Vertex-Weighted Hypergraph Learning for Multi-View Object Classification

Lifan Su, Yue Gao*, Xibin Zhao*, Hai Wan, Ming Gu, Jiaguang Sun
Key Laboratory for Information System Security, Ministry of Education
Tsinghua National Laboratory for Information Science and Technology
School of Software, Tsinghua University, China.
{sulifan,gaoyue,zxb,wanhai,guming,sunjg}@tsinghua.edu.cn
* indicates corresponding authors

Abstract

3D object classification with multi-view representation has become very popular, thanks to the progress on computer techniques and graphic hardware, and attracted much research attention in recent years. Regarding this task, there are mainly two challenging issues, *i.e.*, the complex correlation among multiple views and the possible imbalance data issue. In this work, we propose to employ the hypergraph structure to formulate the relationship among 3D objects, taking the advantage of hypergraph on high-order correlation modelling. However, traditional hypergraph learning method may suffer from the imbalance data issue. To this end, we propose a vertex-weighted hypergraph learning algorithm for multi-view 3D object classification, introducing an updated hypergraph structure. In our method, the correlation among different objects is formulated in a hypergraph structure and each object (vertex) is associated with a corresponding weight, weighting the importance of each sample in the learning process. The learning process is conducted on the vertex-weighted hypergraph and the estimated object relevance is employed for object classification. The proposed method has been evaluated on two public benchmarks, *i.e.*, the NTU and the PSB datasets. Experimental results and comparison with the state-of-the-art methods and recent deep learning method demonstrate the effectiveness of our proposed method.

1 Introduction

Recent advances on computer techniques and graphic hardware have prompted wide applications of 3D objects in various domains [Bimbo and Pala., 2006], such as architecture design, entertainment and the medical industry. Confronting the increasing 3D big data, effective 3D object classification and retrieval techniques [Guo *et al.*, 2014; Chen and Bhanu, 2009] have become an urgent requirement for both the research society and industrial practice. Recently, multi-view based object representation [Chen *et al.*, 2003] has attracted much attention due to its superior performance on visual content description of 3D objects. In multi-view 3D object classification, each object is represented by multiple images, and Figure 1 provides three examples of multi-views of 3D objects. In this work, we focus on multi-view 3D object classification.

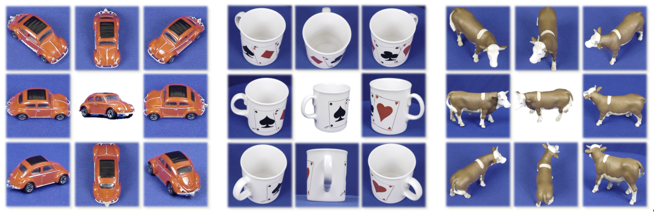


Figure 1: Three examples of multi-views of 3D objects.

Existing methods [Wang *et al.*, 2012b; Gao *et al.*, 2012] proposed to train classifiers, or employ graph/manifold structure to predict the object category. Although there have been much attention on this task, it is still an challenging task. The main challenges of multi-view 3D object classification are two-fold, *i.e.*, the relationship modeling among different objects and the possible imbalance training data, which also exists in other applications [Shao *et al.*, 2014]. First, each 3D object is represented by a set of images, leading to much complicated comparison between 3D objects. How to formulate the correlation among 3D objects with multiple views is a hard task. Second, the 3D data from different categories may vary significantly. Here we take the popular Princeton Shape Benchmark (PSB) [Shilane *et al.*, 2004] as an example. With average 11.26 3D objects for each category, the standard deviation of the number of samples for each category is 13.02, indicating a highly imbalance issue. Existing works may work well when the training data for different data categories are comparable but may degrade the performance when they are not.

To formulate the complex correlation among 3D objects, we propose to employ the hypergraph structure for data modelling. In recent years, hypergraph learning [Zhou *et al.*, 2007] has shown superior performance in many computer vision tasks, such as image retrieval [Huang *et al.*, 2010], object segmentation [Huang *et al.*, 2009] and classification [Gao *et al.*, 2012]. However, traditional hypergraph structure cannot handle the data imbalance issue, which can degrade the performance of hypergraph modeling. To this end, we propose a vertex-weighted hypergraph learning algorithm for multi-view 3D object classification, which introduces an updated hypergraph structure considering the vertex weights.

In our method, the correlation among 3D objects is formulated in a hypergraph structure, and the weights for both vertices and hyperedges are associated with the hypergraph. The vertex weights are used to define the influence of different samples on the learning process, and the hyperedge weights are used to generate optimal representation. The learning process is conducted on the vertex-weighted hypergraph and the learned object relevance is used for classification. The proposed method has been evaluated on two public benchmarks, *i.e.*, the National Taiwan University (NTU) 3D model dataset [Chen *et al.*, 2003] and the Princeton Shape Benchmark (PSB) [Shilane *et al.*, 2004]. Experimental results and comparison with the state-of-the-art methods demonstrate the effectiveness of our proposed method.

2 Related Work

On multi-view object classification, a manifold-to-manifold distance (MMD) was introduced in [Wang *et al.*, 2012b]. Each group of images can be represented in three levels, *i.e.*, point, subspace and manifold. In this method, the compared multi-views of objects were formulated by manifolds and the corresponding manifold-to-manifold distance was calculated to measure the distance between two groups of images. Huang *et al.* [Huang *et al.*, 2014] introduced a hybrid metric learning approach to jointly employ heterogenous statistics, *i.e.*, mean, covariance matrix and Gaussian distribution, multi-view classification. Gao *et al.* [Gao *et al.*, 2012] proposed to employ the hypergraph structure to formulate the relationship among different objects, where the connection among objects were built using view clustering. In this work, multiple hypergraphs were constructed and the relevance among objects and the hypergraph weights were jointly learned. A covariance discriminative learning (CDL) method was introduced in [Wang *et al.*, 2012a]. In this method, the covariance matrix of multiple views was extracted for object representation and the linear discriminant analysis was conducted to measure the distance between two sets of views. Huang *et al.* [Huang *et al.*, 2015] proposed a Log-Euclidean metric learning (LEML) method, which learned a Log-Euclidean metric to transform the matrix logarithms from the raw tangent space of multiple views to a discriminative tangent space for the objects. Regarding the representation of multi-view objects, much research attention has been attracted. Ji *et al.* [Ji *et al.*, 2014] proposed a compact bag-of-patterns descriptor (CBoP) to mine the discriminative visual patterns from 3D point clouds, which innovatively alleviates the ill-posed pattern configurations from 2D images. It is a breakthrough that such descriptor well address both compactness and discriminativity in multi-view representation.

In recent years, there have been successful and serial achievements [Tao *et al.*, 2006; 2007; 2009; Yu *et al.*, 2016] on learning optimal subspace or metric for data modeling. Other works [Liu *et al.*, 2015; Xu *et al.*, 2015; Shao *et al.*, 2016; Liu *et al.*, 2017] further explore the multi-task, multi-view learning algorithms. Recently, hypergraph learning has been widely applied in many applications. Huang *et al.* [Huang *et al.*, 2010] proposed to employ the hypergraph structure to formulate the relationship among images. In this work, only the vertex relevance was learned for image ranking. Huang *et al.* [Huang *et al.*, 2009] also proposed to use the hypergraph structure to represent the spatial-temporal neighborhood correlation among image patches. In this method, hypergraph cut was used for object segmentation in videos. Existing hypergraph learning methods only focus on learning the vertex correlation only or joint learning the vertex relevance and the hyperedge weights simultaneously. There is no effort targeting on considering the vertex weighting issue, which is an important direction to further improve the representation ability of hypergraph.

3 The Proposed Method

In this section, we introduce our vertex-weighted hypergraph learning for multi-view 3D object classification. First, the features are extracted for all images of the 3D objects, and then the pairwise 3D object distance is measured. Based on the distance, a hypergraph is constructed, where the vertex weights are calculated based on the distribution of training samples for each class respectively. Then, the learning process is conducted on the hypergraph to estimate the optimal relevance of each 3D object to each class and the hyperedge weight simultaneously.

3.1 View Feature Extraction and Pairwise Object Distance Measure

For n 3D objects in the database $\{\mathcal{O}_1, \mathcal{O}_2, \dots, \mathcal{O}_n\}$, each object $\mathcal{O}_i$ is represented by a set of views $\{v_{i1}, v_{i2}, \dots, v_{in_i}\}$. Here two features, *i.e.*, Zernike moment and histogram of oriented gradient (HOG), which have been widely used in 3D object analysis tasks, are employed to represent each view in this work. We note that the employed features are flexible and can be replaced/expanded based on the hypergraph structure.

Given two sets of images and corresponding features, we employ the Hausdorff distance to measure the 3D object distance between two objects $\mathcal{O}_1$ and $\mathcal{O}_2$. It is noted that other distance measures can be also used here. As we have two types of features for each view, there are two distances for each pair of 3D objects based on Zernike moment and HOG, respectively.

3.2 Hypergraph Construction

Given the 3D object pairwise distances, the relationship among the n_t training objects and the testing object is formulated in a hypergraph structure $\mathcal{G} = (\mathcal{V}, \mathcal{E}, \mathbf{W})$. In $\mathcal{G}$, $\mathcal{V}$ is the vertex set, where each vertex denotes one object, and there are $n_t + 1$ vertices in total. $\mathcal{E}$ is the hyperedge set and $\mathbf{W}$ is the matrix of hyperedge weights. To generate the connection among vertices, *i.e.*, hyperedges, each time one vertex is selected as the centroid, and one hyperedge is constructed to connect the centroid and its K nearest neighbors in the corresponding feature space. This process repeats twice for the two types of features and generates $2(n_t + 1)$ hyperedges for $\mathcal{G}$.

An incidence matrix $\mathbf{H}$ is then generated to represent the relationship among different vertices. The (a, b)-th entry of the incidence matrix $\mathbf{H}$ indicates whether the a-th vertex is connected via the b-th hyperedge to other vertices. The incidence matrix $\mathbf{H}$ of hypergraph $\mathcal{G} = (\mathcal{V}, \mathcal{E}, \mathbf{W})$ is generated as $\mathbf{H}(v, e) = \begin{cases} 1 & \mathit{if}\, v \in e \\ 0 & \mathit{if}\, v \notin e \end{cases}$. In traditional hypergraph structure, all the vertices are regarded as equal weights. We note that different training samples could have varied importance and those categories with too many training samples may dominate the classification process. Existing hypergraph learning methods fail to deal with the imbalance data issue. To this end, we propose to a vertex weighting matrix $\mathbf{U}$ to weight different vertices, where $\mathbf{U}(v)$ is the vertex weight of vertex v.

Here, the vertex degree of the a-th vertex $v_a \in \mathcal{V}$ and hyperedge degree of the b-th hyperedge $e_b \in \mathcal{E}$ are calculated, respectively, as $d(v_a) = \sum_{e \in \mathcal{E}} \mathbf{W}(e)\mathbf{H}(v_a, e)$, and $\delta(e_b) = \sum_{v \in \mathcal{V}} \mathbf{U}(v)\mathbf{H}(v, e_b)$. We note that different from traditional hypergraph structure, where the hyperedge degree is fully determined by the connection from $\mathbf{H}$, the hyperedge degree δ here takes both the connections on $\mathbf{H}$ and the corresponding vertex weights into consideration simultaneously.

Now, two diagonal matrices $\mathbf{D}_v$ and $\mathbf{D}_e$ can be generated with every entry along the diagonal corresponds to the vertex degree and hyperedge degree, respectively. Note that all the hyperedges are initialized with an equal weight, *e.g.*, 1.

Traditional hypergraph-based methods do not take the importance of vertices into consideration, which may degrade the representative ability. In our method, we introduce the vertex importance in the hypergraph structure. The main idea for vertex weighting is to enhance the samples that may convey discriminative information and weaken the samples which may be redundancy, such as repeated/closed training samples, and bring in little information but much bias. Before conducting hypergraph learning, we first initialize the vertex weights.

For the ith category, here we let $\{\mathcal{O}_{i1}^t, \mathcal{O}_{i2}^t, \ldots, \mathcal{O}_{im}^t\}$ denote all m training data. We can then measure all the pairwise distances $d_o(\mathcal{O}_{ia}^t, \mathcal{O}_{ib}^t)$ between each two objects based on the two features, respectively. The mean distance from $\mathcal{O}_{ia}^t$ to all other training samples for the ith category can calculated as $\overline{d}(O_{ia}^t) = \frac{1}{m} \sum\limits_{b=1}^{m} \sum\limits_{Zernike,HOG} d_o(O_{ia}^t, O_{ib}^t)$. The vertex weight for the training sample $\mathcal{O}_{ia}^t$ can be written as $U(O_{ia}^t) = \frac{\overline{d}(O_{ia}^t)}{\sum\limits_{b=1}^{m} \overline{d}(O_{ib}^t)}$.

These vertex weighs are further normalized for each class respectively. This initialization approach can provide higher weights to the samples which are not close to others and lower weights to the samples which are similar to others. In this way, the repeated/close samples will have relatively smaller influence and the individual samples could becomes more important on the hypergraph learning process.

3.3 Vertex-Weighted Hypergraph Learning

Given the hypergraph, the relationship among vertices can be modeled based on how they are connected via hyperedges. The relevance matrix $\mathbf{F}$ for hypergraph $\mathcal{G} = (\mathcal{V}, \mathcal{E}, \mathbf{W})$ is the to-be-learned matrix, which indicates the relevance of each object to the categories. In this work, it can be learned from a joint optimization process considering the hypergraph structure regularizer, the empirical loss based on labeled data, and the hyperedge weights simultaneously.

Hypergraph Structure Regularizer

As we have a new structure with vertex weights compared with traditional hypergraph, we need to define the new hypergraph ructure regularizer first. Generally, the more two vertices are connected via hyperedges, the more similar their relevance scores are and vice versa. And, the higher weights of the two vertices, the higher cost will be provided. Based on this, the hypergraph structure regularizer $\Omega(\mathbf{F})$ can be written as

$$
\begin{aligned}
\Omega(\mathbf{F}) = & \sum_{k=1}^{n_{\text{cate}}} \sum_{e \in \mathcal{E}} \sum_{u,v \in \mathcal{V}} \\
& \frac{\mathbf{W}(e)\mathbf{U}(u)\mathbf{H}(u,e)\mathbf{U}(v)\mathbf{H}(v,e)}{2\delta(e)} \left(\frac{\mathbf{F}(u,k)}{\sqrt{d(u)}} - \frac{\mathbf{F}(v,k)}{\sqrt{d(v)}} \right)^2 \\
= & \sum_{k=1}^{n_{\text{cate}}} \sum_{e \in \mathcal{E}} \sum_{u,v \in \mathcal{V}} \\
& \frac{\mathbf{W}(e)\mathbf{U}(u)\mathbf{H}(u,e)\mathbf{U}(v)\mathbf{H}(v,e)}{\delta(e)} \left(\frac{\mathbf{F}(u,k)^2}{d(u)} - \frac{\mathbf{F}(u,k)\mathbf{F}(v,k)}{\sqrt{d(u)d(v)}} \right) \\
= & \sum_{k=1}^{n_{\text{cate}}} \left\{ \sum_{u \in \mathcal{V}} \mathbf{U}(u)\mathbf{F}(u,k)^2 \sum_{e \in \mathcal{E}} \frac{\mathbf{W}(e)\mathbf{H}(u,e)}{d(u)} \sum_{v \in \mathcal{V}} \frac{\mathbf{H}(v,e)\mathbf{U}(v)}{\delta(e)} \right. \\
& \left. - \sum_{e \in \mathcal{E}} \sum_{u,v \in \mathcal{V}} \frac{\mathbf{F}(u,k)\mathbf{U}(u)\mathbf{H}(u,e)\mathbf{W}(e)\mathbf{H}(v,e)\mathbf{U}(v)\mathbf{F}(v,k)}{\sqrt{d(u)d(v)}\delta(e)} \right\} \\
= & \sum_{k=1}^{n_{\text{cate}}} \left\{ \sum_{u \in \mathcal{V}} \mathbf{U}(u)\mathbf{F}(u,k)^2 \right. \\
& \left. - \sum_{e \in \mathcal{E}} \sum_{u,v \in \mathcal{V}} \frac{\mathbf{F}(u,k)\mathbf{U}(u)\mathbf{H}(u,e)\mathbf{W}(e)\mathbf{H}(v,e)\mathbf{U}(v)\mathbf{F}(v,k)}{\sqrt{d(u)d(v)}\delta(e)} \right\} \\
= & \sum_{k=1}^{n_{\text{cate}}} \mathbf{F}(:,k)^{\mathrm{T}} \mathbf{\Delta} \mathbf{F}(:,k) = \mathbf{F}^{\mathrm{T}} \mathbf{\Delta} \mathbf{F}
\end{aligned}
\tag{1}
$$

where $\mathbf{F}$ is the to-be-learned relevance matrix, $\mathbf{F}(:,k)$ is the k-th column of $\mathbf{F}$, n_{cate} is the number of 3D object categories, and $\mathbf{\Delta} = \mathbf{U} - \mathbf{\Theta} = \mathbf{U} - {\mathbf{D}_v}^{-\frac{1}{2}} \mathbf{UHW}{\mathbf{D}_e}^{-1}\mathbf{H}^{\mathrm{T}}\mathbf{U}{\mathbf{D}_v}^{-\frac{1}{2}}$ is called vertex-weighted hypergraph Laplacian.

Here we note that the hypergraph Laplacian for traditional hypergraph is $\mathbf{\Delta}_t = \mathbf{I} - {\mathbf{D}_v}^{-\frac{1}{2}}\mathbf{HW}{\mathbf{D}_e}^{-1}\mathbf{H}^{\mathrm{T}}{\mathbf{D}_v}^{-\frac{1}{2}}$ [Zhou *et al.*, 2007]. By comparing $\mathbf{\Delta}$ and $\mathbf{\Delta}_t$, we can notice that the vertex-weighted hypergraph Laplacian takes the weights of different vertices into account when measuring the cost on hypergraph structure.

Empirical Loss

The empirical loss term is defined similarity as traditional hypergraphs by $\mathcal{R}_{emp}(\mathbf{F}) = \sum\limits_{k=1}^{n_{\text{cate}}} \|\mathbf{F}(:,k) - \mathbf{Y}(:,k)\|^2$, where $\mathbf{Y} \in \mathbb{R}^{n \times n_{\text{cate}}}$ is the label matrix and each of its entries denotes the category a subject belongs to, and $\mathbf{Y}(:,k)$ is the k-th column of $\mathbf{Y}$. If the a-th vertex belongs to the first category, then the $(a,1)$-th and $(a,2)$-th entries in $\mathbf{Y}$ are set to 1 and 0, respectively.

Hyperedge Weights

To generate better hypergraph representation, different hyperedges should have different influence, and an optimally learned hyperedge weight can help to improve the representation ability of the hypergraph structure. Based on this, we further learn the weights of hyperedges by adding a hyperedge weight regularizer $\mathrm{t}r(\mathbf{W}^T\mathbf{W})$.

Here, the learning task on vertex-weighted hypergraph is composed of three contents, *i.e.*, the hypergraph structure regularizer, the empirical loss and the hyperedge weight regularizer. We can have the following overall cost function:

$$\begin{aligned}&\mathcal{Q}(\mathbf{F},\mathbf{W}) = \Omega(\mathbf{F},\mathbf{W}) + \lambda\Re_{emp}(\mathbf{F}) + \mu tr(\mathbf{W}^T\mathbf{W})\\&= \mathbf{F}^{\mathrm{T}}\left(\mathbf{U} - {\mathbf{D}_v}^{-\frac{1}{2}}\mathbf{UHWD}_e{}^{-1}\mathbf{H}^{\mathrm{T}}\mathbf{UD}_v{}^{-\frac{1}{2}}\right)\mathbf{F}\\&+\lambda\sum_{k=1}^{n_{\text{it}}}\|\mathbf{F}(:,k) - \mathbf{Y}(:,k)\|^2 + \mu\mathrm{tr}(\mathbf{W}^T\mathbf{W})\end{aligned} \tag{2}$$

The objective function for the learning task on vertex-weighted hypergraph can be written as

$$\begin{aligned}&\mathrm{argmin}_{\mathbf{F},\mathbf{W}}\ \mathcal{Q}(\mathbf{F},\mathbf{W})\\&\text{s.t. } \mathbf{W}(e) \geq 0,\ \sum_{e\in E}\mathbf{H}(v,e)\,\mathbf{W}(e) = \mathbf{D}_v(v)\end{aligned} \tag{3}$$

3.4 Solution

To solve the optimization task in Eq. (3), we conduct an alternative strategy. We first fix $\mathbf{W}$, and optimize $\mathbf{F}$ as

$$\arg\min_{\mathbf{F}}\left\{\Omega(\mathbf{F}) + \lambda\mathcal{R}_{emp}(\mathbf{F})\right\}. \tag{4}$$

The objective function in Eq. 4 can be rewritten as

$$\arg\min_{\mathbf{F}}\mathbf{F}^{\mathrm{T}}\boldsymbol{\Delta}\mathbf{F} + \lambda\sum_{k=1}^{n_{\text{cate}}}\|\mathbf{F}(:,k) - \mathbf{Y}(:,k)\|^2. \tag{5}$$

According to [Zhou *et al.*, 2007], $\mathbf{F}$ can be solved by $\mathbf{F} = \left(\mathbf{I} + \frac{1}{\lambda}(\mathbf{I} - \boldsymbol{\Theta})\right)^{-1}\mathbf{Y}$. We then fix $\mathbf{F}$, and optimize $\mathbf{W}$ as

$$\begin{aligned}&\mathrm{argmin}_{\mathbf{W}}\ \Omega(\mathbf{W}) + \mu tr(\mathbf{W}^T\mathbf{W})\\&\text{s.t. } \mathbf{W}(e) \geq 0,\ \sum_{e\in\mathcal{E}}\mathbf{H}(v,e)\,\mathbf{W}(e) = \mathbf{D}_v(v)\end{aligned} \tag{6}$$

The objective function in Eq. (3) can be rewritten as

$$\begin{aligned}&\mathrm{argmin}_{\mathbf{W}}\ \mathbf{F}^{\mathrm{T}}\left(\mathbf{U} - {\mathbf{D}_v}^{-\frac{1}{2}}\mathbf{UHWD}_e{}^{-1}\mathbf{H}^{\mathrm{T}}\mathbf{UD}_v{}^{-\frac{1}{2}}\right)\mathbf{F}\\&+\mu\mathrm{tr}(\mathbf{W}^T\mathbf{W})\\&\text{s.t. } \mathbf{W}(e) \geq 0,\ \sum_{e\in\mathcal{E}}\mathbf{H}(v,e)\,\mathbf{W}(e) = \mathbf{D}_v(v)\end{aligned} \tag{7}$$

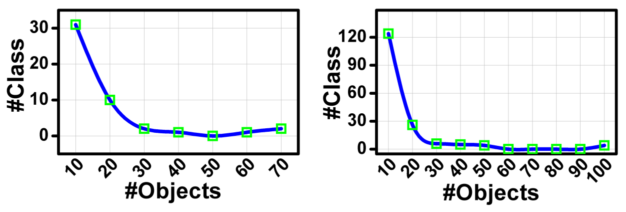
The above optimization task is convex on $\mathbf{W}$ and can be solved via quadratic programming. The above optimization procedure repeats until convergence. As the objective function decreases in each step and has a lower bound of 0, the convergence can be guaranteed.

Based on the learned $\mathbf{F}$, the relevance of the testing 3D object to 3D categories can be obtained. The testing data can be classified to the category with highest relevance value.

4 Experiments

4.1 Testing Datasets and Settings

We have conducted experiments on the National Taiwan University (NTU) 3D model dataset [Chen *et al.*, 2003] and the Princeton Shape Benchmark (PSB) [Shilane *et al.*, 2004]. The NTU dataset consists of 549 3D objects from 46 categories, *e.g.*, *bed*, *bike*, *boat*, and *table*. The PSB dataset contains 1,814 3D models from 161 categories. For each 3D object, 60 virtual cameras are employed to capture multiple views, which are located at the vertices of a polyhedron with the same structure with Buckminsterfullerene (C60). Thus, each 3D object has a group of 60 images. The distributions of sample numbers for each class on the two datasets are shown in Figure 2, from which we can notice that the difference among classes is significant.



(a) On the NTU dataset. (b) On the PSB dataset.

Figure 2: The distributions of sample numbers for each class on the two datasets.

For these two datasets, we further generate three subsets by removing small categories. Here, the standard deviation std of the number of samples for different categories is used to measure the class imbalance. The number of object category n_{cate} and the number of objects n_{all} for each subset and corresponding std are provided in the top rows of Table 1 and Table 2, from where we can notice large variance for the original datasets and corresponding subsets.

In the classification task, 10% to 80% samples for each class are randomly selected as the training data and all left samples are used as the testing data. This procedure repeats 10 times and the average classification accuracy is reported.

The following methods are employed for comparison.

1. Manifold-to-Manifold distance (MMD) [Wang *et al.*, 2012b].
2. Covariance Discriminative Learning with Partial Least Squares (CDL_PLS) [Wang *et al.*, 2012a].
3. Log-Euclidean Metric Learning (LEML) [Huang *et al.*, 2015].
4. Traditional hypergraph learning (HL) [Zhou *et al.*, 2007].
5. Hypergraph learning with hyperedge weight update (HL-E) [Gao *et al.*, 2012].
6. Vertex-weighted hypergraph learning (V-HL), *i.e.*, the proposed method without hyperedge weight update.
7. Vertex-weighted hypergraph learning with hyperedge weight update (V-HL-E), *i.e.*, the proposed method. In all hypergraph-based methods, the number of selected neighbors for hyperedge generation is set as 10, λ is set as 10, and μ is set as 1, respectively.

Here, MMD, CDL_PLS and LEML are the state-of-the-art methods on object classification based on multi-views. HL and HL-E are traditional hypergraph learning methods for comparison.

4.2 Experimental Results

Experimental results are demonstrated in Figure 3 and Figure 4, respectively. We also compare our V-HL method with traditional HL method on the subsets for the NTU and the PSB datasets, respectively, and the results are demonstrated in Table 1 and Table 2. From these results, we can have the following observations.

Table 1: Experimental comparison between HL and V-HL on the NTU subsets in terms of accuracy (%).

Training	NTU		Subset 1		Subset 2		Subset 3	
	$n_{\text{cate}} = 50$		$n_{\text{cate}} = 16$		$n_{\text{cate}} = 8$		$n_{\text{cate}} = 6$	
	$n_{\text{all}} = 549$		$n_{\text{all}} = 401$		$n_{\text{all}} = 291$		$_{\text{all}} = 256$	
	std $= 14.21$		std $= 17.85$		std $= 19.71$		std $= 18.82$	
	HL	V-HL	HL	V-HL	HL	V-HL	HL	V-HL
10%	35.82	40.63	44.16	48.56	59.79	62.62	68.17	68.43
20%	38.53	45.87	47.90	57.74	63.48	72.05	71.51	75.07
30%	44.31	51.54	57.21	64.77	69.93	77.77	74.92	82.57
40%	47.78	53.40	61.51	69.29	73.35	80.89	78.76	84.17
50%	50.49	56.58	64.09	69.75	78.06	82.04	82.29	84.81
60%	50.77	58.17	66.20	71.59	78.83	84.75	83.29	85.10
70%	51.13	59.77	69.04	74.27	81.88	85.54	83.33	89.01
80%	53.15	60.87	71.92	77.56	81.85	85.48	86.67	87.50

Table 2: Experimental comparison between HL and V-HL on the PSB subsets in terms of accuracy (%).

Training	PSB		Subset 1		Subset 2		Subset 3	
	$n_{\text{cate}} = 161$		$n_{\text{cate}} = 51$		$n_{\text{cate}} = 30$		$n_{\text{cate}} = 18$	
	$n_{\text{all}} = 1813$		$n_{\text{all}} = 1168$		$n_{\text{all}} = 898$		$n_{\text{all}} = 679$	
	std $= 13.02$		std $= 18.26$		std $= 21.21$		std $= 24.60$	
	HL	V-HL	HL	V-HL	HL	V-HL	HL	V-HL
10%	29.83	34.00	35.38	46.27	42.86	56.00	53.64	66.46
20%	34.19	37.88	48.61	54.70	58.83	63.82	67.92	74.81
30%	37.32	40.59	54.13	59.80	64.90	67.48	73.80	78.10
40%	39.70	44.20	57.19	62.88	68.39	70.27	77.59	79.89
50%	43.58	47.61	61.32	64.87	71.33	72.25	80.89	82.08
60%	43.93	49.60	63.29	67.35	73.04	74.50	81.72	83.69
70%	45.89	50.27	64.54	67.41	73.65	75.73	83.95	84.35
80%	45.92	53.18	66.17	69.16	75.31	76.80	84.31	85.03

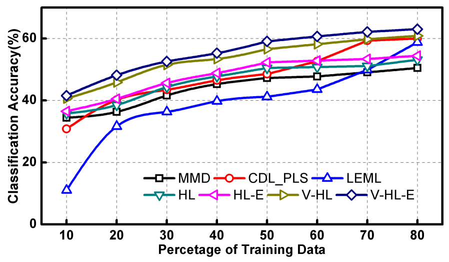


Figure 3: Comparison of different methods on NTU.

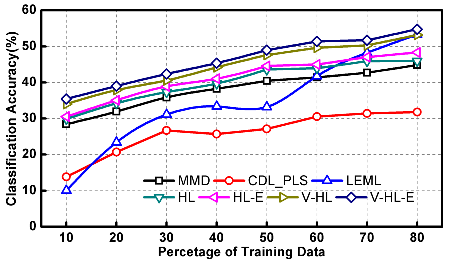


Figure 4: Comparison of different methods on PSB.

1. The proposed V-HL and V-HL-E methods outperform all other compared approaches. Here we take the results with 20% data for training as an example. V-HL-E achieves gains of 72.37%, 19.78%, and 52.10% compared with MMD, CDL_PLS and LEML on the NTU dataset, where the improvements are 22.23%, 88.43%, and 66.74% on the PSB dataset, respectively.

2. The vertex weights on hypergraph leads to improvement of the classification performance. On the NTU dataset, V-HL outperforms HL by 13.4%, 16.3%, 12.1%, and 16.9% when 10%, 30%, 50%, and 70% samples are used as training data, respectively. Similar results can be observed when compare V-HL-E with HL-E. These results can justify that the proposed vertex-weighted hypergraph structure can have better representation for object relevance modeling.

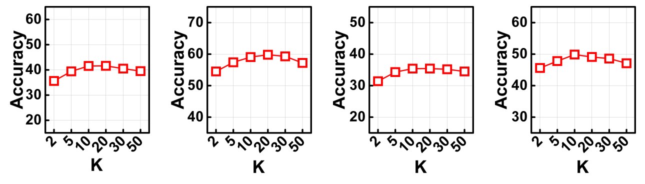
3. Hyperedge weight learning can improve the classification performance. For example, V-HL-E outperforms V-HL by 2.3% and 4.4% when 10% and 50% samples are used as training data on the NTU dataset.

The better performance of our proposed method can be attributed to the following two reasons. First, the hypergraph structure is able to explore complex relationship among 3D objects, which leads to the superior performance of all hypergraph-based methods over other methods. Second, our vertex-weighted hypergraph structure takes the vertex impact into consideration. In our work, we aim to reduce the influence of repeated training data which could be redundancy and illy dominate the classification process. In this way, even the minority of the training samples in the original dataset can have better impact on the classification decision. When some classes have too more samples than others, giving equal weights for all samples will focus more on the huge classes and decrease the classification performance. Compared with traditional hypergraph which ignores the vertex weights, the proposed method is able to measure a more optimal hypergraph Laplician and accordingly generate a better data correlation. We note that the proposed method is a general extension of traditional hypergraph learning methods, and it can be used in other tasks besides 3D object classification.

The limitation of the proposed method lies in the computational load from the relevance learning and hyperedge weight learning procedure when dealing with large datasets. It is noted that both effectiveness and efficiency are important in practice. In this work, we mainly focus on building a robust learning framework towards better effectiveness and put efficiency as our future work. Confronting the computational cost issue, there are two possible solutions. First, data downsampling can be conducted to reduce the data size before hypergraph learning. Second, besides conducting hypergraph learning directly, it is possible to investigate hierarchical hypergraph learning strategy which regards the data in a pyramid structure. Thus, each layer can have less data and the computational cost can be reduced accordingly. We also note that 3D object classification is just one application of the proposed vertex-weighted hypergraph learning method and it can be used in other tasks too.

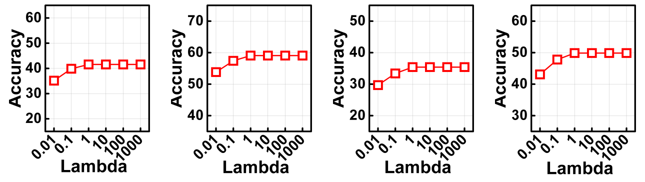
On Hyperedge Generation

In this subsection, we evaluate the influence of the number K of selected neighbors for hyperedge generation. We vary K from 3 to 50, and the experimental results are shown in Figure 5. As shown in the results, the performance is steady when K varies in a large range. When K is too small or too large, the performance becomes slightly worse. When K is too small, such as $K = 3$, each hyperedge connects too few vertices and the relationship among vertices cannot be



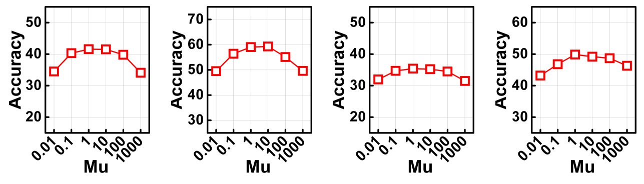
(a) (b) (c) (d)

Figure 5: The performance evaluation by varying K. (a) 10% training data on NTU; (b) 50% training data on NTU; (c) 10% training data on PSB; (d) 50% training data on PSB.



(a) (b) (c) (d)

Figure 6: The performance evaluation by varying λ. (a) 10% training data on NTU; (b) 50% training data on NTU; (c) 10% training data on PSB; (d) 50% training data on PSB.



(a) (b) (c) (d)

Figure 7: The performance evaluation by varying μ. (a) 10% training data on NTU; (b) 50% training data on NTU; (c) 10% training data on PSB; (d) 50% training data on PSB.

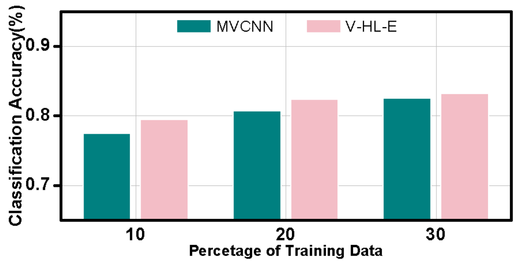


Figure 8: Comparison with MVCNN on the NTU dataset.

fully explored. When K is too large, such as $K = 50$, each hyperedge connects too many vertices and the discriminative ability of the hypergraph structure may be limited. Both small and large K values will degenerate the representation ability of the hypergraph structure.

On the Weights of Regularizers

In our framework, there are two parameters that control the relative impact of different regularizers in the objective function, *i.e.*, λ on empirical loss and μ on hyperedge weights, respectively.

To evaluate the influence of λ and μ on the object classification performance, we first keep μ as 1 and vary λ from 0.01 to 1000, and then keep λ as 10 and vary μ from 0.01 to 1000, respectively. Experimental results are demonstrated in Figure 6 to Figure 7. As shown in these results, the proposed method can achieve steady performance when λ and μ vary in a large range. When λ and μ are too small or too large, the corresponding regulariers will either dominate the objective function or have quite little influence on the results, which could degrade the performance.

4.3 Comparison with Deep Learning Method

The previous experiments have demonstrated that the proposed method with traditional features (Zernike Moments and HOG) can outperform the existing methods. We note that deep learning methods have been investigated in many computer vision tasks in recent years. Su et al. [Su *et al.*, 2015] introduced a 3D shape recognition method (MVCNN) using multi-view convolutional neural networks. Different from the deep learning methods, which serve as an end-to-end classification framework, the proposed method mainly works as a classification method, indicating that the output of the deep learning algorithm, such as MVCNN in [Su *et al.*, 2015] can be used as one feature for processing. We have conducted experiments on the NTU dataset with 10% to 30% training data to compare the proposed method with deep learning features with MVCNN [Su *et al.*, 2015]. Experimental results are demonstrated in Figure 8, from which we can notice that 1) MVCNN works much better than existing methods, and 2) the proposed method with deep learning features can achieve better performance compared with MVCNN [Su *et al.*, 2015]. The improvement compared with MVCNN is not very significant as MVCNN has already achieved very high accuracy.

5 Conclusion and Future Work

In this paper, we propose a vertex-weighted hypergraph learning approach for multi-view 3D object classification. Our method formulates the correlation among 3D objects in a hypergraph structure to explore the high-order relationship underneath the multi-view data. By introducing vertex weights, our method is able to explore the vertex importance in the hypergraph structure and the learning process on hypergraph can be more optimal for object classification, especially when the training data for different categories are not balanced.

To evaluate the proposed method, experiments are conducted on the NTU and the PSB datasets. The proposed method using traditional features, such as Zernike Moments and HOG, shows superior performance compared with both the state-of-the-art methods and traditional hypergraph methods. We have also tested the proposed method with the deep learning features. The experiments demonstrate that the proposed method with deep learning features can achieve better performance compared with the deep learning method.

Acknowledgments

This work was supported by National Natural Science Funds of China (61671267,61527812), National Science and Technology Major Project (No. 2016ZX01038101), MIIT IT funds (Research and application of TCN key technologies) of China, and The National Key Technology R&D Program (No. 2015BAG14B01-02).

References

[Bimbo and Pala., 2006] Alberto Del Bimbo and Pietro Pala. Content-based retrieval of 3D models. *ACM Transactions on Multimedia Computing, Communications, and Applications*, 2(1):20–43, 2006.

[Chen and Bhanu, 2009] Hui Chen and Bir Bhanu. Efficient recognition of highly similar 3d objects in range images. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 31(1):172–179, 2009.

[Chen *et al.*, 2003] Ding-Yun Chen, Xiao-Pei Tian, Yu-Te Shen, and Ming Ouhyoung. On visual similarity based 3D model retrieval. *Computer Graphics Forum*, 22(3):223–232, 2003.

[Gao *et al.*, 2012] Yue Gao, Meng Wang, Dacheng Tao, Rongrong Ji, and Qionghai Dai. 3D object retrieval and recognition with hypergraph analysis. *IEEE Transactions on Image Processing*, 21(9):4290–4303, 2012.

[Guo *et al.*, 2014] Yulan Guo, Mohammed Bennamoun, Ferdous Sohel, Min Lu, and Jianwei Wan. 3d object recognition in cluttered scenes with local surface features: a survey. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 36(11):2270–2287, 2014.

[Huang *et al.*, 2009] Yuchi Huang, Qingshan Liu, and Dimitris Metaxas. Video object segmentation by hypergraph cut. In *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, pages 1738–1745, 2009.

[Huang *et al.*, 2010] Yuchi Huang, Qingshan Liu, Shaoting Zhang, and Dimitris Metaxas. Image retrieval via probabilistic hypergraph ranking. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 3376–3383, 2010.

[Huang *et al.*, 2014] Zhiwu Huang, Ruiping Wang, Shiguang Shan, and Xilin Chen. Hybrid euclidean-and-riemannian metric learning for image set classification. In *Computer Vision–ACCV 2014*, pages 562–577. Springer, 2014.

[Huang *et al.*, 2015] Zhiwu Huang, Ruiping Wang, Shiguang Shan, Xianqiu Li, and Xilin Chen. Log-euclidean metric learning on symmetric positive definite manifold with application to image set classification. In *Proceedings of the 32nd International Conference on Machine Learning*, pages 720–729, 2015.

[Ji *et al.*, 2014] Rongrong Ji, Ling-Yu Duan, Jie Chen, Tiejun Huang, and Wen Gao. Mining compact 3d patterns for low bit rate mobile visual search. In *IEEE Transactions on Image Processing*, 2014.

[Liu *et al.*, 2015] Li Liu, Mengyang Yu, and Ling Shao. Multiview alignment hashing for efficient image search. *IEEE Transactions on image processing*, 24(3):956–966, 2015.

[Liu *et al.*, 2017] Tongliang Liu, Dacheng Tao, Mingli Song, and Stephen J Maybank. Algorithm-dependent generalization bounds for multi-task learning. *IEEE transactions on pattern analysis and machine intelligence*, 39(2):227–241, 2017.

[Shao *et al.*, 2014] Yuan-Hai Shao, Wei-Jie Chen, Jing-Jing Zhang, Zhen Wang, and Nai-Yang Deng. An efficient weighted lagrangian twin support vector machine for imbalanced data classification. *Pattern Recognition*, 47(9):3158–3167, 2014.

[Shao *et al.*, 2016] Ling Shao, Li Liu, and Mengyang Yu. Kernelized multiview projection for robust action recognition. *International Journal of Computer Vision*, 118(2):115–129, 2016.

[Shilane *et al.*, 2004] Philip Shilane, Patrick Min, Michael Kazhdan, and Thomas Funkhouser. The princeton shape benchmark. In *Proceedings of Shape Modeling International*, pages 1–12, 2004.

[Su *et al.*, 2015] Hang Su, Subhransu Maji, Evangelos Kalogerakis, and Erik Learned-Miller. Multi-view convolutional neural networks for 3d shape recognition. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 945–953, 2015.

[Tao *et al.*, 2006] Dacheng Tao, Xiaoou Tang, Xuelong Li, and Xindong Wu. Asymmetric bagging and random subspace for support vector machines-based relevance feedback in image retrieval. *IEEE transactions on pattern analysis and machine intelligence*, 28(7):1088–1099, 2006.

[Tao *et al.*, 2007] Dacheng Tao, Xuelong Li, Xindong Wu, and Stephen J Maybank. General tensor discriminant analysis and gabor features for gait recognition. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 29(10), 2007.

[Tao *et al.*, 2009] Dacheng Tao, Xuelong Li, Xindong Wu, and Stephen J Maybank. Geometric mean for subspace selection. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 31(2):260–274, 2009.

[Wang *et al.*, 2012a] Ruiping Wang, Huimin Guo, Larry S Davis, and Qionghai Dai. Covariance discriminative learning: A natural and efficient approach to image set classification. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 2496–2503. IEEE, 2012.

[Wang *et al.*, 2012b] Ruiping Wang, Shiguang Shan, Xilin Chen, Qionghai Dai, and Wen Gao. Manifold–manifold distance and its application to face recognition with image sets. *IEEE Transactions on Image Processing*, 21(10):4466–4479, 2012.

[Xu *et al.*, 2015] Chang Xu, Dacheng Tao, and Chao Xu. Multi-view intact space learning. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 37(12):2531–2544, 2015.

[Yu *et al.*, 2016] Mengyang Yu, Ling Shao, Xiantong Zhen, and Xiaofei He. Local feature discriminant projection. *IEEE transactions on pattern analysis and machine intelligence*, 38(9):1908–1914, 2016.

[Zhou *et al.*, 2007] Dengyong Zhou, Jiayuan Huang, and Bernhard Schokopf. Learning with hypergraphs: Clustering, classification, and embedding. In *Proceedings of Advances in Neural Information Processing Systems*, pages 1601–1608, 2007.