Learnable Subspace Orthogonal Projection for Semi-supervised Image Classification

Lijian Li[1], Yunhe Zhang[1], and Aiping Huang[2]✉

[1] College of Computer and Data Science, Fuzhou University, Fuzhou 350116, China
[2] College of Physics and Information Engineering, Fuzhou University, Fuzhou 350108, China
sxxhap@163.com

Abstract. In this paper, we propose a learnable subspace orthogonal projection (LSOP) network for semi-supervised image classification. Although projection theory is widely used in various machine learning methods, solving projection matrix is a highly complex process. We employ an auto-encoder to construct a scalable and learnable subspace orthogonal projection network, thus enjoying lower computational consumption of subspace acquisition and smooth cooperation with deep neural networks. With these techniques, a promising end-to-end classification network is formulated. Extensive experimental results on real-world datasets demonstrate that the proposed classification algorithm achieves comparable performance with fewer training data than other projection methods.

1 Introduction

Classification is widely used in various tasks [3, 7, 21] to associate data with one or more semantic labels. To boost predictive performance, many of them require to train classifier on a large-scale labeled data. Nevertheless, obtaining sufficient annotated data is time-consuming and laborious. With the impressive success in image classification using Convolutional Neural Networks (CNN), some deep semi-supervised learning methods [9,10,12,14,16,20] are developed. Among these methods, the pseudo label and the consistency regularization are two commonly used techniques. For example, [24] proposed an unsupervised data augmentation for semi-supervised learning by training with consistency regularization. Inspired by curriculum learning, [2] gradually increases the proportion of unmarked samples until all samples are used for model training. SimMatch [29] considers both semantic and instance similarities and enables semi-supervised learning through self-supervised technique. Some advanced menthods are combinations of approaches involving pseudo label, self-supervision and data augmentation, such as MixMatch [1], FixMatch [19], Flexmatch [27], and SimMatch [29]. These methods focus on learning discriminative feature representations and decision boundaries to improve the quality of downstream tasks, creating potential room for improvement of classification network.

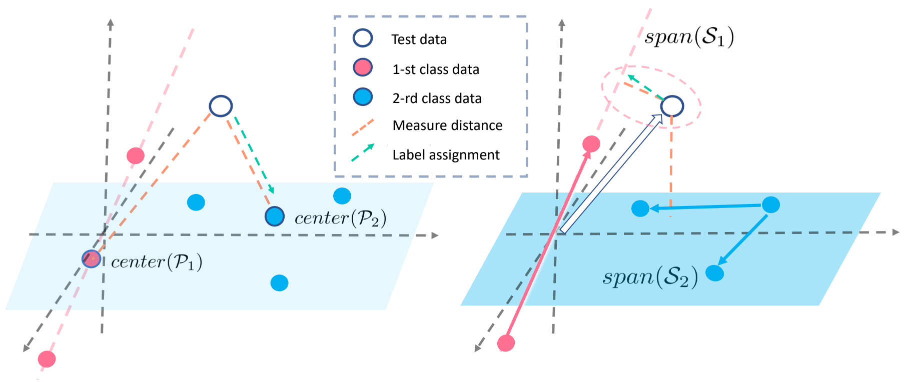


Fig. 1. Comparison of fitting data using a point and a latent subspace. The data are feature vectors in the vector space $\mathbb{R}^{3\times3}$.

Embedding projection theory into deep networks is one of the promising methods for performance improvement. Existing methods normally use a one-dimensional subpace as a center to infer category labels. However, it is possible that features of the same category are distributed in a latent multi-dimensional subspace rather than a point. Fig. 1 shows possible label prediction results in two ways. The point-centered approach may ignore other potential associations among features. It makes more reasonable to assign the test data to the first category. The calculation of the distance to a subspace usually requires solving a projection matrix. However, in these methods the inverse process involved in solution of projection matrix leads to the high computational consumption of subspace acquisition. In addition, deep neural networks usually require multiple iterations, and the small batch processing of data will lead to repeated solution of the projection matrix of each subspace, resulting in further increase in overhead.

In this paper, we propose a learnable subspace orthogonal projection (LSOP) module to reduce the high computational consumption of subspace acquisition. Embedding the module into deep neural network, an effective end-to-end image semi-supervised classification model is subsequently formulated. The whole framework is outlined in Fig. 2. We assume that the similar samples can be distributed around the same latent high-dimensional subspace. This requires each latent subspace to preserve critical features and to be orthogonal to irrelevant features. To this end, a LSOP network is established to learn the corresponding orthogonal projection matrices for these latent subspaces. Followed by defining a classification loss, a deep network architecture for semi-supervised classification is constructed. With these techniques, we make projection theory benefiting from deep learning. Meanwhile, training on high-dimensional subspace reduces the need for label data, which alleviates the scarcity of labeled samples for semi-supervised classification. The main contributions of this paper are summarized as follows:

- The learnable subspace orthogonal projection (LSOP) network. We propose to learn orthogonal projections for high-dimensional subspaces, thus reducing the computational consumption of subspace acquisition and smooth cooperation with deep neural network.
- The extensibility of LSOP network. The proposed LSOP network is generalizable and could be a feasible solution to other orthogonal projection related deep learning, that is beyond the scope of this work.
- The end-to-end semi-supervised classification network. We employ the proposed LSOP network to construct an effective semi-supervised classifier. This facilitates training the classifier on high-dimensional subspaces to alleviate the scarcity of labeled samples.

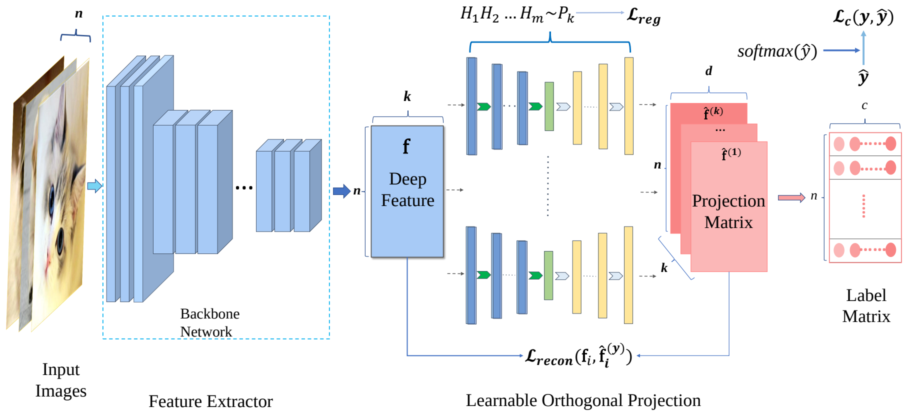


Fig. 2. An overview of the proposed classification method of learnable orthogonal projection. In the figure, we only use an input vector as an example. The parameters of each fully connected layer network are treated as a learnable orthogonal projection matrix of various subspaces. The orthogonal projection matrix is trained for each class of data subspaces by reconstruction loss. Then classification loss is used to keep these subspaces as far away from the other classes of data as possible.

2 Related work

Projection theory has been widely used in many deep network models [4–6, 13, 15, 17, 18, 22, 25]. For example, [5] uses the orthogonal projection space spanned by a principal component analysis projection matrix to alleviate the discrimination difficulty. [25] proposes an auto-encoder framework based on an orthogonal projection constraint for anomaly detection. [4] employs orthogonal projection to design a subspace attention module for denoising. The orthogonal projection is more of an effective method with explicit interpretation for many tasks.

Especially, it is a very practical approach to classification. [28] proposed a Capsule Projection Network (CapProNet) to classify deep features by one type of orthogonal projections. This article uses the closed-form solution of a projection matrix to obtain a projection. The parameters learned in the network is a weight matrix $W_l \in \mathbb{R}^{k \times d}$. The columns of the matrix form a basis of a subspace, *i.e.*, $S_l = span(W_l)$ is spanned by the columns vectors of W_l. The orthogonal projection v_l of a vector x onto subspace $\mathcal{S}_l$ has following solution:

$$v_l = P_l x, \text{ and } P_l = W_l (W_l^T W_l)^{-1} W_l^T. \tag{1}$$

Then the length of projection $||v_l||_2$ is used to measure the affinity of a class. The operations of this procedure are differentiable, so the weight matrix W_l of each subspace can be updated. This paper computes matrix inverse with a hyper-power sequence to alleviate the computational consumption of matrix inverse. However, the consumption also increases significantly with the dimension d of subspace increases. To this end, [23] proposed a Matrix Capsule Convolution Projection (MCCP) module by replacing deep features with a feature matrices. This article reduces the dimensionality of vectors, making it easy to compute projections.

3 Proposed Method

In this section, we aim to construct a LSOP network and then formulate an end-to-end semi-supervised model for image classification.

3.1 Orthogonal Projection

A deep neural network $\mathcal{N}$ can be generally factorized into two phases: feature representation $\mathcal{N}_{fea}$ and data classification $\mathcal{N}_{cla}$. Given a set of data points $\mathcal{X} = \{\mathbf{x}_i, \mathbf{y}_i\}_{i=1}^n$ where $\mathbf{x}_i \in \mathbb{R}^m$ and its label vector $\mathbf{y}_i \in \{0, 1\}^c$, with m being the input dimension and c the number of classes. $\mathbf{f}_i = \mathcal{N}_{fea}(\mathbf{x}_i) \in \mathbb{R}^k$ is a k-dimensional deep feature vector, and $\widehat{\mathbf{y}}_i = \mathcal{N}_{cla}(\mathbf{f}_i) \in \mathbb{R}^c$ is an estimator of $\mathbf{y}_i$. For brevity, we assume that $\mathcal{N}_{cla}$ is a linear classifier, *i.e.*, $\widehat{\mathbf{y}}_i = \mathbf{W}^T \mathbf{f}_i + \mathbf{b}$ for all $i = 1, \cdots, n$, where $\mathbf{W} \in \mathbb{R}^{k \times c}$ is a weight matrix and $\mathbf{b}$ is a bias of the last layer in $\mathcal{N}$. In this phase, we try to minimize $\sum_{i=1}^n loss(\mathbf{y}_i, \widehat{\mathbf{y}}_i) = \sum_{i=1}^n loss(\mathbf{y}_i, \mathbf{W}^T \mathbf{f}_i + \mathbf{b})$, where $loss(\cdot, \cdot)$ is a metric to evaluate the difference between two elements. Letting $\mathbf{W} = [\mathbf{w}_1, \cdots, \mathbf{w}_c]$ with $\mathbf{w}_i \in \mathbb{R}^k$, the classification module $\mathcal{N}_{cla}(\mathbf{f}_i)$ can be interpreted as computing a modified distance of $\mathbf{f}_i$ to c spaces spanned by $\mathbf{w}_i$. The above description is the simplest linear classifier used in various deep networks. We would like to construct a multi-dimensional subspace to obtain effectiveness. Intuitively, each class is represented using a k-dimensional feature vector $\mathbf{w}_i$, which motivates us to characterize a class more accurately by a spanned space of several vectors, instead of only one vector.

Letting $\mathcal{F} = \{\mathbf{f}_i\}_{i=1}^n$. For any $j \in \{1, \cdots, c\}$, we denote $\mathcal{S}_j = \{\mathbf{f}_i \in \mathcal{F} : \mathbf{x}_i \text{ belongs to the } j\text{-th class}\}$, and $\mathcal{S} \in \mathbb{R}^{k \times d_j}$ is a matrix containing all elements of $\mathcal{S}_j$ as columns with $d_j = |\mathcal{S}_j|$. For any test data point $\mathbf{x}$, its low-dimensional feature vector $\mathbf{f} \in \mathbb{R}^k$ can be obtained by a feature representation $\mathcal{N}_{fea}$ with $\mathbf{f} = \mathcal{N}_{fea}(\mathbf{x})$. Naturally, for any test sample $\mathbf{x}$, we try to project its feature vector $\mathbf{f}$ onto the spanned subspaces by $\{\mathcal{S}_j\}_{j=1}^{c}$, *i.e.*,

$$span(\mathcal{S}_j) = \left\{ \sum_{i=1}^{t} \lambda_i \mathbf{v}_i : t \in \mathbb{N}, \mathbf{v} \in \mathcal{S}_j, \lambda_i \in \mathbb{R} \right\}. \tag{2}$$

Accordingly, the probability of the data point $\mathbf{x}$ belonging to the j-th class is defined as the distance of $\mathbf{f}$ to $span(\mathcal{S}_j)$. Actually, this distance is equal to the distance between $\mathbf{f}$ and its projection point onto $span(\mathcal{S}_j)$, denoted as $\mathcal{P}_{span(\mathcal{S}_j)}(\mathbf{f})$. By specifying Euclidean distance, it is expressed as $||\mathbf{f} - \mathcal{P}_{span(\mathcal{S}_j)}(\mathbf{f})||_2$.

The projection point $\mathcal{P}_{span(\mathcal{S}_j)}(\mathbf{f})$ can be regarded as the minimum distance of $\mathbf{f}$ onto the space $span(\mathcal{S}_j)$, *i.e.*,

$$\mathcal{P}_{span(\mathcal{S}_j)}(\mathbf{f}) = \arg \min_{\mathbf{g} \in span(\mathcal{S}_j)} ||\mathbf{f} - \mathbf{g}||_2. \tag{3}$$

Correspondingly, the projection point can be given by

$$\mathcal{P}_{span(\mathcal{S}_j)}(\mathbf{f}) = \mathbf{S}_j (\mathbf{S}_j^T \mathbf{S}_j)^{-1} \mathbf{S}_j^T \mathbf{f} = \mathbf{S}_j \mathbf{S}_j^{\dagger} \mathbf{f}, \tag{4}$$

where $\mathbf{S}_j^{\dagger}$ is the Moore-Penrose pseudo inverse, and $\mathcal{P}_{span(\mathcal{S}_j)} \doteq \mathbf{S}_j (\mathbf{S}_j^T \mathbf{S}_j)^{-1} \mathbf{S}_j^T \doteq \mathbf{S}_j \mathbf{S}_j^{\dagger}$ is the orthogonal projection matrix. It is observed that the sampled data points in the j-th class form a basis, though not necessarily orthonormal. Accordingly, this formulation comes with the following two merits. On the one hand, each class is approximated by a spanned subspace of some basis vectors, rather than a vector. On the other hand, the spanned subspace may exhibit a powerful representation ability, even if several data points are used, which is tailored for semi-supervised learning.

Nevertheless, the closed-form solution mentioned above requires the computational complexity of $\mathcal{O}(\sum_{j=1}^{c} d_j^3 + kd_j^2)$, which is evidently unaffordable when dealing with large-scale datasets. That motivates us to develop a learnable orthogonal projection method to optimize the projection computation in a deep learning network.

3.2 Learnable Orthogonal Projection

In this subsection, we attempt to solve an orthogonal projection matrix in a differentiable manner so that the proposed method serves as a module in deep neural networks. According to Eq. (3), when $\mathbf{f}_i$ belongs to $\mathcal{S}_j$, its projection onto subspace $span(\mathcal{S}_j)$ is exactly equal to itself. Accordingly, for each class, we can find the projection matrix of subspace $span(\mathcal{S}_j)$ by minimizing

$$J(\mathcal{P}_{span(\mathcal{S}_j)}) = \sum_{i=1}^{n} \left\| \mathbf{f}_i - \mathcal{P}_{span(\mathcal{S}_j)} \mathbf{f}_i \right\|_2^2, \tag{5}$$

where $\mathcal{P}_{span(\mathcal{S}_j)}$ is a projection square matrix. We use a neural network to approximate this problem, and the whole network can be considered as a surrogate of the projection matrix. A single-layered neural network is insufficient, because it can be observed from this objective function that the optimal solution may fall into a trivial case, *i.e.*, $\mathcal{P}_{span(\mathcal{S}_j)} = \mathbf{I}$. This is to say that the weights of the neural network tend to converge to the projection matrix of the original vector space. Actually, data points belonging to the same class are not necessarily distributed in only a low-dimensional vector space, thus $\mathcal{S}_j$ tends to expand to a large vector space. We expect to acquire a latent subspace that can contain data points of their associated class and is orthogonal to other subspaces produced by other points. Therefore, a multi-layered network is employed to learn hierarchical subspace structures.

We can use a network structure similar to the matrix structure $\mathbf{S}_j(\mathbf{S}_j^T\mathbf{S}_j)^{-1}\mathbf{S}_j^T$. We denote a subspace basis as $\mathcal{S} \in \mathbb{R}^{k\times d}$, where d is the subspace dimension and k is the full space dimension ($d < k$). Then we set the hiddle layer dimension to d and the input dimension to k. The network structure is like an undercomplete auto-encoder, which overcomes trivial solution. Consequently, it is formulated as the architecture for the projection matrix using m layers:

$$\mathcal{P}_{span(\mathcal{S}_j)} \leftarrow H_1 H_2 \cdots H_m, \tag{6}$$

where H represents the weights of layers. For the j-th class, we denote the output of the network as $f_{\theta_j}(\cdot)$, and then train the network with the following loss:

$$\min_{\theta} \sum_{i=1}^{n} loss(f_\theta(\mathbf{f}_i), \mathbf{f}_i), \tag{7}$$

where $loss(\cdot,\cdot)$ is an alternative loss function, not limited to the mean square error. In this way, we construct a learnable orthogonal projection for each class, which maps all data points onto a subspace so that discriminative components are preserved. It is worth noting that the learnable orthogonal projection can be flexible to be embedded into multiple network structures. After the network is trained, any test data point is projected onto each subspace, and its projection residual represents the probability in the corresponding class.

The time complexity of the feedforward calculation of the network relies on matrix multiplication, and the process of network training avoids the matrix inverse operation of solving the projection matrix directly. Accordingly, the computational complexity of neural networks with back prorogation is $\mathcal{O}(\sum_{j=1}^{c} d_j^2 + kd_j^2)$. Therefore, this approach is more competitive in term of running time as the size of input data increases. In addition, this network can be solved by a mini-batch approach instead of putting all samples into the memory, which reduces the memory requirement.

3.3 Optimization Method

In this subsection, we elaborate how to build an end-to-end model for learnable orthogonal projection. Following Eq. (7), we construct projection matrices for each latent subspace. For simplicity, a projection reconstruction loss with ridge regression is given by:

$$\lambda\mathcal{L}_{recon} + \mu\mathcal{L}_{reg} \doteq \sum_{i=1}^{n} \left\| \mathbf{f}_i - f_{\theta_{(j=y_i)}}(\mathbf{f}_i) \right\|_2^2 + \lambda \left\| \theta \right\|_2^2 . \tag{8}$$

Except the two losses above, we can add additional objective functions to exploit rich hierarchical semantic information from a given data. As an example, cross-entropy can be used to maximize intra-class information and minimize inter-class information in the subspace to be optimized. The predicted class $\widehat{\mathbf{y}}_i$ of the sample $\mathbf{x}_i$ is regarded as the probability distribution of the residual of the projection onto each class subspace. Simultaneously, we define the normalized predicted class probability as

$$\mathbf{p}_i = softmax(\widehat{\mathbf{y}}_i). \tag{9}$$

Accordingly, the classification loss specified as cross-entropy is given by

$$\mathcal{L}_c = -\frac{1}{n}\sum_{i=1}^{n} \mathbf{y}_i \log(\mathbf{p}_i). \tag{10}$$

Minimizing the cross-entropy loss makes the subspaces as orthogonal as possible. Otherwise, the probability matrix will be smoothed, then it can not close to the true label matrix (δ distribution). As mentioned above, the overall loss function is defined as

$$\mathcal{L} = \mathcal{L}_c + \lambda\mathcal{L}_{recon} + \mu\mathcal{L}_{reg}. \tag{11}$$

Gathering all above analyses, the procedures for solving the learnable orthogonal projection are summarized in Algorithm 1.

Unlabeled data training: For unlabeled data training in clustering method, we replace the clustering centers with our subspaces. The subspaces are first constructed with labeled samples. Then we compute the closest subspaces $\mathbf{S}_j$ of unlabeled samples, giving the unlabeled samples a pseudo-label j to update the network. The closest subspace is recalculated in next iteration. The subspace clustering is similar to k-meams. The method enables the subspace classification boundaries to span a low-density region by minimizing the conditional entropy of the class probability of unlabeled data. These unlabeled samples can improve the ability of backbone network to extract features. It also allows the learnable parameters to fix better subspaces. We update the pseudo-labels for unlabeled samples at each iteration and then train at a smaller weight with labeled samples.

4 Experiment

In this section, we conduct comprehensive experiments on classification datasets compared with several state-of-the-art deep neural networks to validate the effectiveness of the proposed method.

Algorithm 1 Learnable Subspace Orthogonal Projection (LSOP)

```
Require: Mimi-batches of the input data $\mathcal{X} = \{\mathbf{x}_i, \mathbf{y}_i\}_{i=1}^n$, the number of classes $c$,
    regularization coefficient $\lambda$, balancing parameter $\mu$ and learning rate $\alpha$, maximum
    epoch of training $max\_epoch$.
Ensure: Neural network parameters $\theta$.
 1: Initialize neural network parameters $\theta_j$ of $c$ subspaces;
 2: for $t = 1$ to $max\_epoch$ do
 3:     for each minbatch do
 4:         Calculate projections onto each subspace by forward propagation;
 5:         Calculate reconstructed loss by Eq. (8);
 6:         Compute label matrix $\{\widehat{\mathbf{y}}_i\}$ by Eq. (9);
 7:         Calculate classification loss by Eq. (10);
 8:         Accumulate overall loss $\mathcal{L}$ by Eq. (11);
 9:         Update network parameters $\theta$ by back propagation;
10:     end for
11: end for
12: return Neural network parameters $\theta$.
```

4.1 Experiment Setting

Datasets. We use two benchmark datasets in experiments to evaluate the classification performance. The CIFAR dataset contains 60,000 colored natural images of 32×32 pixels. CIFAR10 and CIFAR 100 are divided into 10 and 100 classes, respectively. There are 50,000 images in training set, and 10,000 images in test set. For preprocessing, we normalize the data using the channel means and standard deviations. We use all 50,000 training images for the final run and report the final test error at the end of training. Following [8], we give rise to augmentations of CIFAR10 and CIFAR100, and denote augmentation datasets as CIFAR10+ and CIFAR100+, respectively. In semi-supervised classification, we randomly extract labeled images from each class, and the test error is reported.

Compared methods. We test different classifiers in the network architectures such as ResNet [8], WideResNet [26] and DenseNet [11]. We use ResNet with 110 layers and WideResNet ($k = 8$) with 16 layers, as well as DenseNet-BC ($k = 12$) with 100 layers for CIFAR. The last layer of these networks generally serves as a function of classifying deep features. Our classifier can be trained in an end-to-end manner with the backbone network instead of trained independently. The CapProNet [28] and MCCP [23] can also serve as a classifier, and we apply it to these networks with their default setting. CapProNet [28] and MCCP [23] take advantage of an iterative algorithm to update the projection matrix. In contrast, we update weights using a neural network approach without additional matrix operations. We compare the effectiveness of different classifiers to evaluate the performance of the proposed method. In addition, Meta Pseudo Label [16] is an effective regularization strategy. We also apply LSOP to this approach to verify the effectiveness of our method.

Model implementation and training. We implement our code using PyTorch, and all experiments are run on a computer with a Tesla P100 GPU in

Linux systems. We create c-way fully connected layers to replace the last layer of the original network. The 32×32 size images from CIFAR dataset are extracted by backbone networks, and the dimensions of the deep features are tuned as 64, 512 and 342, respectively. We initialize the input and output sizes of our fully connected layer with these values, and the hidden layer dimension is fixed as 8. The parameters of CapProNet are often d times larger than the original single-layered neural network, where d is a dimension of subspace. All parameters in compared deep neural networks are adopted as their default settings in their original papers, and the additional parameters of the final classifier are uniformly set to $c \times k \times d$. In order to provide a fair comparison with CapProNet, we also use a three-layered fully connected layer neural network to fit the projection matrix. Sometimes $S_j^T S_j$ may be irreversible. Therefore, we use a tiny bias on $S_j^\top S_j$ during the experiment to ensure its reversibility. A deep feature matrix $n \times k$ is fed to c-way fully connected layers, then we obtain a projection tensor of the size $n \times c \times k$. We use the defined projection probability distribution to generate a $n \times c$ label prediction matrix.

The Mini-batch stochastic gradient descent is used to train all the networks. On CIFAR datasets, the networks are trained with 64 batches of 300 epochs. The initial learning rate is set to 0.1, and then it is divided by 10 at 150 and 250 training epochs. We use a weight decay (λ) of 10^{-4} and a momentum of 0.9 without dampening. The hyperparameter μ in loss is set to be 0.1. In semi-supervised learning experiments, the loss weight of unlabeled data is gradually incremented to 1.0 from epoch 30 to epoch 120. In a comparison of Meta Pseudo Label [16], we follow its original setup on WideResNet-28-8, training 50,000 iterations in the PyTorch version.

4.2 Experimental Result

Table 1. A comparison of error rates of backbone networks with CapProNet [28], MCCP [23] and LSOP on CIFAR10 and CIFAR100. Results of MCCP are reported by [23]. The datasets with "+" indicate standard data augmentation (*e.g.* horizontal flip or random crop).

Backbone network	Classifier	CIFAR10	CIFAR10+	CIFAR100	CIFAR100+
ResNet [8]	Baseline	13.63	6.41	44.74	27.22
	CapProNet	**13.25**	5.19	42.76	22.78
	MCCP	-	5.24	-	22.86
	LSOP	13.41	**5.12**	**42.22**	**21.73**
WideResNet $k = 8$ [26]	Baseline	11.33	4.81	34.83	22.07
	CapProNet	10.52	**4.04**	**33.10**	**20.12**
	LSOP	**10.35**	4.77	35.42	21.41
DenseNet-BC $k = 12$ [11]	Baseline	7.86	4.51	**26.40**	22.27
	CapProNet	7.93	**4.25**	28.58	**21.19**
	LSOP	**7.72**	4.86	27.34	21.45

Table 2. A comparison of error rates of backbone networks with CapProNet [28] and LSOP on CIFAR10 of training with 4,000 and 10,000 labels. The numbers with"+ PL" indicate training with pseudo labels and data augmentations.

Backbone network	Classifier	4,000	4,000 + PL	10,000	10,000 + PL
ResNet [8]	Baseline	49.20	23.60	38.03	15.21
	CapProNet	**47.48**	23.26	34.48	14.32
	LSOP	48.40	**22.84**	**32.83**	**14.15**
WideResNet $k = 8$ [26]	Baseline	36.85	19.88	**25.04**	12.85
	CapProNet	**35.21**	**19.42**	25.25	12.53
	LSOP	36.95	19.65	25.29	**12.29**
DenseNet-BC $k = 12$ [11]	Baseline	32.88	19.70	**21.06**	13.20
	CapProNet	32.01	**18.96**	21.82	**12.69**
	LSOP	**31.96**	19.33	21.24	12.88

In Table 1, we compare classification performance of different classifiers trained in an end-to-end manner at the same backbone network. In fully supervised experiments, our approach achieves encouraging performance on ResNet. On other backbone networks, the proposed method also comes with competitive performance. The hidden layer dimension of LSOP is uniformly set as 8 to facilitate comparison. The hidden layer dimension can be considered as a subspace dimension and then a single-layer linear classifier is considered as a linear subspace of $D = 1$. The proposed method achieves promising results on classifying deep features extracted from ResNet with $D = 64$, and comes with comparable performance on WideResnet with $D = 512$ and DenseNet with $D = 342$. Our method aims to design an efficient classifier that can be embedded in advanced methods.

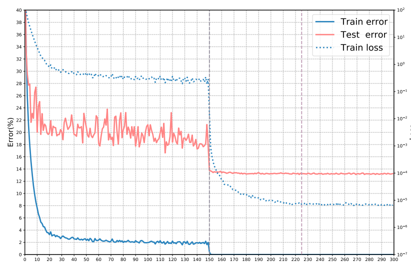
To validate the performance of the proposed method in semi-supervised learning, we report the performance with a part of labeled data in CIFAR10. We use unlabeled data to train networks by pseudo labels and update the networks with lower weights. Table 2 shows that LSOP achieves better performance in semi-supervised learning. In addition, the proposed model is an efficient classifier that can be embedded in advanced methods. We replace the classifier of Meta Pseudo Labels [16] with our designed module and test the performance on CIFAR10 and CIFAR100 in Table 3. We can observe that LSOP achieves better classification accuracy. This shows that our method is a valid module that improves performance with limited labeled data.

Table 3. A comparison of ACC of Meta Pseudo Labels [16] using a linear classifier and LSOP on CIFAR10 with 4,000 training labels and CIFAR100 of training with 10,000 training labels.

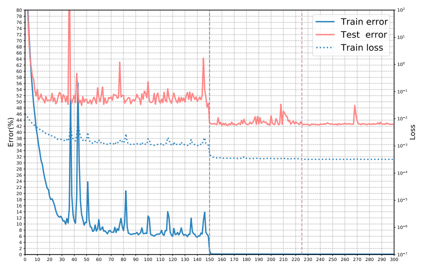
ACC	Meta Pseudo Labels	Meta Pseudo Labels with LSOP
CIFAR10/4k label	95.44	**95.68**
CIFAR100/10k label	77.86	**78.03**

4.3 Convergence and time complexity analysis

Fig. 3 shows the convergence curves of our method on CIFAR10 and CIFAR100. When the learning rate is set to 0.1, the loss converges about 60 epochs. At the 150-th epoch, the loss is further decreased due to the learning rate change.



(a) CIFAR10



(b) CIFAR100

Fig. 3. Convergence curves of LSOP applied to ResNet110. Left is CIFAR10, and right is CIFAR100

We use a single-layer linear classifier as a baseline and consider the linear classifier as a single-dimensional subspace. CapProNet applies multi-dimensional subspace to classify, which brings additional cost. To address the problem, we propose a LSOP network. As shown in Section 3, the computational complexity of the closed-form solution in CapProNet is $\mathcal{O}(\sum_{j=1}^{c} d_j^3 + kd_j^2)$, while the computational complexity of LSOP is $\mathcal{O}(\sum_{j=1}^{c} d_j^2 + kd_j^2)$. Here, k is a deep feature dimension, d_j is the dimension of each latent subspace and c is the number of categories. Compared with the closed-form solution based approach CapProNet, LSOP reduces the computational consumption of subspace acquisition, which facilitates faster optimization for large-scale data.

We also present the time consumption comparison of CapProNet and LSOP to further demonstrate the efficiency of LSOP. Figs. 4&5 show the evolution curves of time consumption with 50,000 samples and 10 categories. The experiments are performed on an AMD Ryzen 9 5900X 12-Core Processor without GPU. In addition, deep features are randomly generated and trained in mini-batch to simulate the training of CIFAR dataset. From the figures, the time consumption of closed-form solution increases significantly with the increase of feature dimension k and subspace dimension d. In contrast, the LSOP method keeps the same level of time consumption as a baseline. This indicates that the proposed method can obtain performance gain by increasing subspace dimensions without computational cost.

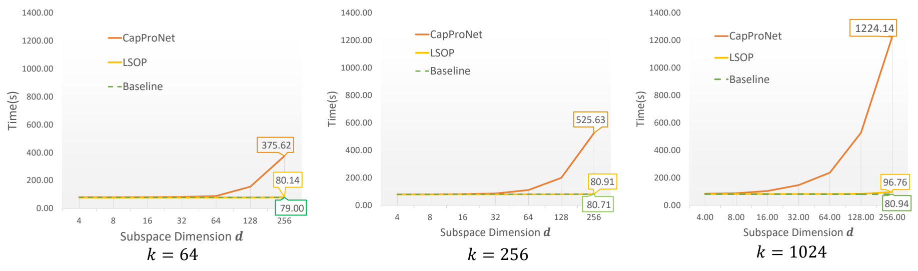


Fig. 4. Run-time comparison as the subspace dimension increases.

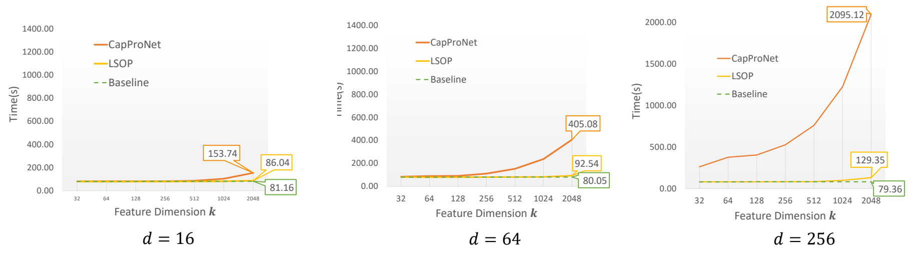


Fig. 5. Run-time comparison as the deep feature dimension increases.

5 Conclusion and Discussion

In this work, we propose a learnable orthogonal projection for deep neural networks that can be trained in an end-to-end fashion. We learn a high-dimensional subspace instead of a vector, bringing acceptable computational consumption. In the proposed method, a latent subspace is used to fit high-dimensional features to train a deep neural network with fewer samples. Although our method is effective in semi-supervised learning, several issues still need to be further explored. For example, the projection reconstruction loss can be constructed in other advanced ways to obtain better subspace projection matrices. Because this network structure is a transformed module, we can use more ways to construct this structure to adapt to various projection subspaces. We hope to propose adaptive network structures to find better projections from more discriminative latent subspaces in future work.

Acknowledgements This work is in part supported by the National Natural Science Foundation of China under Grant U21A20472 and 62276065, and the National Key Research and Development Plan of China under Grant 2021YFB3600503.

References

1. Berthelot, D., Carlini, N., Goodfellow, I., Papernot, N., Oliver, A., Raffel, C.A.: Mixmatch: A holistic approach to semi-supervised learning. In: Proceedings of the Neural Information Processing Systems. pp. 5049–5059 (2019)
2. Cascante-Bonilla, P., Tan, F., Qi, Y., Ordonez, V.: Curriculum labeling: Revisiting pseudo-labeling for semi-supervised learning. In: Proceedings of the 33rd AAAI Conference on Artificial Intelligence. pp. 6912–6920 (2021)
3. Chen, E., Lee, C.: Towards fast and robust adversarial training for image classification. In: Proceedings of the 15th Asian Conference on Computer Vision. pp. 576–591 (2020)
4. Cheng, S., Wang, Y., Huang, H., Liu, D., Fan, H., Liu, S.: NBNet: Noise basis learning for image denoising with subspace projection. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. pp. 4896–4906 (2021)
5. Dou, H., Chen, C., Hu, X., Xuan, Z., Hu, Z., Peng, S.: PCA-SRGAN: incremental orthogonal projection discrimination for face super-resolution. In: Proceedings of the ACM Multimedia Conference. pp. 1891–1899 (2020)
6. Fu, Z., Zhao, Y., Chang, D., Zhang, X., Wang, Y.: Double low-rank representation with projection distance penalty for clustering. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. pp. 5320–5329 (2021)
7. Hatakeyama, Y., Sakuma, H., Konishi, Y., Suenaga, K.: Visualizing color-wise saliency of black-box image classification models. In: Proceedings of the 15th Asian Conference on Computer Vision. pp. 189–205 (2020)
8. He, K., Zhang, X., Ren, S., Sun, J.: Deep residual learning for image recognition. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. pp. 770–778 (2016)
9. He, R., Yang, J., Qi, X.: Re-distributing biased pseudo labels for semi-supervised semantic segmentation: A baseline investigation. In: Proceedings of the IEEE/CVF International Conference on Computer Vision. pp. 6910–6920 (2021)
10. Hu, Z., Yang, Z., Hu, X., Nevatia, R.: Simple: Similar pseudo label exploitation for semi-supervised classification. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. pp. 15099–15108 (2021)
11. Huang, G., Liu, Z., Van Der Maaten, L., Weinberger, K.Q.: Densely connected convolutional networks. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. pp. 4700–4708 (2017)
12. Li, J., Xiong, C., Hoi, S.C.H.: Comatch: Semi-supervised learning with contrastive graph regularization. In: Proceedings of the IEEE/CVF International Conference on Computer Vision. pp. 9455–9464 (2021)
13. Li, X., Lin, C., Li, R., Wang, C., Guerin, F.: Latent space factorisation and manipulation via matrix subspace projection. In: Proceedings of the 37th International Conference on Machine Learning. vol. 119, pp. 5916–5926 (2020)
14. Luo, X., Chen, J., Song, T., Wang, G.: Semi-supervised medical image segmentation through dual-task consistency. In: Proceedings of the 35th AAAI Conference on Artificial Intelligence. pp. 8801–8809 (2021)
15. Mutny, M., Kirschner, J., Krause, A.: Experimental design for optimization of orthogonal projection pursuit models. In: Proceedings of the 34th AAAI Conference on Artificial Intelligence. pp. 10235–10242 (2020)
16. Pham, H., Dai, Z., Xie, Q., Le, Q.V.: Meta pseudo labels. In: IEEE Conference on Computer Vision and Pattern Recognition. pp. 11557–11568 (2021)

17. Ranasinghe, K., Naseer, M., Hayat, M., Khan, S., Khan, F.S.: Orthogonal projection loss. In: Proceedings of the IEEE/CVF International Conference on Computer Vision. pp. 12313–12323 (2021)
18. Saeed, M.S., Khan, M.H., Nawaz, S., Yousaf, M.H., Del Bue, A.: Fusion and orthogonal projection for improved face-voice association. In: Proceedings of the IEEE International Conference on Acoustics, Speech and Signal Processing. pp. 7057–7061 (2022)
19. Sohn, K., Berthelot, D., Carlini, N., Zhang, Z., Zhang, H., Raffel, C., Cubuk, E.D., Kurakin, A., Li, C.: Fixmatch: Simplifying semi-supervised learning with consistency and confidence. In: Proceedings of the Neural Information Processing Systems (2020)
20. Tang, Y., Chen, W., Luo, Y., Zhang, Y.: Humble teachers teach better students for semi-supervised object detection. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. pp. 3132–3141 (2021)
21. Varamesh, A., Tuytelaars, T.: Mix'em: Unsupervised image classification using a mixture of embeddings. In: Proceedings of the 15th Asian Conference on Computer Vision. pp. 38–55 (2020)
22. Wang, J., Feng, K., Wu, J.: SVM-based deep stacking networks. In: Proceedings of the 31st AAAI Conference on Artificial Intelligence. pp. 5273–5280 (2019)
23. Xiang, C., Wang, Z., Tian, S., Liao, J., Zou, W., Xu, C.: Matrix capsule convolutional projection for deep feature learning. IEEE Signal Process. Lett. **27**, 1899–1903 (2020), doi: 10.1109/LSP.2020.3030550
24. Xie, Q., Dai, Z., Hovy, E.H., Luong, T., Le, Q.: Unsupervised data augmentation for consistency training. In: Proceedings of the Neural Information Processing Systems. pp. 6256–6268 (2020)
25. Yu, Q., Kavitha, M.S., Kurita, T.: Autoencoder framework based on orthogonal projection constraints improves anomalies detection. Neurocomputing **450**, 372–388 (2021)
26. Zagoruyko, S., Komodakis, N.: Wide residual networks. In: Proceedings of the British Machine Vision Conference. pp. 1–12 (2016)
27. Zhang, B., Wang, Y., Hou, W., Wu, H., Wang, J., Okumura, M., Shinozaki, T.: Flexmatch: Boosting semi-supervised learning with curriculum pseudo labeling. In: Advances in Neural Information Processing Systems. pp. 18408–18419 (2021)
28. Zhang, L., Edraki, M., Qi, G.: Cappronet: Deep feature learning via orthogonal projections onto capsule subspaces. In: Proceedings of the Neural Information Processing Systems. pp. 5819–5828 (2018)
29. Zheng, M., You, S., Huang, L., Wang, F., Qian, C., Xu, C.: Simmatch: Semi-supervised learning with similarity matching. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 14471–14481 (2022)