Multimedia Tools and Applications
https://doi.org/10.1007/s11042-024-19589-8

Template-centric deep linear discriminant analysis for visual representation

Zongkai Chai[1] · **Liantao Wang**[1] · **Haowen Shi**[1] · **Zhaohui Yuan**[1]

Received: 22 February 2024 / Revised: 19 May 2024 / Accepted: 3 June 2024

Abstract
In some real-world visual recognition tasks, instances are generated according to certain standards, which should serve as references during instance recognition. In this paper, we propose a template-centric representation learning (TCRL) framework that uses these standards as templates during recognition. The TCRL framework aims to learn a feature space where each instance is closely centered around its own template and away from the other templates. Within TCRL framework, we propose a template-centric objective function and a template-centric LDA layer, comprising two concrete models TDCNN and TDLDA. Experiments show that our method is superior to other traditional classification methods. The code will be made public after acceptance.

Keywords Template-centric representation learning · Template-centric LDA · Deep learning · Visual representation

1 Introduction

In some real-world visual recognition tasks, instances are generated according to certain standards. For example, directional arrows in road lanes, whose recognition is crucial for intelligent traffic systems [1, 2], must adhere strictly to national standards during their production process. These standards can be seen as templates that can serve as references during recognition as seen in Fig. 1. The instance should be recognized as the class that best matches a specific template. Similar tasks include handwritten character recognition. When we learn to write characters, we always use standard printed fonts as references, which should also serve as templates for recognition.

In fact, there exists a classical visual recognition technique refered template matching [3–5]. In template matching, the objective to identify instances of a given template image within another image by applying thresholding to specific similarity metrics. The most commonly used metric is the sum-square-difference; however, it is limited in its ability to handle transformations beyond 2D translations. While alternative metrics like normalized cross-correlation

✉ Liantao Wang
ltwang@hhu.edu.cn

[1] College of Information Science and Engineering, Hohai University, 1915 Hohai Avenue, Changzhou 213200, Jiangsu, China

(a) (b)

Fig. 1 Templates and their instances. (a) shows a standard road marking of direction arrow from the Chinese National Standard, and its instances acquired from different perspectives. (b) shows a standard printed number 6, and its handwritten instances

are robust in terms of signal strength, they may struggle to accommodate the intricate variations present in real-world scenarios.

On the other hand, instances generated from an identical template can undergo significant transformations during construction and acquisition. Let's consider directional arrows on road markings as an example. While the production of arrow markings must adhere to national standards, variations often occur during acquisition due to differences in perspective. As depicted in Fig. 1, when we examine a standard directional template, its instances within images can exhibit significant disparities. These variations may arise due to diverse acquisition purposes; for instances, lower perspectives for self-driving applications, higher perspective for intelligent traffic guidance, or traffic violation detection.

In this paper, we seek a feature transformation space where instance recognition can be achieved through similarity-based template matching. To achieve this, we require that the feature representation in this space satisfies the following conditions: each instance is close to its own template and relatively distant from other templates. For this type of feature transformation, the extensive research on linear discriminant analysis (LDA) can offer many insights.

Based on these findings, we aim to find a feature space where different instances from each template are closely clustered around that template while being far from the other templates. To obtain such a feature representation, we combine DCNNs and LDA extensively and propose a template-centric feature representation learning framework. Within this framework, we introduce two different network models. Our contributions can be summarized in three aspects:

- We propose a framework for template-centric representation learning. The objective function encourages the network to learn feature representations that have minimal intra-class variation relative to the template and maximal inter-class differences.
- We introduce two network models, TDCNN and TDLDA. They conduct feature learning at different levels. The TDCNN model directly performs feature learning on the output of DCNNs, while the TDLDA model adds an LDA layer after DCNNs to further refine feature learning.
- We formulize the intra-class and inter-class scattering for template-centric representation learning, and integrate hard-negative mining of template matching.

2 Related work

Our main objective is to learn a meaningful representation of the data and, through metric learning, better measure the differences and similarities between samples during the training

process, enabling the handling of tasks such as classification and clustering. The success of deep learning in recent years depends not only on the algorithms themselves but also on the representation of the data. Representation learning, as a crucial concept in machine learning [6], focuses on how to effectively extract information from complex data, eliminating irrelevant details to better capture the structure and information inherent in the data, forming features.Representation learning primarily includes various approaches such as supervised, unsupervised [7], self-supervised [8], and weakly supervised [9]. Supervised learning methods often acquire stronger feature information compared to other unsupervised methods. Our work specifically concentrates on the classification problem of datasets derived from fixed templates. We use the templates as labels for supervised training, allowing the original data to undergo nonlinear transformations and become a better representation of the data.

Obtaining a better representation of the data will greatly assist us in subsequent classification or clustering tasks. Traditional machine learning methods are limited when dealing with raw data; they require feature engineering to process and extract features before classification or clustering tasks [10]. In recent years, with the development of deep learning, we can directly learn higher-level representations of data in classification tasks. Commonly used softmax loss and its variants have significantly contributed to the success of deep neural networks in classification tasks [11, 12]. However, softmax loss places excessive emphasis on the correctness of classification, whereas we aim to obtain representations based on template features. Methods involving metric distance learning [13, 14] or triplet loss [15] have been employed to learn powerful representations. These approaches focus on capturing meaningful relationships between data points, which is particularly valuable for our objective of better representing data based on template features.

We draw inspiration from some supervised traditional metric learning methods such as LDA. The original LDA aimed to seek a linear projection that makes the transformed features more intra-class coherent and inter-class separated. Recent research has tended to combine this idea with powerful deep convolutional neural networks (DCNNs). Dofer et al. [16] combine the feature representation of DCNNs with the linear projection of LDA by maximizing the smallest eigenvalues, obtaining a low-dimensional feature space representation through end-to-end training. Peng et al. [17] based on the same idea of maximizing inter-class distance while minimizing intra-class distance, propose a new loss function to optimize DCNNs, which outperforms the cross-entropy loss function. Bartan et al. [18] extend LDA using neural networks, and find the optimal two-layer neural network that embeds data points to optimize the same discrimination criterion. Chang [19] combines LDA and the space-folding operation of deep feedforward neural networks to achieve better classification.

To obtain better template representations, TCRL seeks a transformation from a high-dimensional feature space to a lower dimension, where instances can better match their templates. In [20] Yan et al. designed a direct computation of the similarity distance with the target for matching. In [21], Cao et al. minimize the sum of squared L2 norms between the predicted and ground truth vectors to find an optimal mapping function, ensuring that the estimates are as close as possible to the target values for all training images. Furthermore, there exist other studies on matching strategies. In [22], Wang et al. extract sub-centroids from training samples to represent class distributions. This method interprets classification decisions by testing the proximity of test data in the feature space to these sub-centroids. In [23], Wang et al. proposed an algorithm encourages image (instance) representations to be equivariant to geometric transformations, thereby achieving more robust instance-query matching.

3 Methodology

Given a set of templates $\{\mathcal{T}_1, \mathcal{T}_2, \cdots, \mathcal{T}_C\}$, there can exist several various instances that are mutated from each template. Suppose there are N_i instances $\{\mathcal{I}_1^{(i)}, \mathcal{I}_2^{(i)}, \cdots, \mathcal{I}_{N_i}^{(i)}\}$ for template $\mathcal{T}_i$, $i = 1, 2, \cdots, C$. Let $\mathrm{x}_j^{(i)} = f(\mathcal{I}_j^{(i)}, \Theta) \in \mathbb{R}^{D\times 1}$, be the hidden feature representation produced by DCNNs for instance $\mathcal{I}_j^{(i)}$, and $\mathrm{T}_i = f(\mathcal{T}_i, \Theta) \in \mathbb{R}^{D\times 1}$ for template $\mathcal{T}_i$, where $f(\mathrm{x}_i, \Theta)$ is a non-linear function representing DCNNs, and Θ model parameters.

3.1 Framework for template-centric representation learning

Here, we draw upon the extensive research achievements of LDA and combine them with the significant advantages of DCNNs for feature representation. This combination allows us to achieve template-centric feature representation learning. We aim for the learning of the feature space to adhere to the following two principles: (i) Each instance of a class should cluster around its own template distribution, and (ii) it should be distant from other templates, as shown in Fig. 2.

For principle (i), we measure it using the squared sum of the Euclidean distances from each instance within a class to its own template, as shown in the following equation:

$$s_{\mathrm{W}} = \sum_{i=1}^{C}\sum_{j=1}^{N_i} \|\mathrm{x}_j^{(i)} - \mathrm{T}_i\|_2^2. \tag{1}$$

For principle (ii), we incorporate the concept of hard negative mining and design the equation as:

$$s_{\mathrm{B}} = \sum_{i=1}^{C}\sum_{j=1}^{N_i} \min_{l\neq i} \|\mathrm{x}_j^{(i)} - \mathrm{T}_l\|_2^2. \tag{2}$$

To achieve template matching, we seek to maximize s_{B} while minimizing s_{W}, and we use Fisher's criterion as the objective function for template-centric representation learning,

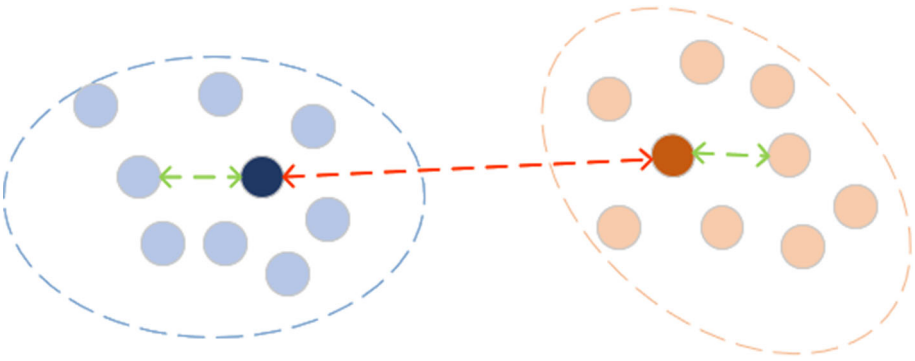


Fig. 2 TCRL principles. Each instance is closely centered around its own template and away from the other templates

which is a Rayleigh quotient to be maximized, denoted as TCRL objective:

$$J(\Theta) = \frac{s_{\mathrm{B}}}{s_{\mathrm{W}}} = \frac{\sum_{i=1}^{C}\sum_{j=1}^{N_i} \min_{l\neq i} \|\mathrm{x}_j^{(i)} - \mathrm{T}_l\|_2^2}{\sum_{i=1}^{C}\sum_{j=1}^{N_i} \|\mathrm{x}_j^{(i)} - \mathrm{T}_i\|_2^2}. \tag{3}$$

Within this framework of template-centric representation learning, we have designed two different convolutional neural network models as shown in Fig. 3: (1) Directly adding our objective function to the DCNNs, requiring the feature output of DCNNs to satisfy the above two principles; (2) Adding an LDA layer after DCNNs, projecting the learned DCNNs features through a linear transformation in a new space to enforce the constraints of the two principles. These two models are referred to as TDCNN and TDLDA, respectively.

3.2 TDCNN

To implement the TCRL framework, the most direct approach is to append the objective function (3) after the DCNNs and require that the output features satisfy the two principles, as shown in Fig. 3. In this case, optimizing the objective function (3) is equivalent to minizing the following loss function (4). This optimization can be directly achieved using back-propagation in an end-to-end manner.

$$l_{\mathrm{TDCNN}}(\Theta) = \frac{\sum_{i=1}^{C}\sum_{j=1}^{N_i} \|\mathrm{x}_j^{(i)} - \mathrm{T}_i\|_2^2}{\sum_{i=1}^{C}\sum_{j=1}^{N_i} \min_{l\neq i} \|\mathrm{x}_j^{(i)} - \mathrm{T}_l\|_2^2}. \tag{4}$$

This design contrasts with the classic DCNNs classification methods based on Cross-Entropy loss (CE-DCNN). CE-DCNN has the following two issues: (1) CE loss does not explicitly control the ratio of inter-class distance to intra-class distance, which can lead to inter-class distances even smaller than intra-class distances. (2) The classifier trained with CE loss can only recognize fixed categories and cannot extend to unseen categories. TDCNN has advantages in both of these aspects compared to CE-DCNN.

Once the model is trained, we can use it to recognize test instance x. Due to the learning of feature representations that make instances cluster around their own template and stay

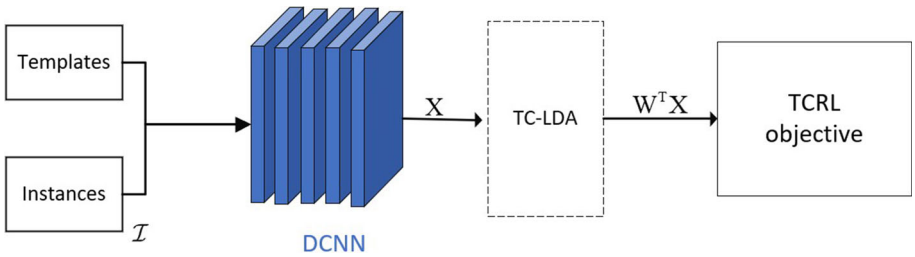


Fig. 3 TCRL framework. The TCRL framework aims to learn a feature representation that satisfies the two principles. One simple implementation is to directly add the template-centric objective function to DCNNs (without the dashed box), while another is to insert a TC-LDA layer after DCNNs

far from the other templates, we can use nearest-neighbour for classification. That is, we recognize x as the category of its closest template.

$$\arg\min_i \|\mathrm{x} - \mathrm{T}_i\|_2^2. \tag{5}$$

3.3 TDLDA

3.3.1 Design of the LDA layer

In this scenario, a template-centric LDA layer (TCLDA) is added after the DCNNs to perform linear projection of features, as shown in Fig. 3.

For an instance with DCNNs feature x, the final output feature becomes $\mathrm{W}^\top\mathrm{x}$. Consequently, the (1) for principle (i) becomes:

$$\begin{aligned} s_\mathrm{W} =& \sum_{i=1}^{C}\sum_{j=1}^{N_i} \|\mathrm{W}^\top\mathrm{x}_j^{(i)} - \mathrm{W}^\top\mathrm{T}_i\|_2^2 \\ =& \mathrm{Tr}(\mathrm{W}^\top\mathrm{S}_\mathrm{W}\mathrm{W}), \end{aligned} \tag{6}$$

where we denote the scattering matrix as

$$\mathrm{S}_\mathrm{W} = \sum_{i=1}^{C}\sum_{j=1}^{N_i}\left(\mathrm{x}_j^{(i)} - \mathrm{T}_i\right)\left(\mathrm{x}_j^{(i)} - \mathrm{T}_i\right)^\top. \tag{7}$$

The difference of (6) lies in the center. The conventional within-class scattering matrix is computed around their class means, while the template-centric scattering is computed around the target template.

Due to the inclusion of TCLDA layer, the (2) for principle (ii) becomes:

$$\begin{aligned} s_\mathrm{B} =& \sum_{i=1}^{C}\sum_{j=1}^{N_i} \|\mathrm{W}^\top\mathrm{x}_j^{(i)} - \mathrm{W}^\top\mathrm{T}_{g(\mathrm{x}_j^{(i)})}\|_2^2 \\ =& \mathrm{Tr}(\mathrm{W}^\top\mathrm{S}_\mathrm{B}\mathrm{W}). \end{aligned} \tag{8}$$

Similarly, we denote the scattering matrix as

$$\mathrm{S}_\mathrm{B} = \sum_{i=1}^{C}\sum_{j=1}^{N_i}\left(\mathrm{x}_j^{(i)} - \mathrm{T}_{g(\mathrm{x}_j^{(i)})}\right)\left(\mathrm{x}_j^{(i)} - \mathrm{T}_{g(\mathrm{x}_j^{(i)})}\right)^\top. \tag{9}$$

where

$$g(\mathrm{x}_j^{(i)}) = \arg\min_{l\neq i} \|\mathrm{x}_j^{(i)} - \mathrm{T}_l\|_2^2. \tag{10}$$

finds the nearest template from those not the target. The conventional between-class scattering matrix is class-wise, and measures the scattering of two class mean vectors. While (8) is an instance-to-template matrix, and measures scattering of each instance around the nearest non-target template. We call it min-neg scattering and validate its effectiveness in the experiments. Then, Fisher's criterion is employed as the objective function for TCLR, which is a Rayleigh quotient (3) to be maximized. The objective function (3) for TCLR takes a specific form

when using the TDLDA network structure, denoted as:

$$J(\Theta, \mathrm{W}) = \frac{s_{\mathrm{B}}}{s_{\mathrm{W}}} = \mathrm{Tr}(\frac{\mathrm{W}^{\top}\mathrm{S_B W}}{\mathrm{W}^{\top}\mathrm{S_W W}}). \tag{11}$$

When Θ is fixed, it is easy to verify that the column vectors of optimal W are the eigenvectors of $\mathrm{S_W^{-1}S_B}$ corresponding to the C-1 eigenvalues. For more derivation details, please refer to [24, 25].

3.3.2 Optimization target

When Θ is fixed,W has closed-form solution in the optimization of (11). In order to better integrate LDA with deep learning, and optimize θ and W simultaneously, we need to reformulate the optimization target. Assume v_i is an eigenvector of matrix $\mathrm{S_W^{-1}S_B}$ and e_i is the corresponding eigenvalue. Each eigenvalue e_i quantifies the separation degree of samples in the direction of its corresponding vector v_i. Therefore, to combine the TCLDA layer with DCNNs, the optimization target should focus on individual eigenvalues. To avoid trivial solutions, following the literature [16], we set the optimization target to maximize the smallest k eigenvalues among all C-1 eigenvalues, as shown in the formula:

$$\max_{\Theta} \frac{1}{k}\sum_{i=1}^{k} e_i \; with \{e_1, \cdots, e_k\} = \{e_j | e_j < \min\{e_1, \cdots, e_{C-1}\} + \epsilon\}. \tag{12}$$

This objective function ensures that the TDLDA model, which combines DCNNs and LDA, can be trained in an end-to-end manner. The intuitive interpretation of this optimization objective is to learn the network parameters of DCNNs to obtain feature representations that are as separable as possible in all dimensions.

3.3.3 Decision phase

After the model has been trained, we now describe how to use it for prediction. Assuming there are N_t test samples, the features obtained after passsing through DCNNs are denote as $\mathrm{X}_t = \{\mathrm{x}_1, \cdots, \mathrm{x}_{N_t}\} \in \mathbb{R}^{D \times N_t}$. For the linear projection matrix of the TCLDA layer, we use all training samples to calculate it to obtain a stable estimate. Specifically, we calculate $\mathrm{S_W^{-1}S_B}$ using the features obtained from all training samples after passing through DCNNs, and then obtain its feature vectors to form the projection matrix $\mathrm{W} = \{\mathrm{v}_1, \mathrm{v}_2, \cdots, \mathrm{v}_{C-1}\} \in \mathbb{R}^{D \times (C-1)}$. Assuming the mean vectors for each class of training samples after DCNNs are denoted as m_i, we can then obtain the matrix $\mathrm{M} = \{\mathrm{m}_1^{\top}, \mathrm{m}_2^{\top}, \cdots, \mathrm{m}_C^{\top}\} \in \mathbb{R}^{C \times D}$. For the test samples X_t, their distances relative to the linear decision hyperplanes are calculated as:

$$\mathrm{D} = \mathrm{X}_t^{\top}\mathrm{T}^{\top} - \frac{1}{2} diag(\mathrm{M}\,\mathrm{T}^{\top}), \;\; \mathrm{T} = \mathrm{MWW}^{\top} \in \mathbb{R}^{\mathrm{C} \times \mathrm{D}}. \tag{13}$$

where, T represents the normal vectors of the decision hyperplanes. Finally, by applying the logistic function and normalizing, we obtain the probabilities of the samples belonging to each class and classify the samples into the class with the highest probability.

3.3.4 Hybrid center

When the instances mutated severely, it is hard to make them centered around the template. We figure out another solution to adjust the projection center by a fraction of the class mean,

and set it to

$$\mathrm{T}_i' = (1-\alpha)\mathrm{M}_i + \alpha \mathrm{T}_i . \tag{14}$$

Note that (14) is a weighted sum of the class mean and the template. With the definition of the new center, the scattering matrices of (7) and (9) become

$$\mathrm{S}_\mathrm{W}' = \sum_{i=1}^{C}\sum_{j=1}^{N_i}\left(\mathrm{X}_j^{(i)} - \mathrm{T}_i'\right)\left(\mathrm{X}_j^{(i)} - \mathrm{T}_i'\right)^\top, \tag{15}$$

$$\mathrm{S}_\mathrm{B}' = \sum_{i=1}^{C}\sum_{j=1}^{N_i}\left(\mathrm{X}_j^{(i)} - \mathrm{T}'_{g(\mathrm{X}_j^{(i)})}\right)\left(\mathrm{X}_j^{(i)} - \mathrm{T}'_{g(\mathrm{X}_j^{(i)})}\right)^\top, \tag{16}$$

where $g(\mathrm{X}_j^{(i)})$ finds the nearest negative center:

$$g(\mathrm{X}_j^{(i)}) = \arg\min_{l\neq i} \|\mathrm{X}_j^{(i)} - \mathrm{T}_l'\|_2^2. \tag{17}$$

When α takes 0, the M_i reduces to class mean that arises in the conventional LDA, and the projection benefits classification. When α takes 1, the M_i becomes T_i, and the projection aims for template matching. The hybrid center of (14) helps the template matching by integrating the class information. It is useful especially when the instances mutation is severe. We will demonstrate its effectiveness with DCNNs in the experiments.

4 Experiments

4.1 Datasets

We assess the performance of our methods by experimenting on our arrow marking dataset [26], MNIST [27], CIFAR-10 [28] and STL-10 [29].

The arrow marking dataset contains a total number of 12,408 images with size of 450×600 pixels, which are manually generated from 11 categories of direction arrows present in Chinese National Standards GB5768.3-2009 and GB51038-2015. The dataset is randomly divided into train-val set with 9933 images, and test set with 2475 images.

MNIST, CIFAR-10 and STL-10 are three standard benchmark datasets. MNIST is a benchmark dataset of handwritten numbers with a size of 28x28 from 0 to 9. The dataset includes 60000 images as a train-val set and 10000 images as a testing set. CIFAR-10 dataset, containing 60,000 color images across ten classes, has become a widely-used benchmark dataset in the computer vision community. The dataset consists of 32x32 pixels natural images and we divide it into 50000 train-val samples and 10000 test samples. STL-10 is a larger-scale dataset which consisting of ten classes compared to CIFAR-10. Each class contain 500 images of 96x96 pixels, resulting in a total of 5000 images including 4000 images for train-val and 1000 images for test.

4.2 Experimental setting

Templates for datasets In the framework of TCRL, templates are needed as references. For arrow marking dataset, we use the direction arrow in Chinese National Standards as templates as shown in Fig. 4. For MNIST, we use standard printed fonts of digits as templates as shown

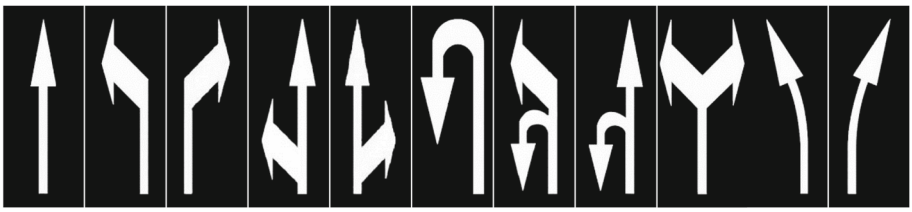


Fig. 4 Standard direction arrow templates

in Fig. 5. We also test different printed fonts, such as SimSun, Arial, Times and so on. For general image datasets without explicit templates, a randomly selected instance is set as the template for its category as illustrated in Fig. 6(a) and (b).

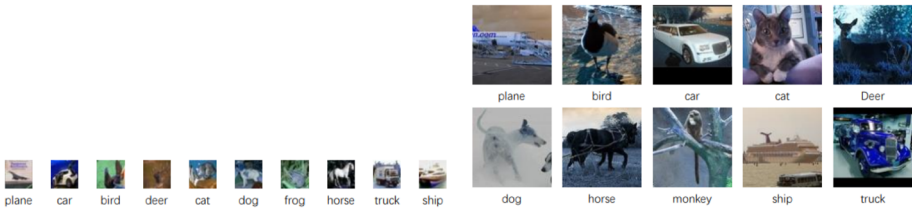
DCNNs Architecture We choose Resnet18 [30] as our DCNNs backbone and make some modifications to it. Different inputs are adjusted based on the specific datasets. Also dropout layer is added to prevent overfitting and batch normalization is helped to increase convergence speed during training. As depicted in Table 1, Model A applied to CIFAR-10 and arrow marking dataset, while for MNIST, we choose Model B, and for STL-10, we choose Model C.

Other settings Batch size is set to 1024. The stochastic gradient descent is adopted with an initial learning rate of 0.001, weight decay of 0.0005 and a momentum of 0.9. All of the hyper-parameter in our method are tuned on validations set. We did not use any data augmentation. The algorithms are implemented using PyTorch [31].

4.3 Results and analysis

Ablation study In order to validate the effect of each component in TCRL, we perform ablation study on arrow dataset. Within the TCRL framework, we proposed two concrete models TDCNN and TDLDA. Compared to TDCNN, TDLDA includes an extra TC-LDA layer. As seen in Table 2, TDLDA performs better than TDCNN, proving the effectiveness of the TC-LDA layer. In the TC-LDA layer, we proposed an instance-to-template scattering

Fig. 5 Templates of MNIST in different Fonts. (a) SimSun (b) KaiTi (c) Arial (d) Calibri (e) Times



(a) Templates of CIFAR-10 (b) Templates of STL-10.

Fig. 6 Randomly selected templates from CIFAR-10 and STL-10

Table 1 Different DCNNs

Layer	ModelA	ModelB	ModelC
Input	32x32	28x28	96x96
Conv1.x	[3x3, 64]x1	[3x3, 64]x1	[3x3, 64]x1
BN	BN-ReLu	BN-ReLu	BN-ReLu
pool_1	2x2 max pool stride 2	2x2 max pool stride 2	2x2 max pool stride 2
Conv2.x	[3x3, 64]x2-Dropout	[3x3, 64]x2	[3x3, 64]x2
Conv3.x	[3x3, 128]x2-Dropout	[3x3, 128]x2-Dropout	[3x3, 128]x2-Dropout
Conv4.x	[3x3, 256]x2-Dropout	[3x3, 256]x2	[3x3, 256]x2-Dropout
Conv5.x	[3x3, 512]x2-Dropout	[3x3, 512]x2-Dropout	[3x3, 512]x2-Dropout
pool_2	Global-avragepool [1x1]	Global-avragepool [1x1]	Global-avragepool [1x1]
FC	64	64	128

[a] BN: Batch Normalization
[b] ReLu: Rectified Linear Activation Function
[c] FC: Fully Connected Layer

Table 2 Ablation experiment

	min-neg	hybrid-center	top-1 score
TDLDA	✓	✓	99.96
	✓		99.88
		✓	99.84
TDCNN			99.76

Table 3 Accuracy on benchmark datasets with DCNNs

Method	MNIST(%)	CIFAR-10 (%)	STL-10(%)
CE-DCNN	99.59	91.17	80.13
Triplet Loss [15]	99.65	90.96	79.79
Center+Softmax [11]	99.62	91.25	80.67
DLDA [16]	99.71	91.80	81.40
I2CS [17]	99.71	91.34	80.33
TDCNN	99.72	92.10	81.43
TDLDA	99.73	92.80	81.65

matrix with hard-negative mining (min-neg), and hybrid-center adjustment. The ablation study in Table 2 shows their effectiveness.

Comparison with other methods We compare TCRL with other classification methods, including DCNNs with Cross-Entropy loss (CE-DCNN), Triplet Loss [15], Center+Softmax [11], DLDA [16], and I2CS [17]. Both models within TCRL framework outperform the baselines on all of the datasets as shown in Table 3. In addition, TDLDA performs better than TDCNN, demonstrating once again the effectiveness of the TC-LDA layer.

Following [22], we also tested several state-of-the-art backbones including both DCNNs and transformer in TCRL. All of them yield promising results as shown in Table 4. DNC [22] represents class distributions using sub-centroids derived from training samples, whereas we use templates as centroids to represent class distributions. Thus we also compare TCRL framework with DNC, and the results are comparable, as demonstrated in Table 5.

Visualization To analyze the reasons for the good performance, we visualize the learned feature representation for test set of MNIST in Fig. 7. Compared to CE-DCNN, the feature representation learned by TDLDA has greater inter-class variation and higher intra-class compactness, and therefore has better discriminability.

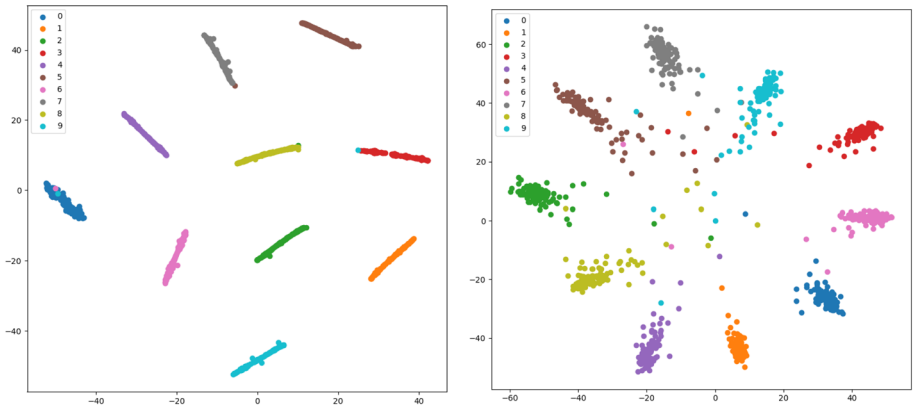
Evaluation on template selection For the recognition of handwritten digits, we used printed digits as templates. However, printed digits also have different fonts, such as SimSun, Arial, Times, and so on. Another scenario is some general image datasets without obvious templates, such as CIFAR-10 and STL-10. We previously used a random image from each class as the template for that class. Since original LDA chooses the class mean as the projection center, we can select the image instance closest to the class mean as the template and compare it with random template. Please note that in this comparison, we used pure templates as the center rather than hybrid centers, i.e., setting the α value to 1, which allows for a better comparison of the effects of different templates. As can be seen from Table 6, there are only minor

Table 4 Test with different backbones on CIFAR-10

Method	Backbone	Params	top-1	top-5
Swin [32]	Swin-T	105.47M	91.17	98.76
Swin+TDLDA		105.45M	92.19	98.85
ResNet [30]	ResNet18	42.70M	91.77	98.44
ResNet+TDLDA		43.81M	92.10	98.67
ResNet [30]	ResNet50	155.96M	95.57	99.16
Resnet+TDLDA		155.97M	95.68	99.52

Table 5 Comparison with DNC [22] on CIFAR-10

Method	Backbone	Params	top-1
ResNet	ResNet50	155.96M	95.57
DNC-ResNet		23.50M	95.78
ResNet+TDCNN		155.96M	95.61
Resnet+TDLDA		155.97M	95.68



(a) TDLDA (b) CE-DCNN

Fig. 7 Illustration of featuer representation in TDCNN, TDLDA and Softmax Loss. (a) TDLDA feature representation, (b) CE-DCNN feature representation

Table 6 Accuracy on different fonts of numbers

	Train Acc(%)	Test Acc(%)
Airial	99.96	99.67
Calibri	99.97	99.65
Times	99.99	99.72
KaiTi	99.92	99.46
SimSun	99.90	99.44

We compare with different templates by using template-centric

Table 7 Accuracy on choice of different templates

	Methods	Train Acc(%)	Test Acc(%)
STL-10	Ran	99.97	81.65
	CM	99.96	81.63
CIFAR-10	Ran	99.98	92.80
	CM	99.97	92.77

[a] Ran means choose random instance from dataset as a template
[b] CM means choose instances closest to the class mean as a template

differences in the recognition results when using different font types as templates, and good recognition results can be obtained. For datasets such as CIFAR-10 and STL-10, randomly selected templates are comparable to those with mean-specified templates in Table 7.

Detection of samples from unseen classes As we know, the linear discriminator, trained using softmax loss and its variations, always assigns a label to any incoming samples. Consequently, softmax loss is ineffective in addressing open-set problems, where incoming samples may belong to unknown classes. However TCRL distinguishes whether a sample belongs to an unseen class. To demonstrate this only digits 0 to 8 from the MNIST dataset were used to train Model B. Then we visualize the feature representation in the training set which is depicted in Fig. 8. As demonstrated in Fig. 8, each class boundary is clearly separated from the boundaries of other classes, making it straightforward to distinguish samples of the digit 9, which was not encountered during training. There is anothe interesting phenomenon, the digit 9 are nearest to the digit 6, 7 and 3, which shows that the network trained by TCRL have learned some intrinsic invariants. All of this implies that TCRL is promising for handling open-set problems.

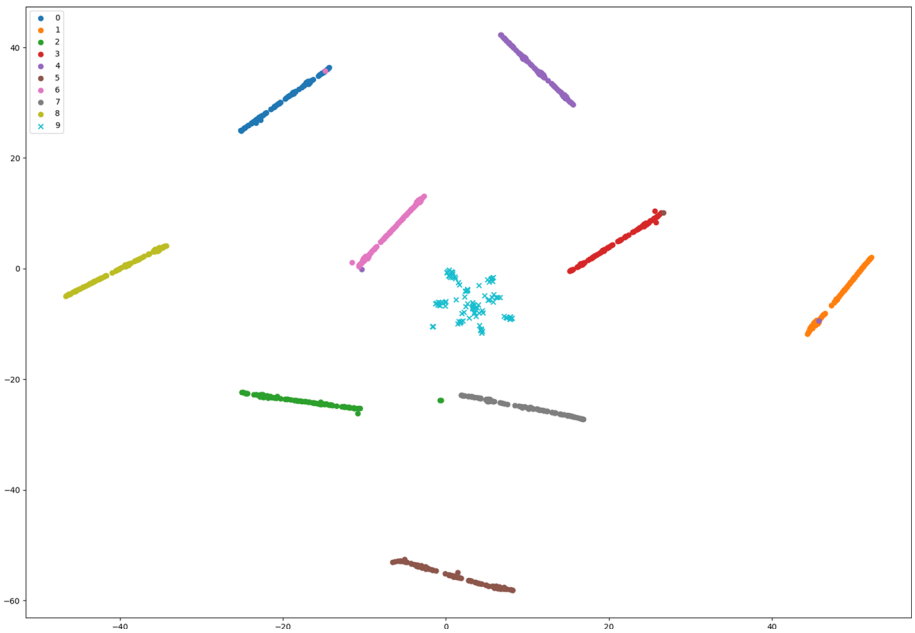


Fig. 8 Feature representation with unseen classes

Fig. 9 Arrow marking dataset from real-world

Real-word application potential In order to explore the potential of TCRL in various real-world tasks, we first apply it to the recognition of actual road arrows. For each of the 11 types of road directional arrows, we collected 20 real images, totaling 220 images to form a test set, as shown in the Fig. 9. Using the model trained on the synthesized arrow marking dataset [26], we test on the actual road arrows, and achieve satisfactory results as presented in the Table 8. This demonstrates the potential for practical applications in the field of road arrow sign recognition.

Table 8 Top-1 classfication accuracy on some real-world datasets

Dataset	Method	Params	top-1
AMD	TDCNN	42.70M	92.70
	TDLDA	42.71M	93.75
YFD citegross2005face	TDCNN	42.69M	99.77
	TDLDA	42.70M	99.80

[a] YFD means Yale Face Dataset
[b] AMD means Arrow Marking Dataset

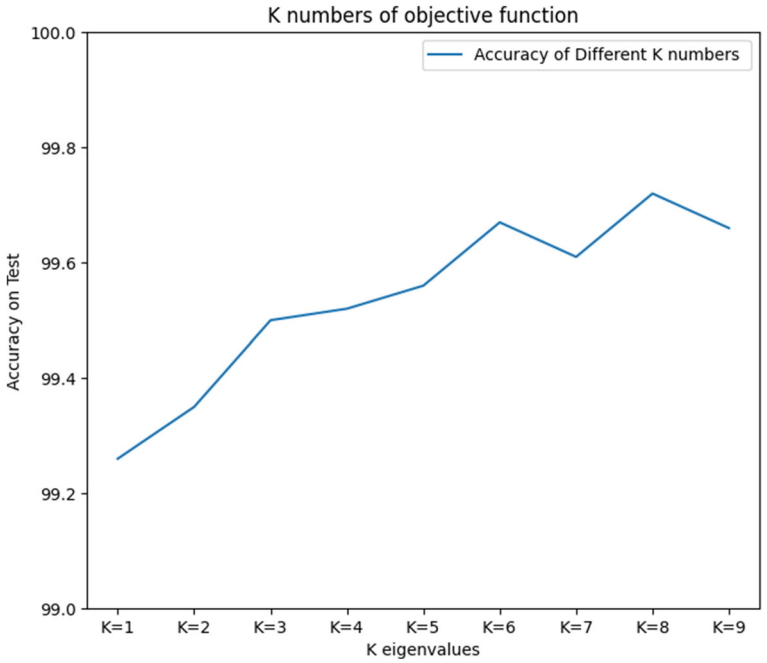


Fig. 10 The performance curve of the model for different values of k

To explore broader application domains, we conducted experiments of facial recognition on the Yale Face Database(YFD) [33]. The YFD contains 165 images from 15 subject, and there are 11 images of each subject with different facial expressions or configurations. The robust results obtained in Table 8 therein underscore the promising prospects for extending our model's applicability into domains such as facial recognition.

Study of hyperparameters TCRL has two hyperparameters ϵ and α, whose values are determined through cross-validation on a validation set in above experiments. We now study the sensitivity of TCRL to these two hyperparameters. The variation of hyperparameter ϵ directly affects the value of k, as shown in the optimization (12). We compared the results under different values of k.As shown in the Fig. 10, it has a significant impact on the performance of the model. In practical applications, we recommend using cross-validation to

Table 9 Choice of hybrid center α on arrow marking dataset

Hybrid Center	Train Acc	Test Acc
α=0.1	99.9	99.36
α=0.3	100	99.76
α=0.5	100	99.76
α=0.7	99.9	99.38
α=0.9	99.28	99.19
M_center	99.91	99.36
T_center	99.34	99.19

[a] M_center means mean center where α=0
[a] T_center means template center where α=1

determine the optimal value of k. As for the hyperparameter α, we tested various possible values in Table 9. The variation of hyperparameter α has little effect on the performance of the model, indicating that TCRL exhibits strong robustness to this hyperparameter.

5 Conclusion

Inspired by the existence of templates in real-word recognition tasks, we propose a template-centric representation learning framework in this paper. The framework aims to learn a feature representation that makes each instance closely centered around its own template and away from other templates. We propose a template-centric representation learning objective function and a template-centric LDA layer, which can be combined with DCNNs learning in an end-to-end manner. Moreover, we explore various ways of constructing templates.

Acknowledgements This work was supported in part by the Fundamental Research Funds for the Central Universities under Grant B230201025, and in part by the Key Research and Development Program of Changzhou (Social Development) under Grant CE20225042.

Data Availability All data included in this study are available upon request by contact with the corresponding author.

Declarations

Conflict of Interest The authors declare that they have no conflict of interest.

References

1. Poggenhans F, Schreiber M, Stiller C (2015) A universal approach to detect and classify road surface markings. In: 2015 IEEE 18th International conference on intelligent transportation systems, pp 1915–1921. https://doi.org/10.1109/ITSC.2015.310
2. Hoang TM, Nam SH, Park KR (2019) Enhanced detection and recognition of road markings based on adaptive region of interest and deep learning. IEEE Access 109817–109832
3. Brunelli R (2009) Template matching techniques in pp 307–318. https://doi.org/10.1002/9780470744055
4. Jurie F, Dhome M et al (2002) Real time robust template matching. In: BMVC, vol. 2002, pp. 123–132
5. Korman S, Reichman D, Tsur G, Avidan S (2013) Fast-match: fast affine template matching. In: Proceedings of the IEEE conference on computer vision and pattern recognition, pp 2331–2338
6. Bengio Y, Courville A, Vincent P (2013) Representation learning: a review and new perspectives. IEEE Trans Pattern Anal Mach Intell 35(8):1798–1828
7. Meng Q, Qian H, Liu Y, Xu Y, Shen Z, Cui L (2023) Unsupervised representation learning for time series: a review. Preprint at arXiv:2308.01578
8. Jing L, Tian Y (2020) Self-supervised visual feature learning with deep neural networks: a survey. IEEE Trans Pattern Anal Mach Intell 43(11):4037–4058
9. Qian H, Pan SJ, Miao C (2021) Weakly-supervised sensor-based activity segmentation and recognition via learning from distributions. Artif Intell 292:103429
10. Kaya M, Bilge HŞ (2019) Deep metric learning: a survey. Symmetry 11(9):1066
11. Wen Y, Zhang K, Li Z, Qiao Y (2016) A discriminative feature learning approach for deep face recognition. In: Computer Vision – ECCV 2016, pp 499–515
12. Liu W, Wen Y, Yu Z, Yang M (2016) Large-margin softmax loss for convolutional neural networks. Preprint arXiv:1612.02295
13. Schroff F, Kalenichenko D, Philbin J (2015) Facenet: a unified embedding for face recognition and clustering. In: Proceedings of the IEEE conference on computer vision and pattern recognition, pp 815–823

14. Weinberger KQ, Saul LK (2009) Distance metric learning for large margin nearest neighbor classification. J Mach Learn Res 10(2)
15. Schroff F, Kalenichenko D, Philbin J (2015) Facenet: a unified embedding for face recognition and clustering. In: 2015 IEEE Conference on computer vision and pattern recognition (CVPR), pp 815–823. https://doi.org/10.1109/CVPR.2015.7298682
16. Dorfer M, Kelz R, Widmer G (2016) Deep linear discriminant analysis. In: Bengio Y, LeCun Y (eds) 4th International Conference on Learning Representations, ICLR 2016, San Juan, Puerto Rico, May 2-4, 2016, Conference Track Proceedings
17. Peng H, Yu S (2021) Beyond softmax loss: Intra-concentration and inter-separability loss for classification. Neurocomputing 438:155–164
18. Bartan B, Pilanci M (2022) Neural fisher discriminant analysis: optimal neural network embeddings in polynomial time. In: International conference on machine learning, pp 1647–1663. PMLR
19. Chang C-C (2023) Fisher's linear discriminant analysis with space-folding operations. IEEE Trans Pattern Anal Mach Intell
20. Yan L, Wang Q, Ma S, Wang J, Yu C (2023) Solve the puzzle of instance segmentation in videos: a weakly supervised framework with spatio-temporal collaboration. IEEE Trans Circuits Syst Video Technol 33(1):393–406. https://doi.org/10.1109/TCSVT.2022.3202574
21. Cao Z, Chu Z, Liu D, Chen Y (2020) A vector-based representation to enhance head pose estimation
22. Wang W, Han C, Zhou T, Liu D (2023) Visual recognition with deep nearest centroids. In: The eleventh international conference on learning representations, ICLR 2023, Kigali, Rwanda, May 1-5, 2023. OpenReview.net, ???
23. Wang W, Liang J, Liu D (2022) Learning equivariant segmentation with instance-unique querying. Adv Neural Inf Process Syst 35:12826–12840
24. Fukunaga K (1990) Introduction to statistical pattern recognition. Academic Press
25. Boroujeni FR, Wang S, Li Z, West N, Stantic B, Yao L, Long G (2018) Trace ratio optimization with feature correlation mining for multiclass discriminant analysis. In: Proceedings of the thirty-second aaai conference on artificial intelligence, New Orleans, Louisiana, USA, February 2-7, pp 2746–2753
26. Wang L, Liu Q (2022) Discriminant distance template matching for image recognition. Mach Vis Appl 33(6):91
27. LeCun Y (1998) The mnist database of handwritten digits. http://yann.lecun.com/exdb/mnist/
28. Krizhevsky A, Hinton G (2010) Convolutional deep belief networks on cifar-10. Unpublished Manuscript 40(7):1–9
29. Coates A, Ng A, Lee H (2011) An analysis of single-layer networks in unsupervised feature learning. In: Proceedings of the fourteenth international conference on artificial intelligence and statistics, pp 215–223. JMLR Workshop and Conference Proceedings
30. He K, Zhang X, Ren S, Sun J (2016) Deep residual learning for image recognition. In: Proceedings of the IEEE conference on computer vision and pattern recognition, pp 770–778
31. Paszke A, Gross S, Massa F, Lerer A, Bradbury J, Chanan G, Killeen T, Lin Z, Gimelshein N, Antiga L et al (2019) Pytorch: An imperative style, high-performance deep learning library. Adv Neural Inf Process Syst 32
32. Liu Z, Lin Y, Cao Y, Hu H, Wei Y, Zhang Z, Lin S, Guo B (2021) Swin transformer: hierarchical vision transformer using shifted windows. In: Proceedings of the IEEE/CVF international conference on computer vision, pp 10012–10022
33. Gross R (2005) Face databases. Handbook of face recognition, 301–327