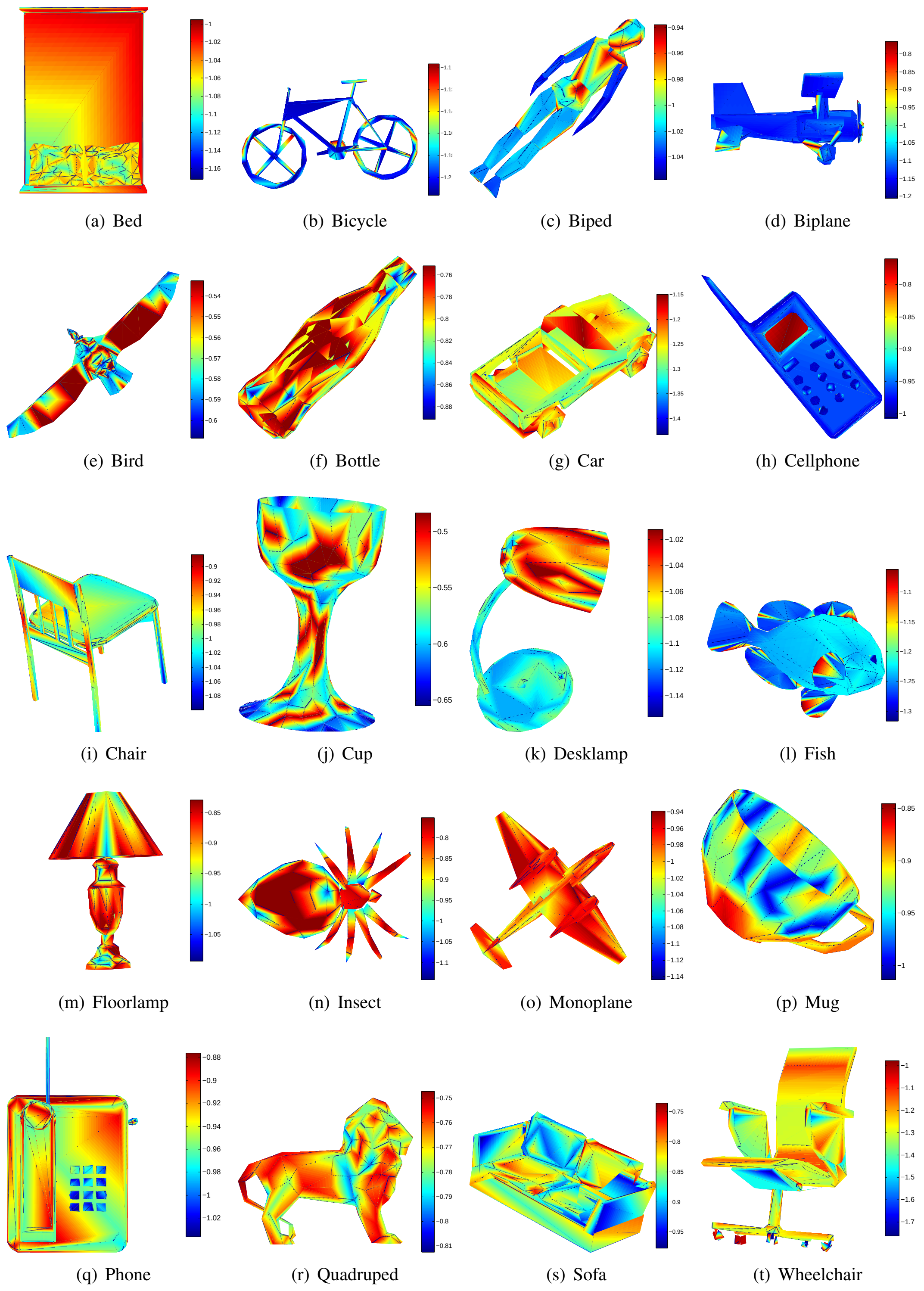


Figure 7: Visualization of the learned weights of the *pyramid quantized Weisfeiler-Lehman kernel* of subtree patterns with depth $h = 1$ of all classes for the SHREC2013 dataset. (Figure best viewed in color.)

	WLpyramid	pyramid BoW	Rendering	Combined
Accuracy	78.00%	73.00%	75.50%	82.93%
AUC	0.6410	0.6361	0.6300	0.6648

Table 5: The mean accuracy and the mean AUC on the neuromuscular dystrophy dataset. The compared methods are (left to right) the *pyramid quantized Weisfeiler-Lehman kernel* (WLpyramid), the pyramid bag of words model (pyramid BoW), the multi-viewpoint rendering images procedure (Rendering) and a combination of the multi-viewpoint rendering procedure with the *pyramid quantized Weisfeiler-Lehman kernel* (Combined).

Class	WLpyramid	pyramid BoW	Rendering	Combined
Bird	0.85	0.83	0.85	**0.86**
Bicycle	0.84	0.87	0.90	0.90
Biped	0.89	0.88	0.99	0.99
Biplane	0.60	0.63	0.68	**0.69**
Bird	0.73	0.73	0.80	0.80
Bottle	0.76	0.76	0.79	**0.80**
Car	0.78	0.79	0.80	0.80
CellPhone	0.74	0.80	0.88	**0.89**
Chair	0.69	0.68	0.70	**0.72**
Cup	0.85	0.84	0.88	0.88
Desklamp	0.80	0.80	0.88	**0.89**
Fish	1.00	1.00	1.00	1.00
Floorlamp	0.80	0.77	0.89	0.89
Insect	0.64	0.60	0.62	**0.66**
Monoplane	0.84	0.82	0.88	**0.90**
Mug	0.82	0.82	0.85	**0.87**
Phone	0.83	0.74	0.72	**0.83**
Quadruped	0.89	0.86	0.97	**0.98**
Sofa	0.76	0.75	0.74	**0.75**
Wheelchair	0.81	0.79	0.88	**0.90**
Average	0.80	0.79	0.84	0.85

Table 6: The mean AUC on the SHREC 2013 dataset. The compared methods are (left to right) the *pyramid quantized Weisfeiler-Lehman kernel* (WLpyramid), the pyramid bag of words model (pyramid BoW), the multi-viewpoint rendering procedure (Rendering), and a combination of the multi-viewpoint rendering procedure with the pyramid quantizes Weisfeiler-Lehman kernel (Combined). In bold are the one-vs-rest classifier where the combined classifier outperforms the multi-viewpoint rendering procedure.

tasks showed consistent improvement over baselines using the proposed method. We have not directly incorporated any features in our node labels capturing surface reflectance, color, or texture. This is an interesting area for future research. Learned shape retrieval by discriminative training of a Mahalanobis metric [58] is another interesting possible future direction that is directly applicable with the proposed representation.

We have made five main contributions in this work, (i) we applied a generalization of the Weisfeiler-Lehman graph kernel to continuous vector node labels, the *pyramid quantized Weisfeiler-Lehman kernel*, (ii) we showed that graph representations provide a rich family of functions for fMRI analysis, (iii) we developed a novel framework for shape classification based on the interpretation of shape meshes as annotated graphs, (iv) we performed experiments on fMRI analysis, medical imaging, and semantic shape classification tasks with improved classification accuracy across diverse applications and (v) we showed visualizations of the learned discriminant function, providing rich information about the discriminative power of the method.

Acknowledgments

This work was partially funded by ERC Grant 259112, FP7-MC-CIG 334380, NIA 1R21DA034954-01, and the AFM-Telethon foundation. We thank Jean-François Deux, Guillaume Bassez, Alain Rahmouni, and Thibault Varacca at Henri Mondor University Hospital, Créteil, France for collecting the muscle dataset. We thank Rita Goldstein at Icahn School of Medicine, Mount Sinai, and Dimitris Samaras and Jean Honorio at

Stony Brook University for collecting and preprocessing the fMRI dataset.

References

[1] D. M. Greig, B. T. Porteous, A. H. Seheult, Exact maximum a posteriori estimation for binary images, Journal of the Royal Statistical Society. Series B (Methodological) 51 (2) (1989) 271–279.

[2] L. Torresani, V. Kolmogorov, C. Rother, Feature correspondence via graph matching: Models and global optimization, in: D. Forsyth, P. Torr, A. Zisserman (Eds.), Computer Vision – ECCV 2008, Vol. 5303 of Lecture Notes in Computer Science, Springer Berlin Heidelberg, 2008, pp. 596–609. `doi:10.1007/978-3-540-88688-4_44`.

[3] B. Ng, V. Siless, G. Varoquaux, J.-B. Poline, B. Thirion, R. Abugharbieh, Connectivity-informed sparse classifiers for fMRI brain decoding, in: International Workshop on Pattern Recognition in NeuroImaging (PRNI), IEEE, 2012, pp. 101–104.

[4] J. Rao, R. Abugharbieh, G. Hamarneh, Adaptive regularization for image segmentation using local image curvature cues, in: K. Daniilidis, P. Maragos, N. Paragios (Eds.), Computer Vision – ECCV 2010, Vol. 6314 of Lecture Notes in Computer Science, Springer Berlin Heidelberg, 2010, pp. 651–665. `doi:10.1007/978-3-642-15561-1_47`.

[5] I. P. Goldstein, The genetic graph: A representation for the evolution of procedural knowledge, International Journal of Man-Machine Studies 11 (1) (1979) 51–77.

[6] A. A. Canutescu, A. A. Shelenkov, R. L. Dunbrack, A graph-theory algorithm for rapid protein side-chain prediction, Protein Science 12 (9) (2003) 2001–2014. `doi:10.1110/ps.03154503`.

[7] A. Wagner, D. A. Fell, The small world inside large metabolic networks, Proceedings of the Royal Society of London B: Biological Sciences 268 (1478) (2001) 1803–1810. `doi:10.1098/rspb.2001.1711`.

[8] J. Scott, Social Network Analysis: A Handbook, SAGE Publications, 2000.

[9] J. Gasteiger, T. Engel (Eds.), Chemoinformatics: A Textbook, 1st Edition, Wiley-VCH, 2003.

[10] D. Haussler, Convolution kernels on discrete structures, Tech. Rep. UCSC-CRL-99-10, University of California at Santa Cruz (1999).

[11] T. Gärtner, P. Flach, S. Wrobel, On graph kernels: Hardness results and efficient alternatives, in: B. Schölkopf, M. K. Warmuth (Eds.), Learning Theory and Kernel Machines, Vol. 2777 of Lecture Notes in Computer Science, Springer Berlin Heidelberg, 2003, pp. 129–143. `doi:10.1007/978-3-540-45167-9_11`.

[12] S. V. N. Vishwanathan, N. N. Schraudolph, R. Kondor, K. M. Borgwardt, Graph kernels, Journal of Machine Learning Research 11 (2010) 1201–1242.

[13] N. Shervashidze, P. Schweitzer, E. J. van Leeuwen, K. Mehlhorn, K. M. Borgwardt, Weisfeiler-Lehman graph kernels, Journal of Machine Learning Research 12 (2011) 2539–2561.

[14] Z. Harchaoui, F. Bach, Image classification with segmentation graph kernels, in: IEEE Conference on Computer Vision and Pattern Recognition, 2007. `doi:10.1109/CVPR.2007.383049`.

[15] P. Mahé, N. Ueda, T. Akutsu, J.-L. Perret, J.-P. Vert, Extensions of marginalized graph kernels, in: Proceedings of the Twenty-First International Conference on Machine Learning, 2004, pp. 552–559.

[16] H. L. Morgan, The Generation of a Unique Machine Description for Chemical Structures - A Technique Developed at Chemical Abstracts Service., Journal of Chemical Documentation 5 (2) (1965) 107–113.

[17] K. M. Borgwardt, H.-P. Kriegel, Shortest-path kernels on graphs, in: Proceedings of the Fifth IEEE International Conference on Data Mining, 2005, pp. 74–81.

[18] L. Ralaivola, S. J. Swamidass, H. Saigo, P. Baldi, Graph kernels for chemical informatics., Neural Networks 18 (8) (2005) 1093–1110.

[19] N. Shervashidze, S. V. N. Vishwanathan, T. Petri, K. Mehlhorn, K. M. Borgwardt, Efficient graphlet kernels for large graph comparison, in: Proceedings of the Twelfth International Conference on Artificial Intelligence and Statistics, 2009, pp. 488–495.

[20] T. Horváth, T. Gärtner, S. Wrobel, Cyclic pattern kernels for predictive graph mining, in: Proceedings of the tenth ACM SIGKDD international conference on Knowledge discovery and data mining, 2004, pp. 158–167.

[21] F. Costa, K. De Grave, Fast neighborhood subgraph pairwise distance kernel, in: Proceedings of the 26th International Conference on Machine Learning, 2010, pp. 255–262.

[22] J. Ramon, T. Gaertner, Expressivity versus efficiency of graph kernels, in: Proceedings of the First International Workshop on Mining Graphs, Trees and Sequences, 2003, pp. 65–74.

[23] P. Mahé, J.-P. Vert, Graph kernels based on tree patterns for molecules, Machine Learning 75 (1) (2009) 3–35.

[24] F. R. Bach, Graph kernels between point clouds, in: Proceedings of the 25th International Conference on Machine learning, 2008, pp. 25–32.

[25] B. Weisfeiler, A. Lehman, A reduction of a graph to a canonical form and an algebra arising during this reduction, Nauchno-Technicheskaya Informatsia 2 (9) (1968) 12–16.

[26] M. Neumann, N. Patricia, R. Garnett, K. Kersting, Efficient graph kernels by randomization, in: P. A. Flach, T. De Bie, N. Cristianini (Eds.), Machine Learning and Knowledge Discovery in Databases, Vol. 7523 of Lecture Notes in Computer Science, Springer Berlin Heidelberg, 2012, pp. 378–393. `doi:10.1007/978-3-642-33460-3_30`.

[27] K. Borgwardt, A. Feragen, N. Kasenburg, J. Petersen, M. de Bruijne, Scalable kernels for graphs with continuous attributes, in: Advances in Neural Information Processing Systems 26, 2013, pp. 216–224.

[28] Z. Harris, Distributional structure, Word 10 (3) (1954) 146–162.

[29] Y. Ko, A study of term weighting schemes using class information for text classification, in: Proceedings of the 35th international ACM SIGIR conference on Research and development in information retrieval, 2012, pp. 1029–1030.

[30] G. Qiu, Indexing chromatic and achromatic patterns for content-based colour image retrieval, Pattern Recognition 35 (8) (2002) 1675–1686. `doi:10.1016/S0031-3203(01)00162-5`.

[31] F.-F. Li, P. Perona, A bayesian hierarchical model for learning natural scene categories, in: IEEE Computer Society Conference on Computer Vision and Pattern Recognition, 2005, pp. 524–531. `doi:10.1109/CVPR.2005.16`.

[32] K. Grauman, T. Darrell, The pyramid match kernel: Efficient learning with sets of features, Journal of Machine Learning Research 8 (2007) 725–760.

[33] K. Grauman, T. Darrell, Approximate correspondences in high dimensions, in: B. Schölkopf, J. Platt, T. Hoffman (Eds.), Advances in Neural Information Processing Systems 19, MIT Press, 2007, pp. 505–512.

[34] J. H. Ward, Hierarchical Grouping to Optimize an Objective Function, Journal of the American Statistical Association

58 (301) (1963) 236–244.

[35] F. Odone, A. Barla, A. Verri, Building kernels from binary strings for image matching, IEEE Transactions on Image Processing 14 (2) (2005) 169–180.

[36] G. R. G. Lanckriet, T. D. Bie, N. Cristianini, M. I. Jordan, W. S. Noble, A statistical framework for genomic data fusion, Bioinformatics 20 (16) (2004) 2626–2635.

[37] A. Zien, C. S. Ong, Multiclass multiple kernel learning, in: Proceedings of the 24th international conference on Machine learning, 2007, pp. 1191–1198.

[38] S. Sonnenburg, G. Rätsch, C. Schäfer, B. Schölkopf, Large scale multiple kernel learning, Journal of Machice Learning Research 7 (2006) 1531–1565.

[39] A. Rakotomamonjy, F. R. Bach, S. Canu, Y. Grandvalet, SimpleMKL, Journal of Machine Learning Research 9 (2008) 2491–2521.

[40] A. Vedaldi, V. Gulshan, M. Varma, A. Zisserman, Multiple kernels for object detection, in: International Conference in Computer Vision, 2009, pp. 606–613.

[41] H. Zou, T. Hastie, Regularization and variable selection via the elastic net, Journal of the Royal Statistical Society Series B 67 (2) (2005) 301–320.

[42] O. Duchenne, A. Joulin, J. Ponce, A graph-matching kernel for object categorization, in: International Conference on Computer Vision, 2011.

[43] Z.-Y. Liu, H. Qiao, X. Yang, S. C. Hoi, Graph matching by simplified convex-concave relaxation procedure, International Journal of Computer Vision 109 (3) (2014) 169–186. `doi:10.1007/s11263-014-0707-7`.

[44] X. Yang, H. Qiao, Z. Liu, Outlier robust point correspondence based on GNCCP, Pattern Recognition Letters 55 (2015) 8–14.

[45] F. Zhou, Factorized graph matching, in: IEEE Conference on Computer Vision and Pattern Recognition, 2012, pp. 127–134.

[46] K. Gkirtzou, J. Honorio, D. Samaras, R. Goldstein, M. B. Blaschko, fmri analysis with sparse weisfeiler-lehman graph statistics, in: G. Wu, D. Zhang, D. Shen, P. Yan, K. Suzuki, F. Wang (Eds.), Machine Learning in Medical Imaging, Vol. 8184 of Lecture Notes in Computer Science, Springer, 2013, pp. 90–97. `doi:10.1007/978-3-319-02267-3_12`.

[47] R. Z. Goldstein, N. Alia-Klein, D. Tomasi, J. H. Carrillo, T. Maloney, P. A. Woicik, R. Wang, F. Telang, N. D. Volkow, Anterior cingulate cortex hypoactivations to an emotionally salient task in cocaine addiction, Proceedings of the National Academy of Sciences 106 (23) (2009) 9453–9458.

[48] J. Honorio, D. Tomasi, R. Z. Goldstein, H.-C. Leung, D. Samaras, Can a single brain region predict a disorder?, IEEE Transactions on Medical Imaging 31 (11) (2012) 2062–2072. `doi:10.1109/TMI.2012.2206047`.

[49] K. Gkirtzou, J. Honorio, D. Samaras, R. Goldstein, M. B. Blaschko, fMRI analysis of cocaine addiction using k-support sparsity, in: IEEE 10th International Symposium on Biomedical Imaging, 2013, pp. 1078–1081. `doi:10.1109/ISBI.2013.6556665`.

[50] E. Belilovsky, K. Gkirtzou, M. Misyrlis, A. B. Konova, J. Honorio, N. Alia-Klein, R. Z. Goldstein, D. Samaras, M. B. Blaschko, Predictive sparse modeling of fMRI data for improved classification, regression, and visualization using the k-support norm, Computerized Medical Imaging and Graphics`doi:10.1016/j.compmedimag.2015.03.007`.

[51] K. J. Friston, A. P. Holmes, K. J. Worsley, J. P. Poline, C. D. Frith, R. S. J. Frackowiak, Statistical parametric maps in functional imaging: A general linear approach, Human Brain Mapping 2 (4) (1994) 189–210.

[52] O. Sporns, Networks of the Brain, MIT Press, 2010.

[53] C.-Y. Wee, P.-T. Yap, W. Li, K. Denny, J. N. Browndyke, G. G. Potter, K. A. Welsh-Bohmer, L. Wang, D. Shen, Enriched white matter connectivity networks for accurate identification of MCI patients, NeuroImage 54 (3) (2011) 1812–1822.

[54] R. Neji, Diffusion tensor imaging of the human skeletal muscle: Contributions and applications, Ph.D. thesis, École Centrale Paris (2010).

[55] K. Gkirtzou, J.-F. Deux, G. Bassez, A. Sotiras, A. Rahmouni, T. Varacca, N. Paragios, M. B. Blaschko, Sparse classification with MRI based markers for neuromuscular disease categorization, in: G. Wu, D. Zhang, D. Shen, P. Yan, K. Suzuki, F. Wang (Eds.), Machine Learning in Medical Imaging, Vol. 8184 of Lecture Notes in Computer Science, Springer, 2013, pp. 33–40. `doi:10.1007/978-3-319-02267-3_5`.

[56] S. Rusinkiewicz, Estimating curvatures and their derivatives on triangle meshes, in: Proceedings of the 3D Data Processing, Visualization, and Transmission, 2nd International Symposium, 2004, pp. 486–493. `doi:10.1109/3DPVT.2004.54`.

[57] R. Ohbuchi, T. Furuya, Distance metric learning and feature combination for shape-based 3d model retrieval, in: Proceedings of the ACM workshop on 3D object retrieval, 2010, pp. 63–68.

[58] K. Weinberger, L. Saul, Distance metric learning for large margin nearest neighbor classification, The Journal of Machine Learning Research 10 (2009) 207–244.