# ORTHOGONAL SUBSPACE DECOMPOSITION: A NEW PERSPECTIVE OF LEARNING DISCRIMINATIVE FEATURES FOR FACE CLUSTERING

**Anonymous authors**
Paper under double-blind review

## ABSTRACT

Face clustering is an important task, due to its wide applications in practice. Graph-based face clustering methods have recently made a great progress and achieved new state-of-the-art results. Learning discriminative node features is the key to further improve the performance of graph-based face clustering. To this end, we propose *subspace learning* as a new way to learn discriminative node features, which is implemented by a new orthogonal subspace decomposition (OSD) module. In graph-based face clustering, OSD leads to more discriminative node features, which better reflect the relationship between each pair of faces, thereby boosting the accuracy of face clustering. Extensive experiments show that OSD outperforms state-of-the-art results with a healthy margin.

## 1 INTRODUCTION

Face clustering has been explored for several years, because of its wide applications, such as online photo albums (Zhu et al., 2011), automatic data labeling or cleaning for dataset construction (Nech & Kemelmacher-Shlizerman, 2017; Guo et al., 2016; Zhan et al., 2018), and face image or video management Klontz & Jain (2013). Several traditional face clustering algorithms have been proposed and applied to the extracted hand-crafted or deep features of faces (Lloyd, 1982; Shi & Malik, 2000; Ester et al., 1996; Zhao et al., 2006; You et al., 2016). However, these algorithms usually rely on different data distribution assumptions, which limits their applicability. Therefore, some graph-based clustering methods (Wang et al., 2019; Yang et al., 2019; 2020; Zhang et al., 2020) are proposed in recent years, which make no assumptions on the data distribution. These graph-based methods achieve new state-of-the-art results on face clustering benchmarks.

To further improve the performance of graph-based clustering methods, learning discriminative node representations can be an effective way. Most previous methods propose to increase the inter-class distance or to decrease the intra-class distance for learning discriminative features, such as center loss (Wen et al., 2016) and AM-Softmax (Wang et al., 2018b;a). However, these methods are directly used on neural networks during training without considering to filter the noise or the redundant information of the learned features. But removing the noise or redundant information from features is a good strategy to obtain more discriminative representations (Lin et al., 2016; Chinta & Murty, 2012; Shang et al., 2019; Sun et al., 2019), and this process is called feature selection. Therefore, we propose a new method to learn discriminative features from the perspective of feature selection. Subspace learning is a classical method for feature selection, which can eliminate redundancy and noise effectively. However, previous subspace learning methods for feature selection (Wang et al., 2015a; Shang et al., 2020; Wang et al., 2015b; Shang et al., 2019) cannot be combined with neural networks and trained end-to-end, because these methods are unable to be optimized via gradient-based algorithms. Thus, we propose an orthogonal subspace decomposition (OSD) module that can be combined with neural networks for feature selection and learning discriminative representations.

In this work, we choose to leverage OSD to graph convolutional networks (GCNs) and the instance pivot subgraphs (IPSs) proposed by Wang et al. (2019). IPSs are built around each instance (pivot), to depict the local context, with each node modeling a pivot-neighbor pair. As for each pivot-neighbor pair, if the neighbor node and the pivot node have the same identity, then they should be linked. Therefore, the face clustering task can also be regarded as a prediction problem on face local

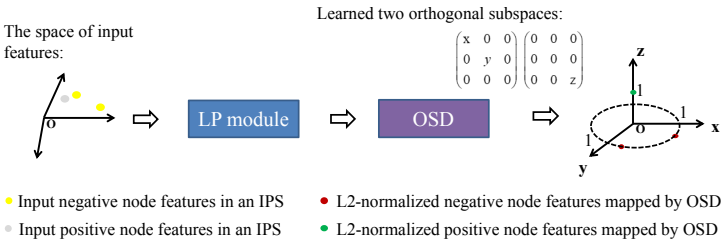


Figure 1: A simple example of using OSD for feature learning.

graphs, where faces indicating the same identity are linked together. In this paper, the "positive" nodes refer to the neighbor nodes that should be linked to the pivot, while the "negative" nodes refer to the neighbor nodes that should not be linked.

A simple example in Fig. 1 explains the main ideas behind OSD: the gray and yellow points denote input node features in an IPS, whose ground truth are positive and negative, respectively, and they lie in the input space. Link prediction (LP) is used to transform the input node features into $l2$-normalized node features and to classify them into two categories. The objective of OSD is to learn a new space that can be decomposed into two orthogonal subspaces and to make $l2$-normalized node features (i.e., the green and red points) lie in the two subspaces separately. In the "two orthogonal subspaces", any base vector in the basis of one subspace is orthogonal to any base vector in the basis of the other subspace, and at the same time, the base vectors in the basis of each subspace are orthogonal to each other. For example, we assume that a newly learned space is an Euclidean space $\mathbf{R}^3$, where each point can be denoted by $(x, y, z)$. Intuitively, the Euclidean space $\mathbf{R}^3$ can be decomposed into two orthogonal subspaces, i.e., $(x, y, 0)$ and $(0, 0, z)$. The $l2$-normalized positive and negative node features learned from OSD lie in these two orthogonal subspaces, respectively. By doing this, OSD implements the feature selection process. As OSD is required to zero out some dimensions of learned $l2$-normalized node features, it has to select the redundancy or noise to filter out, and as a result, only important and useful information can be preserved. As OSD performs feature selection, the new learned $l2$-normalized features will be easier to be classified or clustered closely, because the loss functions (e.g., cross-entropy loss with softmax) only perform on the dimensions that are important, resulting in more discriminative representations. Moreover, the orthogonality property in OSD also introduces a fixed margin to better differentiate the two types of nodes, especially for some hard samples. As illustrated in Fig. 1, the learned $l2$-normalized node features make the Euclidean distance of a pair of nodes fall into a range $[0, 2]$, and as for two nodes located on the two subspaces, the distance between them should be, in theory, around $\sqrt{2}$.

The main contributions of this paper are briefly summarized as follows:

- We propose orthogonal subspace decomposition (OSD), which to our knowledge is the first approach to combine subspace learning and feature selection with neural networks.
- We theoretically prove that, when combined with some regularization terms, matrix addition is a good approximation to the direct sum. As a result, we propose a new loss function, called *space reconstruction (SR) loss*, which enables OSD to be combined with graph convolutional networks (GCNs) and trained in an end-to-end manner.
- OSD not only outperforms some classical metric learning methods that are designed for learning discriminative features, namely center loss (Wen et al., 2016) and AM-Softmax (Wang et al., 2018b;a), but also surpasses state-of-the-art results on IJB-512, IJB-1024, IJB-1845 (Whitelam et al., 2017), and VoxCeleb2 (Chung et al., 2018).

## 2 Related Work

**Face Clustering.** During the past few years, different types of traditional clustering technologies were proposed. Some of them are not very suitable for face embeddings, for their rigid assumptions on the data distributions, such as K-means (Lloyd, 1982) and spectral clustering (Shi & Malik, 2000). Several new clustering methods try to overcome their shortcomings. Zhu et al. (2011) designed a new affinity metric, called rank-order distance, which is more robust to noise and outliers. Lin et al.

(2017) proposed the proximity-aware hierarchical clustering (PAHC) method that employs a linear SVM to classify local positive and negative instances. Some agglomerative hierarchical clustering (AHC) algorithms (e.g., (Zhu et al., 2011; Lin et al., 2017; 2018)) were also proposed to tackle clustering problem with complex data distributions. Zhan et al. (2018) trained a multi-layer perceptron (MLP) classifier to aggregate information and discover robust linkages, and then find connected components to acquire final clusters. Encoder-decoder architectures (e.g., (Dizaji et al., 2017; Xie et al., 2016; Yang et al., 2017)) were proposed to learn low-dimensional embeddings and cluster assignments. Yang et al. (2016) designed a recurrent paradigm to jointly update the clusters and feature representations. Shi et al. (2017) proposed a ConPac algorithm, which is optimized by using loopy belief propagation to maximize the posterior probability of the adjacency matrix. Recently, Yang et al. (2019) proposed a top-down method, based on a detection-segmentation paradigm, which is suitable for clusters with complicated structures. Wang et al. (2019) built the IPS and applied GCNs to the prediction of linkage among a pivot face embedding and its kNN neighbors. Yang et al. (2020) proposed a new clustering framework composed by two components. One component is to estimate the confidence of a vertex, while the other is to estimate the edge connectivity. Zhang et al. (2020) proposed a two-stage GCN framework, consisting of a global and a local GCN. The former classifies all nodes globally, while the latter performs on the low confidence nodes locally.

**Subspace Learning for Feature Selection.** There are many works related to subspace learning, and here we only focus on those designed for feature selection in recent years. Wang et al. (2015a) proposed a subspace learning method based on matrix factorization for unsupervised feature selection, which considers subspace learning as a matrix decomposition problem, and uses the learned feature selection matrix to select important features. Wang et al. (2015b) proposed an unsupervised method to obtain an indicator matrix for feature selection via minimizing the projection error and redundancy. Zhou et al. (2016) proposed a global and local structure preserving sparse subspace learning (GLoSS) algorithm, which preserves the geometric structure information in data. Cai et al. (2010) designed a multi-cluster feature selection (MCFS) method by incorporating sparse subspace learning with feature selection. MCFS introduces spectral analysis techniques and measures the correlations between different features for feature selection, so that the selected features can best preserve the cluster structure of the data. Gu et al. (2011) further improved the MCFS method by using the $L_{2,1}$-norm to enforce the row sparsity of the transformation matrix. Shang et al. (2019) proposed a local discriminative sparse subspace learning method for feature selection, which aims to exploit local discriminative and geometry information contained in the original features for feature selection, since such information is important for differentiating data samples. Shang et al. (2020) proposed a new algorithm called subspace learning for unsupervised feature selection via adaptive structure learning and rank approximation (SLASR), which learns the manifold structure adaptively to make the preserved local geometric structure more accurate and more robust to noise.

## 3 METHODOLOGY

**Preliminaries for OSD module.** The subspace decomposition in the OSD module is implemented by the direct sum (Kreyszig, 1978). Formally, a vector space Z is the *direct sum* of two subspaces X and Y of Z, denoted $Z = X \oplus Y$, if each $z \in Z$ has a unique representation $z = x + y$, where $x \in X$ and $y \in Y$. We call $Y$ an *algebraic complement* of $X$ in $Z$ and vice versa. The direct sum combines the basis of two subspaces to form a new whole space. In our method, we assume that the space of the $l2$-normalized node features learned by OSD is the whole space. Our goal is to learn two subspaces, and the direct sum of the two subspaces is the whole space. In this way, we can also regard the two subspaces as decomposed from the whole space. However, the direct sum cannot be implemented and trained with neural networks in an end-to-end manner, which causes difficulties for learning the two subspaces. Therefore, we provide the following theorem (with proof in Appendix A), which reveals a relationship between the direct sum and subspace matrix addition. (In this paper, we organize the base vectors of a subspace as a matrix *by row* for simplicity).

**Theorem 1.** *Given two subspaces $V_1$ and $V_2$ of $V$. If $V = V_1 + V_2$, then $V_1 \cap V_2 = \{\vec{0}\}$ is equivalent to V being the direct sum of $V_1$ and $V_2$.*

The above result provides the fundament for our subspace decomposition, as we can simply perform the addition operation on two subspace matrices to approximate the direct sum of them by introducing regularization terms to fulfill the condition in this theorem, i.e., making the intersection of

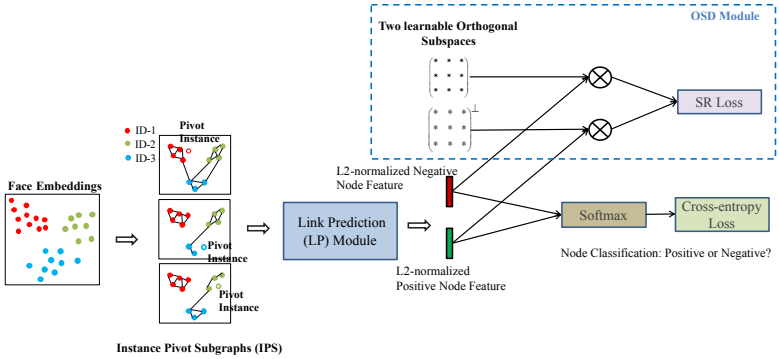


Figure 2: How to use the OSD module (shown in the dotted box). Face embeddings are extracted by using the ArcFace model (Deng et al., 2019). OSD only contains two learnable matrices, which are regarded as the two subspaces for the two types of $l2$-normalized node features.

the two learned subspaces only contain $\vec{0}$. In fact, how to design regularization terms to fulfill this condition is an open question. Here, we make the basis of one subspace orthogonal to the basis of the other subspace, as the inner product of any two vectors from the two subspaces is zero. This orthogonality can easily be implemented by adding a regularization term that enforces the product of the two subspace matrices to be the zero matrix. With this regularization term and matrix addition, we can learn the basis of the two decomposed subspaces and obtain the basis of the whole space.

As the learned $l2$-normalized node features are represented in the whole space, each of them can be uniquely decomposed into two feature vectors that lie in the two subspaces, which is immediate by the uniqueness representation property of the direct sum: by the definition of the direct sum, if V is the direct sum of $V_1$ and $V_2$, then any $v \in V$ can be uniquely decomposed into $v_1 \in V_1$ and $v_2 \in V_2$ through $v = v_1 + v_2$. Thus, the decomposition of features into each subspace is unique, and the projected features disjoint from each other. As for a $l2$-normalized node feature, there are many choices in respect of decomposing it into two feature vectors that lie in these two subspaces. Considering that OSD aims to implement feature selection, which is a process that zeros out some dimensions, we can design some regularization terms that make the base vectors in the basis of each subspace linearly independent of each other, and also make the representation of positive $l2$-normalized node features in the subspace of negative ones to be $\vec{0}$ and vice versa. In this way, some dimensions of $l2$-normalized positive and negative node features are compulsorily filtered out by OSD, and thus, it should select the important information to preserve and zero out useless or redundant information. For example, the representation of a $l2$-normalized node feature in a subspace can be considered as a multiplication of the feature and the subspace matrix, which is a linear combination of the basis of the subspace based on the feature vector. Due to the linearly independent property of the subspaces, the basis of each subspace only contains the zero base vector and a set of non-zero base vectors. If the representation of the $l2$-normalized positive node features in the subspace that negative node features lie in is $\vec{0}$, the dimensions of the $l2$-normalized positive node features that are corresponding to the non-zero base vectors should be equal to zero.[1] In addition, as discussed above, when the two types of $l2$-normalized node features only lie in their own type of subspace, orthogonality introduces a fixed margin between these two types of $l2$-normalized node features, resulting in a more discriminative representation of them.

**LP Module.** The LP module, denoted LP(GCN-M), is a GCN constructed by stacking four graph convolutional layers (GCLs), following the GCN-M by Wang et al. (2019) by using the GCL with the mean aggregator. In detail, the first GCL performs on the input node features $X \in R^{N \times d_{in}}$, while the other GCLs receive the transformed features from the previous layer. The output node features from the last graph convolution layer are $l2$-normalized and form a matrix $\mathcal{Y} \in R^{N \times d_{out}}$, which is fed to the classification (softmax) layer for link prediction. Here, $N$ is the number of nodes, while $d_{in}$ and $d_{out}$ are the dimensions of input and output node features, respectively.

[1]For instance, considering a subspace matrix $V \in R^{5 \times 5}$, then the first two rows are linearly independent base vectors $v_1$ and $v_2$, while the other three rows are all $\vec{0}$. For a given feature vector $z \in R^{1 \times 5}$, if $zV = \vec{0}$, then the first two dimensions of $z$ should be zero, because $v_1$ and $v_2$ are linearly independent.

**Algorithm 1:** Procedure of training the whole framework

**Input:**
Training data {X, A}, Training data size: M, Batch size: m, Training epochs: T, LP(GCN-M) with param $\theta$: $LP_\theta$, OSD module with two subspaces: $S_1$ and $S_2$.
**for** i **in** range(0, T):
  **for** j **in** range(0, $\frac{M}{m}$):
    Sample j-th batch of data $\{x^{(i)}, A^{(i)}\}_{i=1}^{m}$.
    Feed-forward pass: $\mathcal{Y}$ = $LP_\theta(\{x^{(i)}, A^{(i)}\})$.
    Compute $L_{\text{entropy}}$ and $L_{\text{SR}}$ with $\mathcal{Y}$ and $S_1$, $S_2$ .
    Compute gradients $\nabla_{\text{entropy}}$ and $\nabla_{\text{SR}}$.
    Update: $\nabla_{\text{SR}} \rightarrow S_1, S_2$, and $LP_\theta$;  $\nabla_{\text{entropy}} \rightarrow LP_\theta$.
  **end for**
**end for**

**OSD Module.** The OSD module (see Fig. 2) aims to learn the two decomposed orthogonal subspaces, which are implemented by two learnable matrices, denoted $S_1, S_2 \in R^{d_{out} \times d_{out}}$. Hence, OSD merely introduces extremely tiny additional computational overhead to the whole framework during training. We consider each row of the subspaces $S_1$ and $S_2$ as a base vector of their basis.

By the definition of the direct sum, the original whole space $S_{\text{ori}}$ where the learned $l2$-normalized node features lie in can be regarded as the direct sum of two subspaces $S_1$ and $S_2$:

$$S_{\text{ori}} = S_1 \oplus S_2. \tag{1}$$

To learn the decomposed subspaces, the problem lies in optimizing:

$$L_{\text{recon}} = \mathbb{E}[(\mathcal{Y}S_1 \oplus \mathcal{Y}S_2 - \mathcal{Y})^{\circ 2}], \tag{2}$$

where $\mathbb{E}[\cdot]$ takes the expectation or overall mean value of an input matrix or batch of input matrices. $(\cdot)^{\circ 2}$ takes the Hadamard square of the input matrix. Here, $\mathcal{Y} \in R^{N \times d_{out}}$ is the representation in the original whole space, and $\mathcal{Y}S_1 \in R^{N \times d_{out}}$ and $\mathcal{Y}S_2 \in R^{N \times d_{out}}$ are the representation of $\mathcal{Y}$ in the subspaces $S_1$ and $S_2$, respectively.[2]

To implement the direct sum, we try to approximate the subspace matrix addition to the direct sum, as the addition operation enables the gradient to be backpropagated from the top to the bottom. Therefore, based on Theorem 1, we propose the following regularization term and combine it with the addition operation to implement the direct sum:

$$L_{orth} = \mathbb{E}[(S_1 S_2^T)^{\circ 2}] + \mathbb{E}[(S_1 S_1^T)^{\circ 2}] + \mathbb{E}[(S_2 S_2^T)^{\circ 2}]. \tag{3}$$

Minimizing the first term of Eq. (3) encourages the two subspaces to be orthogonal to each other, which ensures that the intersection of the two subspaces can only contain the zero vector, satisfying Theorem 1. Minimizing the second term and the third term of Eq. (3) penalizes both diagonal and off-diagonal elements of $S_1 S_1^T$ and $S_2 S_2^T$. Penalizing off-diagonal elements ensures that the base vectors of each subspace are linearly independent of each other. Penalizing the diagonal elements encourages OSD to learn low-rank subspace matrices, and by doing this, there will not be too many non-zero base vectors in the two subspaces, avoiding filtering out too much information from $\mathcal{Y}$.

Moreover, as what we have discussed above, we make the representation of learned node features on the other subspace be $\vec{0}$ by introducing the following regularization term:

$$L_{\text{norm}} = ||\mathcal{Y}_{(+)} S_2||_2 + ||\mathcal{Y}_{(-)} S_1||_2. \tag{4}$$

Here, without loss of generality, we use $\mathcal{Y}_{(+)}$ and $\mathcal{Y}_{(-)}$ to denote a positive $l2$-normalized node feature and a negative $l2$-normalized node feature respectively. In addition, we also assume that $S_1$ and $S_2$ are the subspaces of positive and negative node features, respectively. Minimizing Eq. (4) leads the network to learning features that are only distributed in their own subspace, thereby implementing the feature selection. Finally, according to the simple example in Sec. 1, the trivial solution $I$ (i.e., the identity matrix) is a good solution for the original space $S_{\text{ori}}$, but not the best. $I$ restricts the search space of the two decomposed subspaces, as the addition of non-diagonal elements of them

[2]In general, given a vector $z$, the representation of $z$ in a subspace $V$ is $zV^TV$, where $zV^T$ denotes the projection. However, in this paper, we assume that the image of $S_1^T - I$ and $S_2^T - I$ are contained in the $Ker(\mathcal{Y})$, where $I$ denotes identity matrix. Therefore, the projection of the two types of nodes on the subspaces are still $\mathcal{Y}$, i.e., $\mathcal{Y}S_1^T = \mathcal{Y}$ and $\mathcal{Y}S_2^T = \mathcal{Y}$. The representation of the two types of node features on the two subspaces can be written as $\mathcal{Y}S_1$ and $\mathcal{Y}S_2$ respectively.

| $u$ \ $k_2$ | 2 | 3 | 4 | 5 | 6 | 7 | 8 |
|---|---|---|---|---|---|---|---|
| 3 | 92.0 | 94.2 | - | - | - | - | - |
| 4 | 92.0 | 94.1 | 94.3 | - | - | - | - |
| 5 | 90.4 | 93.3 | 93.9 | 94.4 | - | - | - |
| 6 | 87.9 | 92.2 | 93.4 | 93.9 | 94.0 | - | - |
| 7 | 87.1 | 91.6 | 93.1 | 93.7 | 84.1 | 93.7 | - |

Table 1: Ablation study on the effects of different $u$ and $k_2$ with $k_1 = 40$ in terms of the BCubed F-measure (%). "-": the result is lower than 60%.

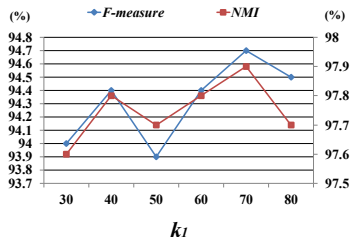


Figure 3: Ablation study on the effects of different $k_1$ with $u = 5$ and $k_2 = 5$ in terms of B-Cubed F-measure (%, left) and NMI (%, right).

| Methods | F-measure (%) | NMI |
|---|---|---|
| LP(GCN-M) | 90.3 | 0.968 |
| LP(GCN-M) with Center Loss | 91.1 | 0.970 |
| LP(GCN-M) with AM-Softmax (m=0.35) | 92.7 | 0.974 |
| LP(GCN-M) with AM-Softmax (m=0.50) | 92.7 | 0.975 |
| LP(GCN-M) with AM-Softmax (m=0.80) | 92.9 | 0.975 |
| OSD-LP(GCN-M) w/ $L_{\text{orth}}$ | 90.3 | 0.968 |
| OSD-LP(GCN-M) w/ $L_{\text{diag}}$ | 90.4 | 0.968 |
| OSD-LP(GCN-M) w/ $L_{\text{recon}}$ | 90.7 | 0.969 |
| OSD-LP(GCN-M) w/ $L_{\text{norm}}$ | 90.9 | 0.969 |
| OSD-LP(GCN-M) w/o ($L_{\text{recon}} + L_{\text{orth}}$) | 91.3 | 0.969 |
| OSD-LP(GCN-M) w/o ($L_{\text{recon}} + L_{\text{norm}}$) | 90.5 | 0.967 |
| OSD-LP(GCN-M) w/o ($L_{\text{recon}} + L_{\text{diag}}$) | 93.5 | 0.973 |
| OSD-LP(GCN-M) w/o ($L_{\text{orth}} + L_{\text{norm}}$) | 91.0 | 0.969 |
| OSD-LP(GCN-M) w/o ($L_{\text{orth}} + L_{\text{diag}}$) | 92.5 | 0.971 |
| OSD-LP(GCN-M) w/o ($L_{\text{norm}} + L_{\text{diag}}$) | 93.0 | 0.972 |
| OSD-LP(GCN-M) w/o $L_{\text{orth}}$ | 93.4 | 0.973 |
| OSD-LP(GCN-M) w/o $L_{\text{diag}}$ | 94.4 | 0.978 |
| OSD-LP(GCN-M) w/o $L_{\text{recon}}$ | 94.1 | 0.976 |
| OSD-LP(GCN-M) w/o $L_{\text{norm}}$ | 93.7 | 0.975 |
| OSD-LP(GCN-M) | **94.7** | **0.979** |

Table 2: Top: Combination of LP(GCN-M) and the two types of classical metric learning methods, i.e., center loss (Wen et al., 2016) and AM-softmax (Wang et al., 2018b;a). Bottom: Ablation study of OSD-LP(GCN-M) and SR loss.

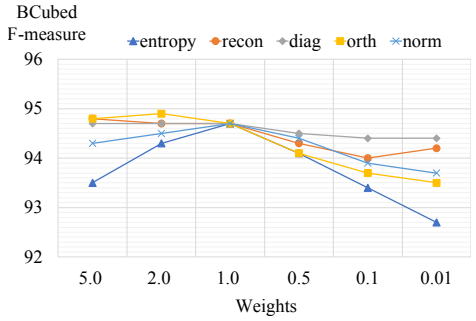


Figure 4: Ablation study on the weights of different loss terms.

should be zero. Therefore, we propose to penalize the diagonal elements to avoid the trivial solution by designing:

$$L_{\text{diag}} = ||Diag(S_{\text{ori}})||_2, \tag{5}$$

where $Diag(\cdot)$ takes the diagonal elements of the input matrix. Hence, in the real implementation, we can directly add the two subspaces to approximate the direct sum of them by combining the above regularization terms together, resulting in a space reconstruction (SR) loss:

$$L_{\text{SR}} = L_{\text{recon}} + L_{\text{orth}} + L_{\text{norm}} + L_{\text{diag}}. \tag{6}$$

Finally, the total loss of the whole framework is:

$$L_{\text{total}} = L_{\text{entropy}} + L_{\text{SR}}. \tag{7}$$

Here, $L_{\text{entropy}}$ refers to the cross-entropy loss with softmax for classifying the positive and negative $l2$-normalized node features.

In the testing phase, the two subspace matrices are no longer needed, and only LP(GCN-M) is used, so that no additional computation burden is put on the inference. We follow Wang et al. (2019) to construct IPS by looping over all the instances. LP(GCN-M) predicts the likelihood of a linkage between the involved instances and their related pivot nodes. After the loop, we accordingly adopt the pseudo-label propagation strategy by Zhan et al. (2018) to merge the links and form clusters.

**Optimization.** The OSD module and LP(GCN-M) are jointly trained, and the whole optimization process is summarized in **Algorithm 1**. From now on, we use "OSD-LP(GCN-M)" to denote the whole framework that consists of LP(GCN-M) and the OSD module.

| Method | IJB-B-512 | | IJB-B-1024 | | IJB-B-1845 | |
|---|---|---|---|---|---|---|
| | F | NMI | F | NMI | F | NMI |
| K-Means (Lloyd (1982)) | 61.2 | 0.858 | 60.3 | 0.865 | 60.0 | 0.868 |
| Spectral (Shi & Malik (2000)) | 51.7 | 0.784 | 50.8 | 0.792 | 51.6 | 0.785 |
| AHC (Jain & Dubes (1988)) | 79.5 | 0.917 | 79.7 | 0.925 | 79.3 | 0.923 |
| AP (Frey & Dueck (2007)) | 49.4 | 0.854 | 48.4 | 0.864 | 47.7 | 0.869 |
| DBSCAN (Ester et al. (1996)) | 75.3 | 0.841 | 72.5 | 0.833 | 69.5 | 0.814 |
| ARO (Otto et al. (2018)) | 76.3 | 0.898 | 75.8 | 0.908 | 75.5 | 0.913 |
| $PAHC_{ori}$ (Lin et al. (2018)) | - | - | 63.9 | 0.890 | 61.0 | 0.890 |
| $ConPac_{ori}$ (Shi et al. (2017)) | 65.6 | - | 64.1 | - | 63.4 | - |
| DDC (Lin et al. (2017)) | 80.2 | 0.921 | 80.5 | 0.926 | 80.0 | 0.929 |
| GCN-D&S (Yang et al. (2019)) | 81.3 | 0.924 | 81.7 | 0.928 | 80.9 | 0.931 |
| GCN-V&E (Yang et al. (2020)) | 83.2 | 0.930 | 82.8 | 0.938 | 82.4 | 0.938 |
| Global-local GCN (Zhang et al. (2020)) | 82.7 | 0.924 | 82.5 | 0.930 | 81.1 | 0.926 |
| GCN-M (Wang et al. (2019)) | 82.1 | 0.920 | 81.9 | 0.928 | 81.0 | 0.927 |
| OSD-LP(GCN-M) | **84.2** | **0.937** | **83.6** | **0.942** | **83.1** | **0.943** |

Table 3: Comparison with previous methods in terms of the BCubed F-measure (%) and the NMI score. Most results employ the ArcFace (Deng et al., 2019) model to extract face embeddings, except for those that have the subscript '$ori$'.

| Method | IJB-B-512 | | IJB-B-1024 | | IJB-B-1845 | |
|---|---|---|---|---|---|---|
| | F | NMI | F | NMI | F | NMI |
| AHC (Jain & Dubes (1988)) | 68.8 | 0.874 | 69.4 | 0.880 | 67.6 | 0.867 |
| ARO (Otto et al. (2018)) | 62.4 | 0.852 | 62.8 | 0.853 | 61.9 | 0.848 |
| DDC (Lin et al. (2017)) | 70.4 | 0.887 | 70.8 | 0.892 | 69.7 | 0.889 |
| GCN-M (Wang et al. (2019)) | 73.6 | 0.910 | 73.3 | 0.913 | 72.6 | 0.908 |
| OSD-LP(GCN-M) | **75.3** | **0.919** | **74.7** | **0.922** | **74.9** | **0.915** |

Table 4: Comparison of methods based on face embedding from ResNet-50 and softmax.

## 4 EXPERIMENTS

**Datasets.** We conducted extensive experiments on the following datasets. As for face clustering, we used CASIA-Webface (Yi et al., 2014) as our training set, which consists of 10,575 identities with around 494,414 face images. FaceScrub (Ng & Winkler, 2014) is a celebrity face dataset that comprises a total of 106,863 face images. We randomly picked 517 identities, and we removed duplicates to construct a subset with 33607 face images. We used this subset as our validation set. IJB-B (Whitelam et al., 2017), which has a clustering protocol, was applied to our testing set. The protocol has 7 subtasks, and we chose the three largest subtasks. In particular, those three subtasks include 18,171, 36,575, and 68,195 face images from 512, 1,024, and 1,845 identities respectively, denoted IJB-B-512, IJB-B-1024, and IJB-B-1845. As for multi-view testing, we used VoxCeleb2 (Chung et al., 2018), which has 145K videos of 5,994 different identities. We followed the experimental setting of multi-view testing in (Wang et al., 2019) for a fair comparison, i.e., constructing two testing sets by picking 512 and 2048 identities. The other data were used for the training set. The implementation details are given in Appendix A.

**Ablation Studies.** We now report on our ablation studies on FaceScrub (Ng & Winkler, 2014).

Firstly, we explored how the hyperparameters of IPS impact the performance of OSD-LP(GCN-M), which include the nearest neighbors of the pivot ($k_1$), the nearest neighbors of each 1-hop node ($k_2$), and the linked nearest neighbors for picking edges ($u$). The specific meaning and function of these hyperparameters can be found in (Wang et al., 2019). It is not necessary to use the same configuration of constructing IPS as in the training phase. So, we first set $k_1 = 40$ and only studied different $u$ and $k_2$, and our results are reported in Table 1. We conclude that, for any fixed $u$, increasing $k_2$ always leads to a performance gain until $k_2$ equals to $u$. Once $k_2$ is larger than $u$, a significant drop can be observed in terms of the F-measure. In contrast, given a fixed $k_2$, larger $u$ results in a lower BCubed F-measure in most cases. The highest BCubed F-measure is obtained when $k_2 = u = 5$. Then, using this setting, we tested the effect brought by different $k_1$. According to the results in Fig. 3, we found that the F-measure and NMI reach the peak when $k_1 = 70$. Therefore, from now on, we *fix* $k_1 = 70$, $k_2 = 5$, and $u = 5$ in all of the following experiments.

Secondly, we compare the OSD module with two other classical metric learning methods, which are widely used for learning discriminative features, as well as the SR loss. In detail, in the first part of Table 2, we report the result of LP(GCN-M) combined with center loss (Wen et al., 2016) and AM-softmax (Wang et al., 2018b;a). Although center loss and AM-softmax can improve the performance

| Method | Face | | Audio | | Face + Audio | |
|---|---|---|---|---|---|---|
| | F | NMI | F | NMI | F | NMI |
| K-Means (Lloyd (1982)) | 64.8 | 0.877 | 22.9 | 0.644 | 63.6 | 0.874 |
| Spectral (Shi & Malik (2000)) | 59.2 | 0.825 | 21.4 | 0.619 | 54.1 | 0.782 |
| AHC (Jain & Dubes (1988)) | 75.5 | 0.913 | 35.8 | 0.704 | 83.3 | 0.934 |
| ARO (Otto et al. (2018)) | 57.5 | 0.875 | 26.1 | **0.834** | 31.9 | 0.835 |
| GCN-M (Wang et al. (2019)) | 80.1 | 0.921 | 39.5 | 0.497 | 84.1 | 0.940 |
| OSD-LP(GCN-M) | **81.9** | **0.933** | **41.4** | 0.507 | **86.0** | **0.948** |

Table 5: Multi-view testing on VoxCeleb2 with 512 identities in terms of the BCubed F-measure (%) and NMI.

| Method | Face | | Audio | | Face + Audio | |
|---|---|---|---|---|---|---|
| | F | NMI | F | NMI | F | NMI |
| K-Means (Lloyd (1982)) | 58.9 | 0.871 | 15.2 | 0.650 | 58.2 | 0.871 |
| AHC (Jain & Dubes (1988)) | 69.5 | 0.908 | 22.8 | 0.686 | 78.5 | 0.938 |
| ARO (Otto et al. (2018)) | 58.3 | 0.858 | 27.7 | **0.813** | 37.0 | 0.873 |
| GCN-M (Wang et al. (2019)) | 76.6 | 0.932 | 31.1 | 0.452 | 81.0 | 0.946 |
| OSD-LP(GCN-M) | **77.8** | **0.941** | **33.6** | 0.469 | **82.9** | **0.961** |

Table 6: Multi-view testing on VoxCeleb2 with 2048 identities in terms of the BCubed F-measure (%) and NMI.

of the baseline, i.e., LP(GCN-M), the improvement on LP(GCN-M) brought by OSD is higher than that brought by either of the two methods (last line of Table 2), indicating OSD's superiority in helping with learning discriminative representations. The second part of Table 2 shows the results of different combinations of subterms in the SR loss to explore the effect of the SR loss on the OSD module: once we remove any subterm of the SR loss, a decline can be observed with regard to F-measure and NMI, confirming the correctness of the theory and the importance of the SR loss.

Finally, we explored the weights of the loss terms in $L_{\text{total}}$, and the results are shown in Fig. 4. Note that each curve in Fig. 4 reflects the weight of a loss term, and when the weight changes, the weight of the other loss terms are fixed to 1. We can conclude that increasing or decreasing the weight of $L_{\text{entropy}}$ and $L_{\text{norm}}$ leads to a performance degradation. As for the other loss terms, declining their weights from 1.0 to 0.001 will hurt the performance of the model. By contrast, further increasing the weights does not affect the performance too much. Concerning that, it is difficult to do a comprehensive grid search of those hyperparameters in order to find the best ones, thus we chose equal weights for those loss terms during training.

**Comparison with State-of-the-Art Methods.** We now compare our method to previous ones. The first part of Table 3 shows some traditional clustering methods on the IJB-B dataset. Due to their restricted assumptions on features (except for AHC), most of those methods always perform worse than ours with a significant margin. As for AHC, it performs best among traditional methods, reaching approximately 79.5% F-measure and 0.920 NMI on the test set. However, OSD-LP(GCN-M) still outperforms AHC with a large margin.

In the second part of Table 3, we list the state-of-the-art methods proposed in recent years. It is obvious that our method surpasses previous state-of-the-art methods on the three subtasks with regard to both F-measure and NMI score. Note that OSD is additively used on the LP(GCN-M), and therefore the fact that OSD-LP(GCN-M) outperforms GCN-M in (Wang et al., 2019) directly demonstrates the availability of the proposed method.

Besides, we followed Wang et al. (2019) and tested the OSD module on the face embeddings from ResNet-50 (He et al., 2016) model trained with softmax and cross-entropy loss. The results are shown in Table 4, and we can conclude that OSD-LP(GCN-M) can still outperform previous methods based on different face embeddings, validating the versatility of the OSD module.

**Multi-View Testing.** To further verify the generalization of our OSD module, we followed Wang et al. (2019) and also evaluated our model on multi-view data. We utilized face embeddings and audio embeddings to evaluate OSD on video face data clustering. According to the results in Tables 5 and 6, although the data distribution of face embeddings is quite different from that of audio embeddings, OSD can improve the performance on both of them, which means that OSD is a general module, and it works well when used for different data modalities. Moreover, OSD also achieves new state-of-the-art results in the combination of multi-view data.

## 5 Summary and Outlook

In this work, we have proposed orthogonal subspace decomposition (OSD), which implements a new way to help the network to learn discriminative features via subspace learning and feature selection. We have also reported on extensive experiments, which verify the effectiveness of the proposed method. Moreover, to our knowledge, this is the first work to simultaneously combine subspace learning and feature selection with neural networks. Finally, this work provides a new research direction, i.e., how to leverage feature selection and subspace learning to encourage neural networks to learn more discriminative representations.

## REFERENCES


E. Amigó, J. Gonzalo, J. Artiles, and F. Verdejo. A comparison of extrinsic clustering evaluation metrics based on formal constraints. *Information Retrieval*, 12(4):461–486, 2009.

D. Cai, C. Zhang, and X. He. Unsupervised feature selection for multi-cluster data. In *Proceedings of the 16th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pp. 333–342, 2010.

Q. Cao, L. Shen, W. Xie, O. M. Parkhi, and A. Zisserman. VGGFace2: A dataset for recognising faces across pose and age. In *International Conference on Automatic Face & Gesture Recognition*, 2018.

P. M. Chinta and M. N. Murty. Discriminative feature analysis and selection for document classification. In *International Conference on Neural Information Processing*, pp. 366–374. Springer, 2012.

J. S. Chung, A. Nagrani, and A. Zisserman. VoxCeleb2: Deep speaker recognition. *Interspeech*, 2018.

J. Deng, J. Guo, N. Xue, and S. Zafeiriou. ArcFace: Additive angular margin loss for deep face recognition. In *Computer Vision and Pattern Recognition*, 2019.

D. G. Dizaji, A. Herandi, C. Deng, W. Cai, and H. Huang. Deep clustering via joint convolutional autoencoder embedding and relative entropy minimization. In *International Conference on Computer Vision*, 2017.

M. Ester, H. Kriegel, J. Sander, and X. Xu. A density-based algorithm for discovering clusters in large spatial databases with noise. In *Knowledge Discovery and Data Mining*, 1996.

B. J. Frey and D. Dueck. Clustering by passing messages between data points. *Science*, 315(5814): 972–976, 2007.

Q. Gu, Z. Li, and J. Han. Joint feature selection and subspace learning. In *International Joint Conference on Artificial Intelligence*, volume 22, pp. 1294. Citeseer, 2011.

Y. Guo, L. Zhang, Y. Hu, X. He, and J. Gao. MS-Celeb-1M: A dataset and benchmark for large-scale face recognition. In *European Conference on Computer Vision*, 2016.

K. He, X. Zhang, S. Ren, and J. Sun. Deep residual learning for image recognition. In *CVPR*, pp. 770–778, 2016.

A. K. Jain and R. C. Dubes. Algorithms for clustering data. *Technometrics*, 32(2):227–229, 1988.

J. C. Klontz and A. K. Jain. A case study of automated face recognition: The Boston marathon bombings suspects. *Computer*, 46(11):91–94, 2013.

E. Kreyszig. *Introductory functional analysis with applications*, volume 1. wiley New York, 1978.

W. Lin, J. Chen, and R. Chellappa. A proximity-aware hierarchical clustering of faces. In *International Conference on Automatic Face & Gesture Recognition*. IEEE, 2017.

W. Lin, J. Chen, C. D. Castillo, and R. Chellappa. Deep density clustering of unconstrained faces. In *Computer Vision and Pattern Recognition*, 2018.

Y. Lin, Q. Hu, J. Zhang, and X. Wu. Multi-label feature selection with streaming labels. *Information Sciences*, 372:256–275, 2016.

S. P. Lloyd. Least squares quantization in PCM. *IEEE Transactions on Information Theory*, 28(2): 129–137, 1982.

L. V. D. Maaten and G. Hinton. Visualizing data using t-SNE. *Journal of Machine Learning Research*, 9(Nov):2579–2605, 2008.

A. Nech and I. Kemelmacher-Shlizerman. Level playing field for million scale face recognition. In *Computer Vision and Pattern Recognition*, 2017.

H. Ng and S. Winkler. A data-driven approach to cleaning large face datasets. In *International Conference on Image Processing*, 2014.

C. Otto, D. Wang, and A. K. Jain. Clustering millions of faces by identity. *IEEE Transactions on Pattern Analysis & Machine Intelligence*, 40(2):1–1, 2018.

R. Shang, Y. Meng, W. Wang, F. Shang, and L. Jiao. Local discriminative based sparse subspace learning for feature selection. *Pattern Recognition*, 92:219–230, 2019.

R. Shang, K. Xu, and L. Jiao. Subspace learning for unsupervised feature selection via adaptive structure learning and rank approximation. *Neurocomputing*, 413:72–84, 2020.

J. Shi and J. Malik. Normalized cuts and image segmentation. *Departmental Papers (CIS)*, pp. 107, 2000.

Y. Shi, C. Otto, and A. K. Jain. Face clustering: representation and pairwise constraints. *IEEE Transactions on Information Forensics & Security*, PP(99):1–1, 2017.

Z. Sun, J. Zhang, L. Dai, C. Li, C. Zhou, J. Xin, and S. Li. Mutual information based multi-label feature selection via constrained convex optimization. *Neurocomputing*, 329:447–456, 2019.

F. Wang, J. Cheng, W. Liu, and H. Liu. Additive margin softmax for face verification. *IEEE Signal Processing Letters*, 25(7):926–930, 2018a.

H. Wang, Y. Wang, Z. Zhou, X. Ji, D. Gong, J. Zhou, Z. Li, and W. Liu. CosFace: Large margin cosine loss for deep face recognition. In *Computer Vision and Pattern Recognition*, 2018b.

S. Wang, W. Pedrycz, Q. Zhu, and W. Zhu. Subspace learning for unsupervised feature selection via matrix factorization. *Pattern Recognition*, 48(1):10–19, 2015a.

S. Wang, W. Pedrycz, Q. Zhu, and W. Zhu. Unsupervised feature selection via maximum projection and minimum redundancy. *Knowledge-Based Systems*, 75:19–29, 2015b.

Z. Wang, L. Zheng, Y. Li, and S. Wang. Linkage-based face clustering via graph convolution network. In *Computer Vision and Pattern Recognition*, 2019.

Y. Wen, K. Zhang, Z. Li, and Y. Qiao. A discriminative feature learning approach for deep face recognition. In *European conference on computer vision*. Springer, 2016.

C. Whitelam, E. Taborsky, A. Blanton, B. Maze, J. Adams, T. Miller, N. Kalka, A. K. Jain, J. A. Duncan, and K. Allen. IARPA Janus Benchmark-B Face dataset. In *Computer Vision and Pattern Recognition, Workshops*, 2017.

J. Xie, R. Girshick, and A. Farhadi. Unsupervised deep embedding for clustering analysis. In *International Conference on Machine Learning*, 2016.

B. Yang, X. Fu, N. D. Sidiropoulos, and M. Hong. Towards k-means-friendly spaces: Simultaneous deep learning and clustering. In *International Conference on Machine Learning*, 2017.

J. Yang, D. Parikh, and D. Batra. Joint unsupervised learning of deep representations and image clusters. In *Computer Vision and Pattern Recognition*, 2016.

L. Yang, X. Zhan, D. Chen, J. Yan, C. C. Loy, and D. Lin. Learning to cluster faces on an affinity graph. In *Computer Vision and Pattern Recognition*, 2019.

Y. Yang, D. Chen, X. Zhan, R. Zhao, C. C. Loy, and D. Lin. Learning to cluster faces via confidence and connectivity estimation. In *Computer Vision and Pattern Recognition*, 2020.

D. Yi, Z. Lei, S. Liao, and S. Z. Li. Learning face representation from scratch. *arXiv:1411.7923*, 2014.

C. You, D. Robinson, and R. Vidal. Scalable sparse subspace clustering by orthogonal matching pursuit. In *Computer Vision and Pattern Recognition*, 2016.

X. Zhan, Z. Liu, J. Yan, D. Lin, and C. C. Loy. Consensus-driven propagation in massive unlabeled data for face recognition. In *European Conference on Computer Vision*, 2018.

Y. Zhang, W. Deng, M. Wang, J. Hu, X. Li, D. Zhao, and D. Wen. Global-local GCN: Large-scale label noise cleaning for face recognition. In *Computer Vision and Pattern Recognition*, 2020.

M. Zhao, Y. W. Teo, S. Liu, T. Chua, and R. Jain. Automatic person annotation of family photo album. In *International Conference on Image and Video Retrieval*, 2006.

N. Zhou, Y. Xu, H. Cheng, J. Fang, and W. Pedrycz. Global and local structure preserving sparse subspace learning: An iterative approach to unsupervised feature selection. *Pattern Recognition*, 53:87–101, 2016.

C. Zhu, F. Wen, and J. Sun. A rank-order distance based clustering algorithm for face tagging. In *Computer Vision and Pattern Recognition*, 2011.

# A APPENDIX

## A.1 PROOF OF THEOREM 1

*Proof.* Given the two subspaces $V_1$ and $V_2$ of $V$.

*Sufficiency:* Suppose $V = V_1 + V_2$ and $V_1 \cap V_2 = \{\vec{0}\}$. We treat a subspace as a set of vectors. As for the addition of two sets, $V_1 + V_2 = \{v_1 + v_2 \colon v_1 \in V_1, v_2 \in V_2\}$, which means that any vector in $V$ is the sum of a vector from $V_1$ and a vector from $V_2$, and then we should prove the uniqueness of this addition. As for a vector $v \in V$, we assume $v_1 \in V_1$, $v_2 \in V_2$, $v = v_1 + v_2$, and $v_3 \in V_1$, $v_4 \in V_2$, $v = v_3 + v_4$. Since $V_1 \cap V_2 = \{\vec{0}\}$, $v_1 + v_2 = v_3 + v_4 \rightarrow v_1 - v_3 = v_4 - v_2$. Due to $v_1 - v_3 \in V_1$ and $v_4 - v_2 \in V_2$, $v_1 - v_3 = v_4 - v_2 = \vec{0}$. Thus, $v_1 = v_3$ and $v_4 = v_2$.

*Necessity:* Suppose $V$ is the direct sum of $V_1$ and $V_2$. By definition, $V = V_1 + V_2$ holds. If $v \in V_1 \cap V_2$, considering $\vec{0} \in V$, we obtain $v + (-v) = \vec{0}$ with $v \in V_1$ and $-v \in V_2$ by the definition of the direct sum. If v is not a $\vec{0}$, $\vec{0}$ can be represented by $\vec{0} + \vec{0}$ and $v + (-v)$, which is not a unique representation and violates the definition of the direct sum. So, $V_1 \cap V_2 = \{\vec{0}\}$.

## A.2 IMPLEMENTATION DETAILS

Initially, the embeddings of faces in CASIA-Webface (Yi et al., 2014), FaceScrub (Ng & Winkler, 2014), and IJB-B (Whitelam et al., 2017) were obtained by using the ArcFace model (Deng et al., 2019) with ResNet-101. The model was trained on the union of MS-Celeb-1M (Guo et al., 2016) and VGGFace2 (Cao et al., 2018). Then, we used the method in (Wang et al., 2019) to construct the instance pivot subgraph (IPS) as the input of whole framework, and we followed their training setup by setting $k_1 = 80$, $k_2 = 8$, and $u = 8$. The LP(GCN-M) is the same as the link prediction model constructed by GCN-M in (Wang et al., 2019). In particular, the dimensions of the output features of the first three graph convolution layers were all $512$, while that of the last one was $256$. As for OSD, the dimensions of the two learned subspace matrices were all $256 \times 256$. The whole framework was optimized by stochastic gradient descent (SGD) with 0.9 momentum and 0.0005 weight decay. We set the batch size to 16, and the model was trained for 3 epochs. The learning rate was set to $10^{-3}$ at first, and then multiplied by 0.1 after each epoch. The last saved model was used for evaluation, and the evaluation metrics that we used are the same as those in (Wang et al., 2019), namely, normalized mutual information (NMI) and the BCubed F-measure (Amigó et al., 2009). As for the multi-view testing, we simply concatenated face embeddings with matched audio embeddings and constructed IPS to conduct the "Face + Audio" experiment, using the same training process as described above.

## A.3 GENERALITY OF OSD

OSD is designed to learn discriminative representations that can better differentiate images from different categories. Hence, OSD may have the potential to be further extended in other computer vision tasks, such as object classification and face identification. We further studied the generality of OSD by conducting an experiment with respect to object classification on CIFAR-10 and FashionMNIST. According to the results in Table 7, when the neural networks were equipped with OSD, the performance of them was improved. The improvement brought by OSD is even slightly higher than the most common margin-based method (i.e., AM-Softmax), demonstrating the generality of OSD and its potential in terms of learning superior representations.

| Network | CIFAR-10 | | | FashionMNIST | | |
|---|---|---|---|---|---|---|
| | Baseline | AM-Softmax | OSD | Baseline | AM-Softmax | OSD |
| ResNet-32 | 7.55 | 7.47 | **7.35** | 6.80 | 6.71 | **6.43** |
| ResNet-56 | 6.98 | 6.86 | **6.67** | 6.72 | 6.36 | **6.23** |
| ResNet-110 | 6.41 | 6.33 | **6.13** | 6.60 | 6.49 | **6.16** |

Table 7: Top-1 error rates on CIFAR-10 and FashionMNIST. Note that the network architectures (i.e., ResNets) and training details strictly follow the experimental settings on CIFAR-10 in the original paper (He et al., 2016).

### A.4 VISUALIZATION OF NODE FEATURES LEARNED BY OSD

To better explain what OSD learns, we visualize the node features learned by OSD. We randomly sampled 1000 nodes consisting of positive or negative ones to their pivot nodes from the training set, and used t-SNE (Maaten & Hinton, 2008) to perform dimension-reduction and visualize the node features in Fig. 5. Without OSD (Fig. 5, left), the representation of positive and negative nodes is not discriminative enough; the distribution of the negative node features is dispersed, and some positive and negative node features are even mixed. Such learned features may have bad effects on the performance of graph-based face clustering. With OSD-LP(GCN-M) (Fig. 5, right), there is still a clear gap between the learned node features, although we do not specifically design penalties or loss functions to maximize the margin between the two types of nodes.

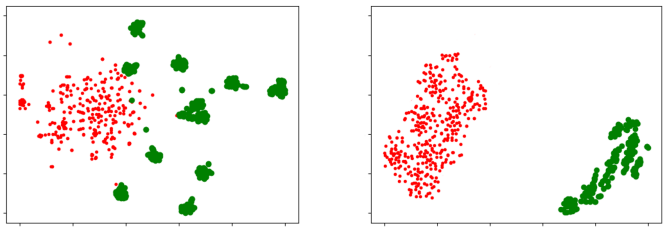

Figure 5: Visualization of node features via the t-SNE method (Maaten & Hinton, 2008). The green and red points denote positive and negative nodes, respectively. Left: node features learned by LP(GCN-M) only; right: node features learned by OSD-LP(GCN-M).