IET Computer Vision

WILEY

The Institution of Engineering and Technology

ORIGINAL RESEARCH OPEN ACCESS

A Multi-Layer Convolutional Sparse Network for Pattern Classification Based on Sequential Dictionary Learning

Farhad Sadeghi Almalou[1] | Farbod Razzazi[1] | Arash Amini[2]

[1]Department of Mechanical, Electrical and Computer Engineering, SR.C, Islamic Azad University, Tehran, Iran | [2]Department of Electrical Engineering, Sharif University of Technology, Tehran, Iran

Correspondence: Farbod Razzazi (farbod_razzazi@iau.ac.ir)

Received: 16 July 2025 | **Revised:** 25 November 2025 | **Accepted:** 26 December 2025

Handling Editor: Wenguan Wang

Keywords: compressed sensing | convolutional neural nets | dictionaries | feature extraction | image classification | learning (artificial intelligence) | sparse matrices

ABSTRACT

Convolutional sparse coding (CSC) using learnt convolutional dictionaries has recently emerged as an effective technique for emphasising discriminative structures in signal and image processing applications. In this paper, we propose a multilayer model for convolutional sparse networks (CSNs), based on hierarchical convolutional sparse coding and dictionary learning, as a competitive alternative to conventional deep convolutional neural networks (CNNs). In the proposed CSN architecture, each layer learns a convolutional dictionary from the feature maps of the preceding layer (if available), and then uses it to extract sparse representations. This hierarchical process is repeated to obtain high-level feature maps in the final layer, suitable for pattern recognition and classification tasks. One key advantage of the CSN framework is its reduced sensitivity to training set size and its significantly lower computational complexity compared to CNNs. Experimental results on image classification tasks show that the proposed model achieves up to 7% higher accuracy than CNNs when trained with only 150 samples, while reducing computational cost by at least 50% under similar conditions.

1 | Introduction

There is increasing evidence suggesting that human perception operates on a sparse representation of sensory input [1]. Both artificial and natural learning systems face inherent limitations in the quality and quantity of training data they can acquire and process [2, 3]. Consequently, sparse coding (SC) algorithms have received substantial attention in a wide range of image processing tasks, including de-noising, image enhancement and beyond [4–9]. However, the rapid advancement of deep learning—particularly convolutional neural networks (CNNs)—and their remarkable success in computer vision applications have, to some extent, marginalised the role of sparse coding techniques [10, 11].

CNNs have proven highly effective in visual recognition by automating task-specific feature extraction (FE) [12]. Their convolutional layers hierarchically capture increasingly abstract visual features, whereas fully connected layers interpret these features into high-level semantic representations. Over the past decade, architectures, such as LeNet [11], AlexNet [13], VGGNet [14] and GoogleNet [15], have achieved notable success across various benchmark datasets.

Despite these advances, the internal structure and theoretical underpinnings of CNNs remain relatively underexplored. Mallat et al. proposed the scattering transform, which replaces learnt filters with fixed wavelets, demonstrating robustness to geometric transformations such as translation and rotation [16, 17].

Moreover, they showed that jointly learning a dictionary and a classifier can improve classification performance [18]. Other studies have explored the use of deep networks with fully connected layers initialised with independent and identically distributed (i.i.d) random weights, showing that such networks can preserve the metric structure of input data across layers, enabling stable data reconstruction from deep features [19].

Another prominent approach to feature extraction in classification tasks is sparse representation, which has been widely studied in signal and image processing applications [20–23]. In this architecture, a signal can be represented based on a linear combination of a few columns from a matrix called dictionary. That is, the signal can be represented as the product of a dictionary matrix and a sparse coefficient vector. Several algorithms have been proposed to obtain such sparse representations, including L_1-norm optimisation thresholding algorithm [24] and its iterative variant [25].

In traditional implementations, images are often divided into overlapping patches to capture local features for downstream tasks [26]. Convolutional sparse coding (CSC) has emerged as an alternative method for extracting local image representations [27–29]. Although sparse modelling has been explored for over 2 decades, recent theoretical advances have highlighted the role of convolutional operators in enhancing representation capacity.

In 2014, Mairal et al. introduced a layered convolutional sparse coding (CSC) framework inspired by deep neural networks [21]. One of the dictionary training strategies adopted in their work follows the method proposed by Zeiler [30]. In this study, the Lasso equation has been utilised as the cost function in each layer of a two-layer deconvolution network, after which the extracted features have been pooled. Subsequently, these features are unpooled to compute the reconstruction error with respect to the input image. Moreover, the input image channels—or equivalently, each feature map—are independently involved in the sparse coding process. Mairal et al. also explored the relationship between dictionary learning and the backpropagation algorithm commonly used in deep neural networks [21]. They examined the conditions under which such a learning mechanism could have applied in a dictionary-based architecture. However, this formulation results in a highly nonconvex and nonsmooth optimisation problem with respect to the dictionary D. The authors proposed an approximate solution based on the single-layer task-driven dictionary learning (TDDL) approach [31]. This approximation relies on using a large number of samples, employing multiple smoothing approximations to the cost function and consequently reducing the influence of sparsity within the overall optimisation.

Papyan et al. introduced a formal CSC framework [32, 33] and established connections between CSC and CNNs through the concept of multilayer convolutional sparse coding (ML-CSC) [33, 34]. They addressed the inverse problem of signal decomposition using an overcomplete dictionary and sparse representations via the soft thresholding algorithm (STA) [35]. Additionally, they introduced the concept of layered basis pursuit (LBP) and proposed the layered iterative soft thresholding algorithm (LISTA), demonstrating that CNNs can be viewed as approximations of LBP solvers, under specific stability and uniqueness conditions. Sulam et al. introduced ML-CSC pursuit and projection algorithms [36], demonstrating that reconstruction error tends to increase with network depth—a phenomenon that led to the formulation of *stability bands*. In the same study, the authors proposed a multilayer convolutional dictionary learning (ML-CDL) algorithm, which introduces the concept of the *effective dictionary*. The effective dictionary is constructed from the product of dictionaries across all layers. The learning process for this architecture occurs in two main steps. In the first step, similar to single-layer architectures, the final layer sparse codes are obtained by solving a lasso optimisation problem using the initial effective dictionary. In the second step, using the obtained sparse codes, the components of the effective dictionary (i.e., the middle layer dictionaries) are updated. In other words, first, the dictionary of each layer is updated, and subsequently, the sparse codes of the previous layer are generated through it. This process continues sequentially from the end layer towards the first one until convergence is achieved.

Another line of research proposed an architecture based on the effective dictionary concept, structurally similar to the ML-CSC framework, and introduced as multilayer ISTA along with its accelerated variant, multilayer FISTA [34]. This work specifically focuses on the ML-CSC stage within the ML-FISTA and ML-ISTA architectures that utilise trained dictionaries. The study investigated the convergence properties of these iterative algorithms and incorporated a backpropagation-based training procedure for dictionary learning—an approach analogous to filter training in conventional convolutional neural networks.

Recent developments in sparse signal representation have also led to the introduction of ML-SC, a nonconvolutional model proposed in ref. [37], which utilises the Holistic Pursuit (HP) algorithm for sparse code computation. Although this approach relies on a global dictionary, it lacks a defined mechanism for dictionary learning. The study further highlights that layered pursuit strategies are ineffective for accurate sparse code extraction in this framework as reconstruction error tends to increase with network depth. Additionally, the projection pursuit method—being inherently single-layered—fails to leverage the full representational capacity available in multilayer architectures.

A supervised sparse coding model has been proposed that applies dictionary learning to image patches as input [38]. In this framework, input images are initially divided into patches, from which sparse codes are extracted layer by layer. These codes are subsequently passed through an average pooling layer before being forwarded to the next layer. Each dictionary is trained via backpropagation, guided by the loss function of the final classifier. In related work, the K-SVD algorithm has also been employed to learning the dictionaries [39], producing results comparable to earlier methods. Furthermore, several studies have explored supervised feature extraction and classification using deep dictionary learning (DDL) models [40, 41], in which dictionaries across successive layers are optimised by minimising the reconstruction error, typically quantified by the L2-norm of the pursuit equation.

This study proposes a novel architecture based on a multi-layer convolutional sparse network (CSN). In each layer of the CSN, two sequential operations are performed: (1) convolutional dictionary learning and (2) convolutional sparse coding (CSC) for feature extraction. During the dictionary learning stage, the convolutional dictionary of the current layer is learnt using the sparse codes generated by the previous layer. As a result, dictionaries are trained independently at every layer. In the feature extraction stage, sparse codes—also referred to as sparse features—are obtained from the input images using the trained dictionaries and passed to the next layer. This hierarchical mechanism enables the construction of high-level features in the deeper layers, starting from low-level features extracted in the initial layers. The sparse features from the final layer are then used to train a classifier. In summary, the key contributions of this work are as follows.

- We introduce a layer-wise convolutional architecture that jointly integrates convolutional sparse coding (CSC) and dictionary learning (DL) at each layer. Despite its modular design, the proposed model demonstrates performance that is competitive with conventional CNN architectures.
- Although the CSC component of our model shares conceptual similarities with previous studies [32–34, 37], it notably differs in its choice of sparse coding algorithms and the explicit coupling with a dictionary learning mechanism.
- Unlike the approach in ref. [36], which performs sparse coding using a pretrained *effective dictionary* and applies multilayer training only to the dictionary component, our architecture executes both sparse coding and dictionary learning in a multilayered fashion, providing a deeper feature hierarchy.
- Furthermore, the proposed model serves as a flexible platform that facilitates the integration of operations, such as pooling, which were difficult to incorporate in earlier CSC-based frameworks [33].

The remainder of this paper is organised as follows:

Section 2 provides the theoretical background and formalisation of the convolutional sparse coding (CSC) model and its multilayer extension, highlighting the advantages and limitations of the proposed approach. Section 3 presents the architecture of the proposed learning model. Section 4 outlines the experimental setup and implementation details. Section 5 is dedicated to the ablation study, and Section 6 discusses the experimental results. Sections 7 and 8 also present the future works and concluding remarks, respectively.

2 | Multi-Layer Convolutional Sparse Coding and Dictionary Learning Framework

Traditionally, convolutional sparse coding problem to represent a particular signal or image vector $\boldsymbol{s}$ is an optimisation problem called as convolutional basis pursuit denoising (CBPDN):

$$\widehat{\boldsymbol{x}}_{\boldsymbol{m}} = \underset{\{x_m\}}{argmin} \frac{1}{2} \sum_m \|\boldsymbol{d}_{\boldsymbol{m}} * \boldsymbol{x}_{\boldsymbol{m}} - \boldsymbol{s}\|_2^2 + \lambda \sum_m \|\boldsymbol{x}_{\boldsymbol{m}}\|_1 \tag{1}$$

In the above equation, $\{\boldsymbol{d}_m\}$ denotes a set of M dictionary filters (atoms), λ is a regularisation parameter, "$*$" represents convolution notation, $\{\boldsymbol{x}_m\}$ indicates the sparse coefficients or feature maps and $\boldsymbol{s}$ is the input signal (images). By defining D_m in a linear convolution operator, such as $D_m \boldsymbol{x}_m = \boldsymbol{d}_m * \boldsymbol{x}_m$ and defining block matrices and vectors $\mathrm{D} = (D_0 \ D_1 \ \ldots D_m)$ and $\boldsymbol{x} = (\boldsymbol{x_0} \ \boldsymbol{x_1} \ \ldots \boldsymbol{x_m})^T$, Equation (1) can be reformulated into the standard form of the basis pursuit denoising (BP) [42]:

$$\widehat{\boldsymbol{x}} = \underset{\{\boldsymbol{x}\}}{\operatorname{argmin}} \frac{1}{2} \|\mathrm{D}\boldsymbol{x} - \boldsymbol{s}\|_2^2 + \lambda \|\boldsymbol{x}\|_1 \tag{2}$$

where $\boldsymbol{x}$ and $\boldsymbol{s}$ are the sparse representation and input images, respectively, and λ is a regularisation parameter. In the following, for all optimisation equations, the hat notation indicates the value of each variable after solving its corresponding minimisation problem. It can be supposed that D is a convolutional dictionary operator defined in the condition of Equation (2), corresponding to the definition in refs. [33, 36].

The single-layer model described above can be extended to introduce a multilayer model by cascading Equation (2). If the sparse codes of the first layer are considered as the input to the second layer, and this process is repeated up to L layers, a multilayer model is formed, which can be mathematically formulated as follows:

$$\begin{aligned}
\widehat{\boldsymbol{\Gamma}}_{\boldsymbol{1}} &= \underset{\{\Gamma_1\}}{argmin} \frac{1}{2} \|\mathrm{D}_{\boldsymbol{1}}\boldsymbol{\Gamma}_{\boldsymbol{1}} - \boldsymbol{s}\|_2^2 + \lambda_1 \|\boldsymbol{\Gamma}_{\boldsymbol{1}}\|_1, \\
\widehat{\boldsymbol{\Gamma}}_{\boldsymbol{2}} &= \underset{\{\Gamma_2\}}{argmin} \frac{1}{2} \left\| \mathrm{D}_{\boldsymbol{2}}\boldsymbol{\Gamma}_{\boldsymbol{2}} - \widehat{\boldsymbol{\Gamma}}_{\boldsymbol{1}} \right\|_2^2 + \lambda_2 \|\boldsymbol{\Gamma}_{\boldsymbol{2}}\|_1, \\
&\vdots \\
\widehat{\boldsymbol{\Gamma}}_{\boldsymbol{L}} &= \underset{\{\Gamma_L\}}{argmin} \frac{1}{2} \left\| \mathrm{D}_{\boldsymbol{L}}\boldsymbol{\Gamma}_{\boldsymbol{L}} - \widehat{\boldsymbol{\Gamma}}_{\boldsymbol{L-1}} \right\|_2^2 + \lambda_L \|\boldsymbol{\Gamma}_{\boldsymbol{L}}\|_1
\end{aligned} \tag{3}$$

The above structure is called a layered basis pursuit (LBP) model [33]. In the above equation, D_L denotes the layer convolutional dictionary operator defined in Equation (2), $\{\boldsymbol{\Gamma}_L\}$ indicates the layer sparse coefficients or feature maps, $\boldsymbol{s}$ is the input signal (images) and λ_L is the layer regularisation parameter. In the context of recent investigations into the solutions for Equation (3), two predominant methodologies have emerged: the layered iterative soft thresholding algorithms (LISTA) and the layered basis pursuit (LBP) algorithms as referenced in studies [33, 34, 37]. However, this analysis reveals a notable limitation, particularly the tendency for reconstruction error to escalate with increased network depth, which is exacerbated in the presence of noise within images. Additionally, while the focus has predominantly been on ML-CSC methods, there remains a conspicuous absence of discourse surrounding dictionary learning. In the learning phase, filters are trained akin to those of convolutional neural networks and then, with pretrained weights, subsequently employed as dictionaries in ML-CSC configurations. It is imperative to highlight that earlier research employed the CSC models through feed-forward convolutional neural networks, as documented in [32]. Some of the other methodologies often involve processing patched images in a nonconvolutional format, as seen in studies [38, 39], which introduces significant challenges; notably, the independent

generation of image patches at each layer diminishes overall efficiency, while training dictionaries within fully connected networks amplifies computational demands.

Conversely, a second remedy to Equation (3) hinges on a global pursuit strategy, wherein an effective dictionary adeptly decomposes the underlying signal $\boldsymbol{s}$. If the dictionaries of an ML-CSC model with L layers are assumed to be $\boldsymbol{D}_1$, $\boldsymbol{D}_2$, ... , $\boldsymbol{D}_i$, respectively, then the effective dictionary of the mentioned model is defined as $\boldsymbol{D}^{(i)} = \boldsymbol{D}_1\boldsymbol{D}_2...\boldsymbol{D}_i$. Therefore, signal $\boldsymbol{s}$ is decomposed on $\boldsymbol{D}^{(i)}$ as follows:

$$\boldsymbol{s} = \boldsymbol{D}^{(i)}\boldsymbol{\Gamma_i}, 1 \le i \le L \tag{4}$$

In the above equation, $\boldsymbol{s}$ is the input signal (image) and $\boldsymbol{D}^{(i)}$ and $\boldsymbol{\Gamma_i}$ are the *effective dictionary* and sparse codes (coefficients) of the ith layer, respectively. Considering the above definitions in Equation (4) and substituting them into Equation (3), the standard BP pursuit equation can be written for the Lth layer of an ML-CSC model with L layers as follows:

$$\widehat{\boldsymbol{\Gamma}}_{\boldsymbol{L}} = \underset{\Gamma_L}{\arg}\ min\frac{1}{2}\left\| \mathrm{D}^{(L)}\boldsymbol{\Gamma_L} - \widehat{\boldsymbol{\Gamma}}_{\boldsymbol{L-1}} \right\|_2 + \lambda_L\|\boldsymbol{\Gamma_L}\|_1 \tag{5}$$

where D_L denotes effective convolutional dictionary operator defined in the Equation (2). In the second solution, by solving Equation (5), the CSC step is first performed, similar to previous methods for the Lth layer (the last layer), and the sparse coefficients $\boldsymbol{\Gamma_L}$ are calculated [32–34]. Then, using the multilayer convolutional dictionary learning algorithm [36], the dictionary $\boldsymbol{D}^{(L)}$ is updated. These two phases are executed from last to first for $\boldsymbol{\Gamma_{L-1}}$ and $\mathrm{D}^{(L-1)}$, $\boldsymbol{\Gamma_{L-2}}$ and $\mathrm{D}^{(L-2)}$, ... $\boldsymbol{\Gamma_1}$ and $\mathrm{D}^{(1)} = \mathrm{D}_1$, respectively. As mentioned in the previous section, sparse coding is actually performed as a single layer (for the last layer), and only dictionary training is performed in a multilayer regime.

In all the mentioned methods, the ML-CSC model has been presented without implying dictionary training and only in the last work has dictionary training in the ML-CSC model been explicitly stated. In this work, as we said, the CSC stage was implemented as a single-layer model. Inspired by the mentioned method, we propose an ML-CSC architecture that includes both multilayer convolutional dictionary learning (ML-CDL) and ML-CSC in all layers.

3 | Multi-Layer Convolutional Sparse Network

Multilayer convolutional neural networks (CNNs) have demonstrated remarkable performance in recent years, primarily due to their hierarchical feature extraction capabilities, where lower-level features are progressively transformed into more abstract representations in deeper layers. This progression is driven by the transfer and combination of spatial dependencies across layers.

In this study, we present a novel multilayer convolutional architecture based on convolutional sparse coding (CSC), designed to offer a deep learning framework that competes with convolutional neural networks (CNNs) in performance while remaining conceptually distinct. The architecture comprises two key components: dictionary learning (DL) and convolutional sparse coding (CSC), represented as separate modules within each convolutional layer, as illustrated in Figure 1. In the forward pass, the CSC module extracts sparse features from both training and test samples—either directly from input images in the first layer or from the outputs of preceding layers (Figure 1-**top**). During the DL stage, convolutional dictionaries are trained exclusively using the training data. These learnt dictionaries are subsequently employed within the CSC module to enable feature extraction. As shown in Figure 1- **down**, these two stages are executed alternately in each convolutional layer.

To construct the proposed model, we cascade CSC-DL blocks into a multilayer architecture referred to as the convolutional sparse network (CSN), which implements multilayer convolutional sparse coding with dictionary learning (ML-CSC-DL). Additional components, such as pooling layers, can also be integrated as needed to enhance model performance. Figure 2 illustrates a typical CSN architecture designed for classification tasks, comprising convolutional, pooling and fully connected (FC) layers. In one branch, the dictionaries within the CSC-DL blocks are trained using the training samples. In the parallel branch, the CSC blocks extract features from these learnt dictionaries. Both training and testing features are then passed through the FC layer and fed into the classifier for final prediction.

Although earlier models—such as those proposed in refs. [34, 37, 43]—demonstrate notable strengths, their methodology of extracting sparse vectors independently at each layer restricts interlayer information flow, thereby hindering the formation of high-level semantic representations. To overcome this limitation, we proposed a strategy that distributes sparsity information across layers by tuning the regularisation coefficients. In doing so, we generated sparse representations not only in the spatial domain but also along the channel dimension, thereby capturing image correlations in all three dimensions—width, height and depth. This unified representation enables both feature maps and channels to share structural information across layers, resulting in a consolidated output feature map. To support this framework, we employed 3D filters in the dictionary learning algorithm, where an additional dimension was introduced to represent the number of filters. In the following text, to avoid using multiple indices and to simplify the understanding of mathematical equations, it has been assumed

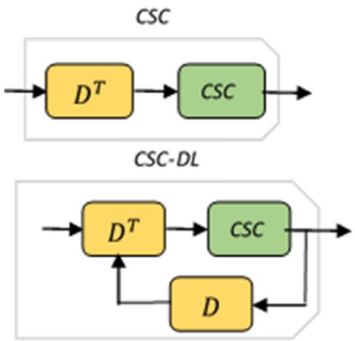

FIGURE 1 | Convolutional unit. Top: CSC block. Down: CSC-DL block.

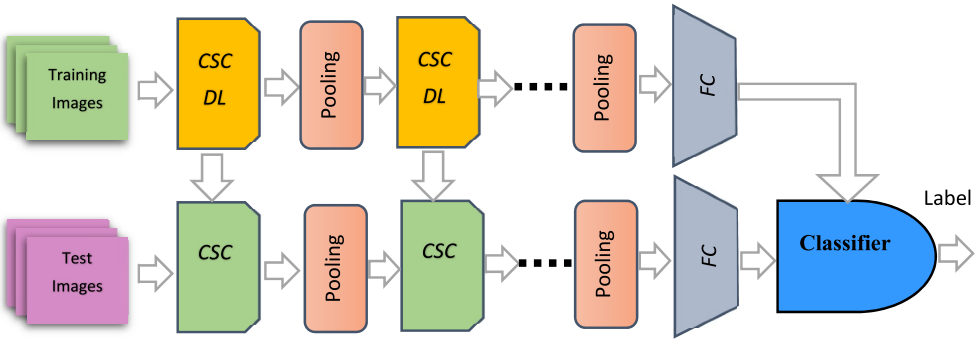

FIGURE 2 | The CSN model for classification task.

that all equations are written in the Lth layer and $\lambda_L = \lambda$. Consequently, the basis pursuit (BP) problem for both the CSC and CDL components can be reformulated as follows:

$$\arg\min_{\{x_m\},\{d_{m,c}\}} \frac{1}{2}\sum_{c,k}\left\|\sum_m \boldsymbol{d}_{c,m} * \boldsymbol{x}_{m,k} - \boldsymbol{s}_{c,k}\right\|_2^2 + \lambda\sum_{m,k}\left\|\boldsymbol{x}_{m,k}\right\|_1 \tag{6}$$

where k, m and c are the number of samples, filters and channels in each layer, respectively. In each layer, $\boldsymbol{s}$ denotes the input feature map and $\boldsymbol{x}$ is the out put sparse feature map. In later layers, the number of image channels equals the number of filters from the previous layer, determining the number of maps in those layers. Thus, the number of maps in each layer depends solely on the preceding layer's filters, a benefit of our model. Based on the above definitions, Equation (6) can be rewritten for an image in each layer, and the use of image batches can be deferred to practical implementations. Additionally, by defining three-dimensional dictionaries with c channels per layer ($\boldsymbol{d}_{c,m} = \boldsymbol{d}_m$), Equation (6) can be framed as a simple convolutional basis pursuit (CBP) problem for a single image.

$$\arg\min_{\{x_m\},\{d_m\}} \frac{1}{2}\left\|\sum_m \boldsymbol{d}_m * \boldsymbol{x}_m - \boldsymbol{s}\right\|_2^2 + \lambda\sum_m \|\boldsymbol{x}_m\|_1 \tag{7}$$

where in each layer, $\boldsymbol{s}$ denotes the input feature map with c channels (the multichannel image or feature map), $\boldsymbol{d}_m$ is the mth filter from the layer dictionary with c channels and $\boldsymbol{x}_m$ is the mth output sparse feature map.

One of the effective methods in solving Equation (7) is the alternating direction method of multipliers (ADMM) [44]. Equation (7) can be divided into two problems, CSC and DL, which can be integrated with ADMM. This approach is performed by alternating between a sparse coding step that updates the sparse representation of the training data $\boldsymbol{x}_m$ given the current dictionary and a dictionary updating step that updates the current dictionary $\boldsymbol{d}_m$ given the new sparse representation (Figure 1).

Convolutional dictionary learning: We can apply ADMM to Equation (7) by variable splitting, introducing an auxiliary variable $\boldsymbol{g}_m$ that is constrained to be equal to the primary variable $\boldsymbol{d}_m$, and leading to the equivalent problem:

$$\operatorname{argmin}_{\{d_m\},\{g_m\}} \frac{1}{2}\left\|\sum_m \boldsymbol{d}_m * \boldsymbol{x}_m - \boldsymbol{s}\right\|_2^2 + \sum_m \iota C_{PN}(\boldsymbol{g}_m)\ s.t.\quad \boldsymbol{d}_m - \boldsymbol{g}_m = 0\ \forall\, m \tag{8}$$

where ιC_{PN} is the indicator function of the constraint set C_{PN}, which causes Equation (8) to be solved in the unconstrained state [45]. The corresponding ADMM iterations are as follows:

$$\begin{aligned}\{\boldsymbol{d}_m\}^{(j+1)} = \underset{\{d_m\}}{argmin}\ &\frac{1}{2}\left\|\sum_m \boldsymbol{d}_m * \boldsymbol{x}_m - \boldsymbol{s}\right\|_2^2 \\ &+ \frac{\sigma}{2}\left\|\sum_m \boldsymbol{d}_m - \boldsymbol{g}_m{}^{(j)} + \boldsymbol{h}_m{}^{(j)}\right\|_2^2\end{aligned} \tag{9}$$

$$\begin{aligned}\{\boldsymbol{g}_m\}^{(j+1)} = \underset{\{g_m\}}{\operatorname{argmin}}\ &\frac{1}{2}\sum_m \iota C_{PN}(\boldsymbol{g}_m) \\ &+ \frac{\sigma}{2}\left\|\sum_m \boldsymbol{d}_m{}^{(j+1)} - \boldsymbol{g}_m + \boldsymbol{h}_m{}^{(j)}\right\|_2^2\end{aligned} \tag{10}$$

$$\boldsymbol{h}_m{}^{(j+1)} = \boldsymbol{h}_m{}^{(j)} + \boldsymbol{d}_m{}^{(j+1)} - \boldsymbol{g}_m{}^{(j+1)} \tag{11}$$

Here, h_m is an auxiliary ADMM variable introduced to decouple the dictionary-update constraint, acting as the consensus variable for d_m during optimisation. The step of Equation (11) involves simple arithmetic calculations, and the solution of Equation (10) can be formulated in closed-form [44]. The computationally expensive step of the algorithm is Equation (9), which can be solved very efficiently by exploiting the Sherman–Morrison formula in the DFT domain [46].

Convolutional sparse coding: By rewriting Equation (2) in suitable form for ADMM by introducing auxiliary variables $\boldsymbol{y}_m$, we have:

$$\operatorname{argmin}_{\{x_m\},\{y_m\}} \frac{1}{2}\left\|\sum_m \boldsymbol{d}_m * \boldsymbol{x}_m - \boldsymbol{s}\right\|_2^2 + \lambda\sum_m \left\|\boldsymbol{y}_m\right\|_1\ ;s.t.\quad \boldsymbol{x}_m - \boldsymbol{y}_m = 0 \tag{12}$$

for which the corresponding iterations, with dual variables $\boldsymbol{u}_m$, are as follows:

$$\{\boldsymbol{x_m}\}^{(j+1)} = \underset{\{x_m\}}{\operatorname{argmin}} \frac{1}{2}\left\|\sum_m \boldsymbol{d_m} * \boldsymbol{x_m} - \boldsymbol{s}\right\|_2^2 + \frac{\rho}{2}\left\|\sum_m \boldsymbol{x_m} - \boldsymbol{y_m}^{(j)} + \boldsymbol{u_m}^{(j)}\right\|_2^2 \tag{13}$$

$$\{\boldsymbol{y_m}\}^{(j+1)} = \underset{\{y_m\}}{\operatorname{argmin}} \lambda \sum_m \|\boldsymbol{y_m}\|_1 + \frac{\rho}{2}\left\|\sum_m \boldsymbol{x_m}^{(j+1)} - \boldsymbol{y_m} + \boldsymbol{u_m}^{(j)}\right\|_2^2 \tag{14}$$

$$\boldsymbol{u_m}^{(j+1)} = \boldsymbol{u_m}^{(j)} + \boldsymbol{x_m}^{(j+1)} - \boldsymbol{y_m}^{(j+1)} \tag{15}$$

Where u_m denotes the ADMM dual/consensus variable associated with the sparse coding constraint, enforcing consistency between the convolutional reconstruction and the updated sparse code. The subproblem Equation (14) can be solved via shrinkage/soft thresholding algorithm which involves a closed-form solution [47]. The only computationally expensive step is solving Equation (13), which again can be solved by exploiting the Sherman–Morrison formula in the DFT domain.

We couple the alternating optimisation of the two subproblems through auxiliary variables. Using ADMM in the DFT domain to solve Equations (8) and (12) yields a more stable method with fast convergence, eliminating the need for multiple iterations. Thus, a single iteration updating Equations (9)–(11) and (13)–(15) suffices.

The CDL and CSC algorithms were implemented as sublayers within TensorFlow and Keras, allowing their integration with CNN layers in Keras. Additionally, CSC layers enable the integration of pretrained models such as ResNET and DensNET (not covered in this article).

Figure 3 shows the sparse representation of feature maps from the first two layers of this model.

4 | Experiments and Results

The objective of this study is to evaluate the performance accuracy of the CSN model and also to investigate the effect of the proposed CSN architecture on classification accuracy using classical datasets. We first implement a multilayered feature extraction (FE) model for classification with CSN architecture to evaluate the performance of the proposed architecture. Then, in erosion studies, we analyse the role of some internal and external components in the proposed model.

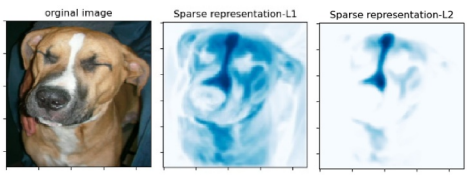

FIGURE 3 | Sparse representation of the first and second convolution layer. **Left:** origenal image. **Middle:** first layer representation. **Right:** second layer representation.

4.1 | Experimental Setup

4.1.1 | Datasets

We used the standard datasets of MNIST [11], SVHN [48], CFAR10 [49], FMNIST [50] and Cats versus Dogs [51] to demonstrate the effectiveness of the proposed model and compare its performance with a comparable CNN model, details of which are given in Table 1. All the mentioned datasets have image dimensions of 32 × 32. To evaluate the proposed model with larger images, the Cats versus Dogs dataset was selected, and their images were used in the baseline models, resized to 224 × 224 and 112 × 112.

4.1.2 | Implementation Detail

To evaluate the CSN architecture on classical datasets, we developed a baseline feature extraction model, termed CSN-b, composed of three-layer CSC-DL with dictionary dimensions of (3 × 3 × 3 × 16), (3 × 3 × 16 × 32) and (3 × 3 × 32 × 32), respectively. The filter sizes in all layers are 3 × 3, and the channel numbers are 3, 16 and 32, respectively. Figure 4 shows the architectures using images from the MNIST dataset. Both models have three convolution and two maxpooling layers. As shown in Figure 4, three CSC-DL blocks (class layer) are used with 16, 32 and 32 filters in layers 1, 3 and 5, respectively. Also, two pooling layers are applied in layers 2 and 4. For comparative analysis, a conventional baseline CNN (b-CNN) was constructed using the same architectural configuration and filter specifications, ensuring consistency in the processing pipeline. The

TABLE 1 | Classical datasets.

Datasets	# Class	# Train	# Test
MNIST	10	50,000	10,000
CFAR10	10	50,000	10,000
SVSN	10	50,000	10,000
FMNIST	10	50,000	10,000
Dogs versus Cats	2	12,500	12,500

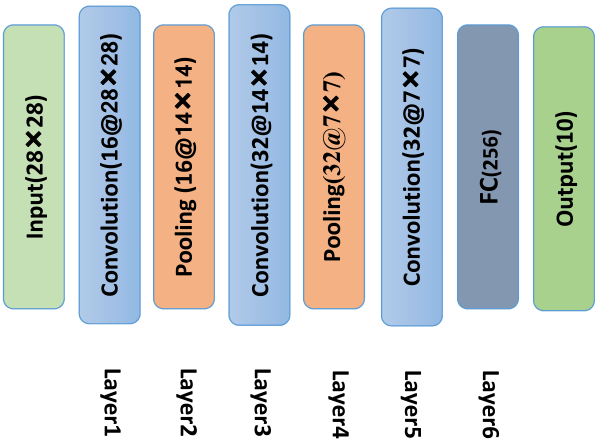

FIGURE 4 | Baseline architecture: CSN-b and CNN-b.

output feature maps of these models are flattened and then fed into the classifier network, which consists of FC (layer 6) and softmax layers, to determine the labels. The regularising coefficients λ in the CSN-b model have been adjusted to the most suitable values in the convolutional layers, with the values for each layer separately detailed for the dictionary learning (DL) and feature extraction (sparse coding) stages in Table 2. The classification epoch of 200 and batch size of 10 were used for both baseline models to ensure fairness, and assessments took place on a system equipped with an i7 CPU and 32 GB of RAM.

4.1.3 | Assessment Metrics

To evaluate and compare the results—particularly in the ablation studies—three sparsity-related metrics were also employed.

Hoyer Index: The first metric, known as the Hoyer index, is defined as follows [52]:

$$H(\boldsymbol{x}) = \frac{1}{1-\sqrt{n}}\left(\sqrt{n} - \frac{\sum |x_i|}{\sqrt{(x_i)^2}}\right) \tag{16}$$

where $\sum_{i=0}^{n} |x_i| = \|x\|_1$ and $\sqrt{(x_i)^2} = \|x\|_2$. This index equal to one if and only if x contains exactly one nonzero element (i.e., it is perfectly sparse) and equals 0 if and only if all elements of x are identical (i.e., completely dense). Therefore, $H(x)$ takes values in the range [0, 1].

Mean Absolute Activation (MA): This metric, which corresponds to the mean of the L_1-norm of vector x, is defined as follows:

$$MA = \frac{1}{n}\sum_{i=0}^{n} |x_i| \tag{17}$$

where n denotes the dimensionality of $\boldsymbol{x}$. Unlike the Hoyer index—which is based on the ratio of the L_1 and L_2 norms and is therefore insensitive to the scale of the samples—the MA metric reflects the average activation intensity of $\boldsymbol{x}$. It provides an estimate of the typical amplitude of the nonzero entries.

Nonzero Ratio: In our experiments, we also report the ratio of nonzero entries to the total number of samples, defined using a threshold level τ, which corresponds to the L_0 -norm behaviour of vector $\boldsymbol{x}$.

For the baseline model, analysis of the data percentiles showed that 50% of the activations (the median) had amplitudes below 0.0045; therefore, we selected a threshold of $\tau = 10{-}3$ for this model.

In the other models, approximately 50% of the activations exhibited amplitudes below about 0.35, and consequently, for the models, including a normalisation layer, the threshold was set to $\tau = 10{-}1$.

4.2 | Computation Complexity

To evaluate computational complexity, both models were trained on 30,000 training samples and tested on 10,000 samples. We measured the time taken for dictionary learning (DL) and feature extraction (FE) in each baseline model. It is important to note that classification time was excluded from this comparison. As summarised in Table 3, the proposed model demonstrated superior performance in computational efficiency, requiring less time for both training and feature extraction compared to the competing approach.

4.3 | The Performance of Proposed Architecture

We evaluated the model's performance by classifying the mentioned datasets using the CSN-b and CNN-b models based on the architecture in Figure 4. Training samples were set to 30,000 and testing samples to 10,000, except for the Dogs versus Cats dataset, which had 5000 and 3000 samples, respectively. Other parameters are in Table 2 and the results are in Table 3. As can be observed, the CSN-b model has comparable accuracy to that of the CNN-b model with larger sample sizes.

TABLE 3 | Comparison of accuracy and time consumption.

	CNN-b		CSN-b	
Datasets	**Acc. (%)**	**Time (s)**	**Acc. (%)**	**Time (s)**
MNIST	99.27	900	98.5	495
SVHS	86.30	1950	81.3	755
FMNIST	88	1200	89.8	495
CFAR10	60.33	2100	58	755
Cat versus Dogs	76	9540	71	1630

TABLE 2 | The regularising coefficients λ in the CSN-b.

	DL			FE (sparse coding)		
Datasets	L_1	L_3	L_5	L_1	L_3	L_5
MNIST	1×10^{-1}	5×10^{-2}	1×10^{-2}	1×10^{-2}	1×10^{-4}	1×10^{-4}
SVHN	35×10^{-2}	5×10^{-2}	1×10^{-2}	1×10^{-1}	5×10^{-4}	1×10^{-4}
FMNIST	1×10^{-3}	5×10^{-4}	1×10^{-4}	1×10^{-4}	5×10^{-5}	1×10^{-4}
CFAR10	5×10^{-2}	15×10^{-4}	5×10^{-4}	1×10^{-2}	1×10^{-4}	1×10^{-4}
Dogs versus Cats	1×10^{-2}	1×10^{-2}	1×10^{-3}	1×10^{-2}	1×10^{-3}	1×10^{-5}

4.4 | The Effect of Training Samples

We evaluated the CSN model variations using the MNIST dataset, fixing the regularising coefficients, filter count and kernel size according to Table 2 and baselines. The test sample size has half the training samples, and classification accuracy has been computed for both baseline models. Figure 5 displays the accuracy graph relative to sample numbers.

The proposed model outperforms the baseline model with fewer training samples. It was validated using other datasets with 150 training and 100 test samples to demonstrate the CSN model's efficiency. Results are shown in Table 4 and Figure 6.

4.5 | Regularising Coefficient Effect

We selected 2000 training samples and 1000 test samples from the MNIST dataset to assess the impact of the λ coefficient in the convolution layers as illustrated in Figure 4 and Table 2. The classification accuracy for various λ values in the final layer was calculated while keeping λ constant in the first two layers during dictionary learning and feature extraction. This experiment was repeated three times with different λ values in the first two layers, and the results are presented in Figure 7.

On the other hand, during feature extraction on test samples, λ was held constant in two layers while varying it in one. The classification accuracy was then calculated as shown in Figure 8. Results indicate that λ significantly impacts the first layer and should remain small in subsequent layers.

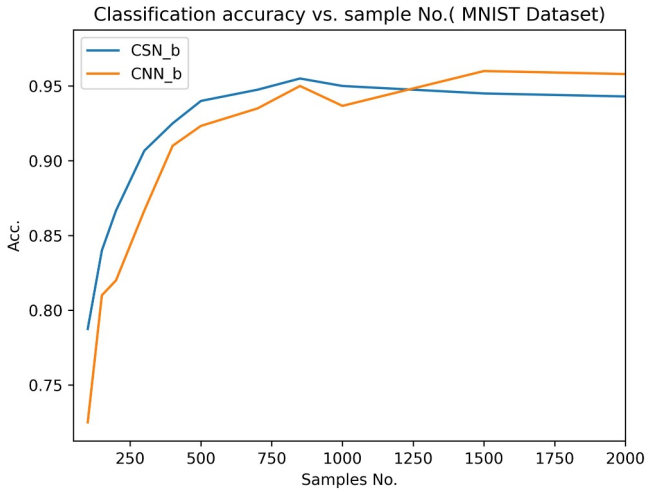

FIGURE 5 | Accuracy comparison against number of samples in MNIST datasets.

TABLE 4 | Accuracy comparison with 150 training samples.

Datasets	CNN-b Acc. (%)	CSN-b Acc. (%)
MNIST	74	68
SVHS	30	36
FMNIST	73	80
CFAR10	21	29

4.6 | Comparison of the CSN With Standard Deep Neural Networks

To compare the proposed CSN with standard deep neural networks, we considered a baseline model with four blocks consisting of convolutional and max-pooling layers. The filter size in the baseline model was set to 3×3, and the number of filters in each layer was selected as 16, 32, 64 and 128, respectively. For the second model, we used ResNet-50 pretrained on ImageNet. In this model, the final layer was unfrozen to allow fine-tuning.

For evaluation, we employed two datasets: CIFAR-10 and Dogs versus Cats. The Dogs versus Cats dataset, as a well-known benchmark for binary classification of pet images, is particularly suitable due to its apparent simplicity and the significant visual similarity between the two classes. Moreover, unlike

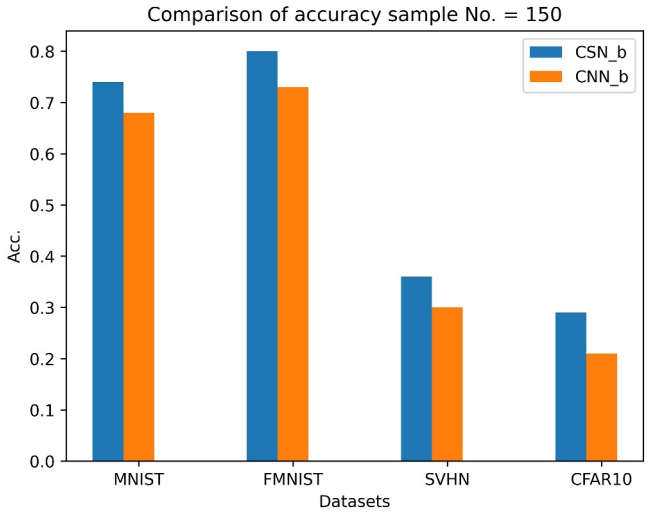

FIGURE 6 | Accuracy comparison with 150 training samples in different datasets.

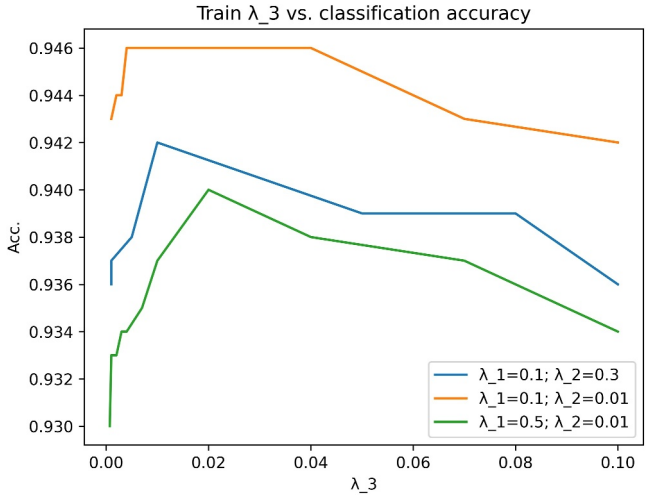

FIGURE 7 | Comparison of Classification accuracy against the regulariser coefficients: DL stage.

CIFAR-10, the image resolution in this dataset is considerably larger, which allows us to assess the proposed model from this perspective as well. We evaluated both models on these two datasets using limited training samples of 150, 500 and 1000 images for image classification. The results of these experiments appear in Table 5.

In these experiments, the images of both datasets were resized to 112 × 112 for the ResNet-50 model to ensure compatibility. In the proposed CSN model, CIFAR-10 images were fed to the network without any resizing, while in the Dogs versus Cats dataset, images were resized to 112 × 112 to evaluate the model on larger image resolutions.

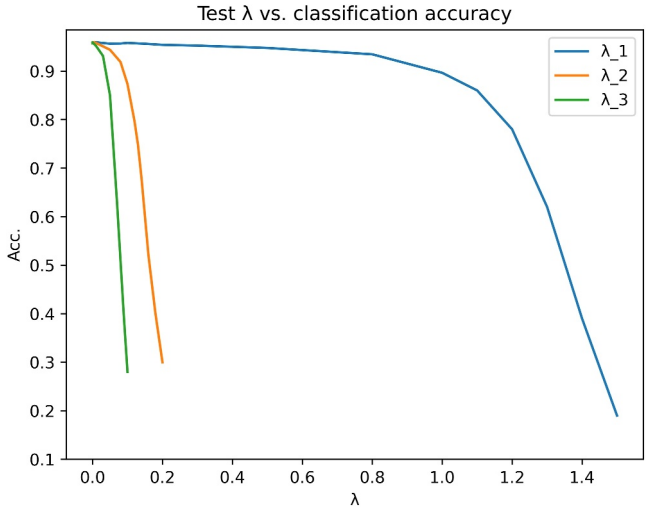

FIGURE 8 | Comparison of classification accuracy against the regulariser coefficients: FE stage.

TABLE 5 | Comparison of accuracy: CSN versus ResNet-50.

Datasets	Models	Train/ Test NO.	Acc.- 1 top %
CFAR10	CSN	150/70	27.14 ± 3.38
		500/250	36.56 ± 2.18
		1000/500	39.96 ± 1,69
	ResNet50	150/70	29.42 ± 2.14
		500/250	37.60 ± 3.18
		1000/500	38.32 ± 2.08
Dogs versus Cats	CSN	150/70	61.96 ± 5.07
		500/250	62.32 ± 1.88
		1000/500	62.72 ± 2.87
	ResNet50	150/70	55.13 ± 4.65
		500/250	62.40 ± 3.10
		1000/500	65.93 ± 1.15

As shown in Table 5, the proposed model performs favourably when the number of training samples is limited. As the number of samples increases, its performance becomes slightly lower than that of ResNet-50. It is important to note that ResNet-50 contains 23,587,712 parameters and is pre-trained on ImageNet, whereas the proposed CSN model uses only 60,336 parameters in its feature-extraction layers. These models are hence not directly comparable in scale. Therefore, the results of the proposed CSN are considered competitive with respect to its parameter count and model complexity.

5 | Ablation Studies

For the ablation studies, a CSN-b model with four blocks was considered. The first three blocks were identical to the baseline model used in Section 4, and an additional fourth block was appended, consisting of a convolutional layer with 64 filters followed by a pooling layer. The regularisation coefficients for the convolutional layers were set to 0.1, 0.01, 0.001 and 0.001 for the first through fourth layers, respectively. These coefficients were used consistently in both the DL and SC stages. All other settings and hyperparameters of the baseline model were kept the same as in the previous configuration. The initial experiments were conducted on the end layer of the baseline model using the aforementioned sparsity metrics and training sets of 500 and 5000 samples on the CFAR10 dataset. The results are reported in Table 6. In these experiments, the sparsity level is approximately 28%, whereas the activation magnitudes remain very weak.

5.1 | Normalisation Effect

In the baseline model, a normalisation module was incorporated after each convolutional layer in every block. A normalisation layer was used to normalise each sample (i.e., each feature map) independently of the others. This operation restored the average activation magnitude, which had been significantly weakened after the SC step. The results of this experiment are presented in Table 7.

It should be noted that in this experiment—and in all subsequent experiments—the sparsity regularisation coefficient was set to $\lambda = 0.2$ for all layers. Table 7 also reports the results for the case in which the last block does not include a normalisation layer (indicated with an "*"). In the remaining configurations, all blocks include normalisation. As shown, although the sparsity level remains approximately 25%–30% across all cases, the activation magnitudes are severely diminished when only the final block lacks normalisation.

TABLE 6 | Classification accuracy and sparsity measurements-baseline.

Train/test no.	NR	MA	H	Acc. (%)
500/200	0.8200 ± 0.0208	0.0069 ± 0.0007	0.2850 ± 0.0098	36.00 ± 0.84
5000/2000	0.8187 ± 0.0220	0.0068 ± 0.0011	0.2823 ± 0.0046	44.62 ± 0.74

By examining the results, it becomes clear that despite a slight reduction in accuracy, the addition of normalisation layers yields an important benefit. One of the key challenges in the CSN model was the severe attenuation of activation magnitudes as the number of layers increased, which led to vanishing effects in deeper networks. The above experiments demonstrate that this issue is resolved by adding normalisation layers, thereby enabling the construction of much deeper CSN architectures—an important achievement.

5.2 | Convolution Sparse Coding Without Dictionary Learning

To demonstrate the importance of dictionary learning, we conducted an experiment in which feature extraction was performed solely using CSC operations with randomly initialised dictionaries in all four blocks. Features were extracted from 5000 train samples and 2000 test samples and then fed to the classifier. This experiment used the same settings as Experiment 5-1, with the only difference being that the DL stage was eliminated. The results showed that when dictionary training is not performed, the classification accuracy drops from $42.68 \pm 0.95\%$ to $19.91 \pm 0.30\%$.

5.3 | Network Depth

To evaluate the impact of network depth, we first removed the third block of the baseline model and then added a second instance of the same block to construct three-block (16, 32 and 64 - filters) and five-block (16, 32, 32, 32 and 64 - filters) CSN models. Using 500 and 5000 training samples, both models performed CSC-DL and forwarded the extracted features to the classifier. The results of these experiments are presented in Tables 8 and 9. To further assess the effect of the sparsity coefficient, each experiment was repeated with $\lambda = 0.2$ and $\lambda = 0.25$.

The number of trainable parameters in the three-block model is 56,464, whereas the five-block model contains 75,024 parameters. Examination of the results in these tables—compared with Table 7—shows that reducing the network depth to three blocks does not significantly affect classification accuracy, especially when the number of training samples is large. Conversely, increasing the depth to five blocks does not improve accuracy; in fact, accuracy slightly decreases even with more training samples. These experiments demonstrate that the CSN model behaves similarly to CNNs, where increasing depth does not necessarily lead to higher accuracy.

5.4 | Convergence of CSC Algorithms

As demonstrated throughout this paper, by leveraging convolutional sparse coding-based optimisation principles, we introduced an interpretable and analytically grounded architecture that functions similarly to CNNs. Earlier sections examined several key aspects of the CSN model. In this experiment, we specifically evaluate the effectiveness of the ADMM algorithm by replacing it with FISTA [53] within the CSN framework.

To perform this comparison, we first trained the model dictionaries using ADMM, following the same procedure as in previous experiments. Then, for feature extraction on both the training and test samples, we applied ADMM and FISTA separately and fed the resulting features into the classifier. To assess convergence behaviour, both experiments were run with 50 iterations, and the convergence of each optimisation algorithm was measured. The results are presented in Figure 9.

In this experiment, the classification accuracy achieved using FISTA was approximately 32% with 500 training samples—about 3% lower than the accuracy obtained using ADMM. More importantly, as illustrated in Figure 9, ADMM exhibits superior convergence behaviour, reaching the required convergence level

TABLE 7 | Classification accuracy and sparsity measurements with normalisation.

Train/test no.	NR	MA	H	Acc. (%)
500/200*	0.3940 ± 0.0049	0.0932 ± 0.0010	0.2503 ± 0.0023	34.10 ± 1.58
500/200	0.6662 ± 0.0112	0.6811 ± 0.0096	0.3150 ± 0.0037	35.10 ± 0.91
5000/2000	0.6629 ± 0.0073	0.6933 ± 0.0071	0.3071 ± 0.0032	42.68 ± 0.95

TABLE 8 | Classification accuracy and sparsity measurements, 3-layer CSN.

Train/test no.	NR	MA	H	Acc. (%)
500/250 $\lambda = 0.2$	0.6546 ± 0.0140	0.6297 ± 0.0339	0.3305 ± 0.0027	34.40 ± 0.97
5000/2500 $\lambda = 0.2$	0.6601 ± 0.0128	0.6268 ± 0.0522	0.3212 ± 0.0050	42.41 ± 0.13
500/250 $\lambda = 0.25$	0.7375 ± 0.0260	0.6225 ± 0.0192	0.3025 ± 0.0082	32.50 ± 1.39
5000/2500 $\lambda = 0.25$	0.7110 ± 0.0323	0.6147 ± 0.0284	0.3161 ± 0.0129	41.60 ± 1.44

within only a few iterations. It demonstrates the effectiveness of using ADMM within the proposed CSN framework. Based on the results of this experiment, the convergence of ADMM in this setting is well guaranteed.

6 | Discussion

Despite differences in the number of layers and filters compared to recent models, our proposed model outperforms CNNs in many respects. As shown in Table 3, CSN-b delivers acceptable results across numerous samples. The sparsity of our cost function reduces data redundancy, enhancing discriminative capacity and lowering complexity, particularly with challenging datasets. Although achieving comparable accuracy, the proposed model significantly surpasses the competing model in complexity metrics as highlighted in Table 3.

The CSN model demonstrates exceptional accuracy with the limited sample size, highlighting its discriminative capacity. As shown in Figure 5, the CSN-b model outperforms the baseline CNN model with fewer than 1000 samples from the MNIST dataset. This advantage continues in other datasets; Table 4 and Figure 6 illustrate that with under 150 samples, the CSN-b's classification accuracy significantly surpasses that of the competition.

Sparsity benefits the CSN model but poses challenges for CNN architectures. High-level feature generation, influenced by deep structures, is impaired when sparsity affects initial layers, disrupting data integration in later layers. To mitigate this issue, our model evenly distributes sparsity across all layers using regulariser coefficients.

Our experiments indicated that an optimal λ_1 in the first layer results in λ_2 falling within the range of $0.1\lambda_1 \leq \lambda_2 \leq 0.3\lambda_1$. Similarly, for the third layer, λ_3 should be selected to satisfy $0.1\lambda_2 \leq \lambda_3 \leq 0.3\lambda_2$, a ratio that applies to subsequent layers, though fine-tuning may be required. We initially set $\lambda_1 = 0.1$ and $\lambda_2 = 0.3$, then calculated classification accuracy for various λ_3. In another test with $\lambda_2 = 0.1$, we obtained better results as shown in Figure 7. However, setting $\lambda_1 = 0.5$ although keeping λ_2 led to poorer outcomes. Therefore, optimising λ_1 in the first layer is critical and should employ a greedy search.

The regulariser coefficient in the first layer is more important than in later layers as most weights during test feature extraction concentrate on the reconstruction term $\boldsymbol{X} = \boldsymbol{D\Gamma}$, which has a closed-form solution for $\boldsymbol{\Gamma}$. As shown in Figure 8, λ_{1-test} should range from 0.1 to 0.01, aligning with $\lambda_{1-train}$, whereas λ in subsequent layers should be kept minimal.

On the other hand, conducting the ablation experiments revealed new dimensions of this challenge. The evaluation results in Section 5-1 indicate that the observed challenge, as well as the sparsity behaviour in CSN, was not primarily related to the number of zero-valued elements but rather to the attenuation of nonzero activation magnitudes. As the network depth increases, this attenuation leads to data vanishing and consequently reduces the model's performance.

The above experiments demonstrate that this problem is resolved by adding normalisation layers, thereby enabling the construction of significantly deeper CSN architectures—an important achievement. Although data normalisation in the CSN mode, applied after the CSC operation, does not substantially increase the intensity of sparsity, it prevents the reduction of activation magnitudes. Therefore, it was observed that with normalisation layers, there is no need to decrease λ across the layers.

Another significant point is that experiments demonstrate the effectiveness of using ADMM within the proposed CSN framework. ADMM exhibits superior convergence behaviour, reaching the required convergence level within only a few iterations. Due to the mini-batch processing used in the proposed model (with batch sizes of 10), ADMM can achieve sufficient

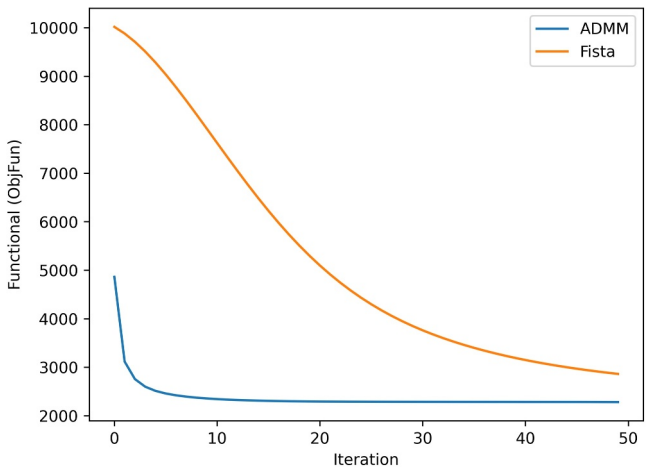

FIGURE 9 | Convergence of ADMM and FISTA in the CSC stage.

TABLE 9 | Classification accuracy and sparsity measurements, 5-layer CSN.

Train/test no.	NR	MA	H	Acc. (%)
500/250 $\lambda = 0.2$	0.6755 ± 0.0141	0.7003 ± 0.0219	0.3055 ± 0.0111	35.40 ± 1.28
5000/2500 $\lambda = 0.2$	0.6818 ± 0.0199	0.6960 ± 0.0253	0.3067 ± 0.0104	40.69 ± 0.28
500/250 $\lambda = 0.25$	0.7718 ± 0.0091	0.7233 ± 0.0212	0.2839 ± 0.0066	32.00 ± 1.16
5000/2500 $\lambda = 0.25$	0.7729 ± 0.0089	0.7314 ± 0.0049	0.2822 ± 0.0012	37.67 ± 1.23

convergence even with a single iteration per batch. For example, with a relatively small dataset of 100 samples, the number of batches already reaches several tens. Therefore, even if ADMM is run only once per batch, the model effectively performs several dozen iterations overall. Based on the results of this experiment, the convergence of ADMM in this setting is well guaranteed.

Although other factors, such as network scale and the number of hyper parameters, also influence performance, the conducted experiments indicate that, in general, the CSN model behaves similarly to conventional CNNs.

7 | Future Work

The proposed CSN model is built upon a mathematically grounded formulation of convolutional sparse coding, which inherently reduces redundancy during feature extraction. This framework can be potentially incorporated into a wide range of convolution-based architectures, enabling hybrid models that combine the interpretability of CSC with the scalability of modern deep neural networks.

Our normalisation strategy effectively mitigates the vanishing-activation problem previously observed in deeper CSN stacks, allowing the regularisation coefficient to remain constant across layers. This advancement opens the door to constructing deeper CSN variants analogous to modern architectures, such as ResNet, DenseNet, and emerging hybrid families, that integrate attention or multibranch processing. Moreover, recent trends in representation learning—including clustering-driven models such as ClusterFormer or transformer-based grouping mechanisms [54–56]—highlight the importance of global structural cues in feature extraction. Combining these mechanisms with the mathematically grounded sparsity-based processing of CSN may lead to new hybrid architectures that benefit from both local convolutional sparsity and global grouping dynamics.

Another promising direction is to study the impact of additional architectural components—such as nonlinear activation functions (e.g., ReLU), attention modules or multiscale feature fusion—within the CSN framework. Integrating such modules may further improve the representational capacity while preserving sparsity-driven interpretability.

8 | Conclusion

This study introduced a novel deep architecture based on multilayer convolutional sparse coding (ML-CSC), extending the CSN framework to support both unsupervised multichannel feature extraction and supervised classification. The proposed model integrates pooling operations between convolutional layers, along with a normalisation step that improves the stability of feature propagation, while remaining fully compatible with standard deep learning libraries such as Keras. Leveraging the sparsity property inherent in CSN, the model effectively captures discriminative features while reducing data redundancy, computational cost and dependence on large-scale training data. Experimental results demonstrate that the baseline CSN significantly outperforms the baseline CNN when trained on limited samples and achieves comparable performance on larger datasets with substantially lower complexity. In addition, the ADMM-based optimisation exhibited faster and more stable convergence compared to alternative solvers used in related sparse coding models. These findings validate the efficiency and scalability of the proposed architecture for both low-resource and high-resource classification tasks.

Author Contributions

Farhad Sadeghi Almalou: conceptualisation, data curation, formal analysis, investigation, methodology, software, visualisation, writing – original draft. **Farbod Razzazi:** conceptualisation, formal analysis, project administration, resources, supervision, validation, writing – review and editing. **Arash Amini:** formal analysis, supervision, project administration, writing – review and editing.

Funding

The authors have nothing to report.

Conflicts of Interest

The authors declare no conflicts of interest.

Data Availability Statement

The data that support the findings of this study are available from the corresponding author upon reasonable request.

References

1. M. Elad, *Sparse and Redundant Representations* Springer New York, (2010), https://doi.org/10.1007/978-1-4419-7011-4.

2. S. Bambach, D. J. Crandall, L. B. Smith, and C. Yu, "Toddler-Inspired Visual Object Learning," in *Proceedings of the International Conference on Neural Information Processing Systems Workshops, 2011* 2018-Decem, no. NeurIPS (2018): 1201–1210, https://papers.nips.cc/paper/7396-toddler-inspired-visual-object-learning.

3. V. Clay, P. König, K. U. Kühnberger, and G. Pipa, "Learning Sparse and Meaningful Representations Through Embodiment," *Neural Networks* 134 (2021): 23–41, https://doi.org/10.1016/j.neunet.2020.11.004.

4. M. Zang, D. Wen, T. Liu, H. Zou, and C. Liu, "A Fast Sparse Coding Method for Image Classification," *Applied Sciences* 9, no. 3 (2019): 505, https://doi.org/10.3390/app9030505.

5. J. Xu, W. An, L. Zhang, and D. Zhang, "Sparse, Collaborative, or Nonnegative Representation: Which Helps Pattern Classification?," *Pattern Recognition* 88 (2019): 679–688, https://doi.org/10.1016/j.patcog.2018.12.023.

6. W. Chu, H. Xue, C. Yao, and D. Cai, "Sparse Coding Guided Spatiotemporal Feature Learning for Abnormal Event Detection in Large Videos," *IEEE Transactions on Multimedia* 21, no. 1 (2019): 246–255, https://doi.org/10.1109/TMM.2018.2846411.

7. B. Wohlberg, "Convolutional Sparse Representation of Color Images," in *Proceedings of the IEEE Southwest Symposium on Image Analysis and Interpretation*, (2016), 57–60, https://doi.org/10.1109/SSIAI.2016.7459174.

8. J.-F. Cai, S. Liu, and W. Xu, "Projected Wirtinger Gradient Descent for Low-Rank Hankel Matrix Completion in Spectral Compressed Sensing," *IEEE Image, Video, Multidimens. Signal Process. Work* 2015,

no. 10 (2015): 1833–1837: [Online]. Available:, http://arxiv.org/abs/1507.03707.

9. C. Zhong, N. Gong, Z. Zhang, Y. Jiang, and K. Zhang, "Litecclknet: A Lightweight Criss-Cross Large Kernel Convolutional Neural Network for Hyperspectral Image Classification," *IET Computer Vision* 17, no. 7 (October 2023): 763–776, https://doi.org/10.1049/cvi2.12218.

10. Y. Lecun, Y. Bengio, and G. Hinton, "Deep Learning," *Nature* 521, no. 7553 (2015): 436–444, https://doi.org/10.1038/nature14539.

11. Y. LeCun, L. Bottou, Y. Bengio, and P. Haffner, "Gradient-Based Learning Applied to Document Recognition," *Proceedings of the IEEE* 86, no. 11 (1998): 2278–2323, https://doi.org/10.1109/5.726791.

12. Y. Guo, H. Wang, Q. Hu, H. Liu, L. Liu, and M. Bennamoun, "Deep Learning for 3D Point Clouds: A Survey," *IEEE Transactions on Pattern Analysis and Machine Intelligence* 43, no. 12 (2021): 4338–4364, https://doi.org/10.1109/TPAMI.2020.3005434.

13. B. A. Krizhevsky, I. Sutskever, and G. E. Hinton, "Imagenet Classification With Deep Convolutional Neural Networks," *Communications of the ACM* 60, no. 6 (2012): 84–90, https://doi.org/10.1145/3065386.

14. K. Simonyan and A. Zisserman, "Very Deep Convolutional Networks for Large-Scale Image Recognition," *arXiv:1409.1556v6 [cs.CV]*, (September 2014), pp. 1–14, http://arxiv.org/abs/1409.1556.

15. C. Szegedy, V. Vanhoucke, S. Ioffe, J. Shlens, and Z. Wojna, "Rethinking the Inception Architecture for Computer Vision," *Proc. IEEE Conf. Comput. Vis. Pattern Recognition, Las Vegas, NV, USA* (December 2016): 2818–2826, https://doi.org/10.1109/CVPR.2016.308.

16. B. Joan and S. Mallat, "Invariant Scattering Convolution Networks," *IEEE Transactions on Pattern Analysis and Machine Intelligence* 35, no. 8 (2013): 1872–1886, https://doi.org/10.1109/TPAMI.2012.230.

17. S. Mallat, "Group Invariant Scattering," *Communications on Pure and Applied Mathematics* 65, no. 10 (2012): 1331–1398, https://doi.org/10.1002/cpa.21413.

18. J. Zarka, L. Thiry, T. Angles, and S. Mallat, "Deep Network Classification by Scattering and Homotopy Dictionary Learning," *ICLR* (2020): 1–12: [Online]. Available:, http://arxiv.org/abs/1910.03561.

19. R. Giryes, G. Sapiro, and A. M. Bronstein, "Deep Neural Networks With Random Gaussian Weights : A Universal Classification Strategy," *IEEE Transactions on Signal Processing* 64, no. 13 (2016): 3444–3457, https://doi.org/10.1109/TSP.2016.2546221.

20. M. Elad, *Sparse & Redundant Representations and Their Applications in Signal and Image Processing*. 1st ed. Springer Publishing Company, (2010).

21. J. Mairal, F. Bach, and J. Ponce, "Sparse Modeling for Image and Vision Processing," *Foundations and Trends in Computer Graphics and Vision* 8, no. 2–3 (2014): 85–283, https://doi.org/10.1561/0600000058.

22. G. J. Peng, "Joint and Direct Optimization for Dictionary Learning in Convolutional Sparse Representation," *IEEE Transactions on Neural Networks and Learning Systems* 31, no. 2 (2020): 559–573, https://doi.org/10.1109/TNNLS.2019.2906074.

23. Y. Wang, J. T. Kwok, and L. M. Ni, "Generalized Convolutional Sparse Coding With Unknown Noise," *IEEE Transactions on Image Processing* 29 (2020): 5386–5395, https://doi.org/10.1109/TIP.2020.2980980.

24. J. Kalifa, S. Mallat, and B. Rougé, "Deconvolution by Thresholding in Mirror Wavelet Bases," *IEEE Transactions on Image Processing* 12, no. 4 (2003): 446–457, https://doi.org/10.1109/TIP.2003.810592.

25. I. Daubechies, M. Defrise, and C. De Mol, "An Iterative Thresholding Algorithm for Linear Inverse Problems With a Sparsity Constraint," *Communications on Pure and Applied Mathematics* 57, no. 11 (2004): 1413–1457, https://doi.org/10.1002/cpa.20042.

26. Y. Romano and M. Elad, "Patch-Disagreement as a Way to Improve K-SVD Denoising," in *2015 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, South Brisbane, QLD, Australia IEEE, (2015), https://doi.org/10.1109/ICASSP.2015.7178176.

27. F. Heide, W. Heidrich, and G. Wetzstein, "Fast and Flexible Convolutional Sparse Coding," *Proc. IEEE Comput. Soc. Conf. Comput. Vis. Pattern Recognit.* (2015): 5135–5143, https://doi.org/10.1109/CVPR.2015.7299149.

28. B. Wohlberg, "Efficient Convolutional Sparse Coding," *ICASSP, IEEE International Conference on Acoustics, Speech and Signal Processing - Proceedings* (2014): 7173–7177, https://doi.org/10.1109/ICASSP.2014.6854992.

29. V. Papyan, J. Sulam, and M. Elad, "Working Locally Thinking Globally: Theoretical Guarantees for Convolutional Sparse Coding," *IEEE Transactions on Signal Processing* 65, no. 21 (2017): 5687–5701, https://doi.org/10.1109/TSP.2017.2733447.

30. M. D. Zeiler, G. W. Taylor, and R. Fergus, "Adaptive Deconvolutional Networks for mid and High Level Feature Learning," in *Proceedings of the IEEE International Conference on Computer Vision*, (2011), 2018–2025, https://doi.org/10.1109/ICCV.2011.6126474.

31. J. Mairal, F. Bach, and J. Ponce, "Task-Driven Dictionary Learning," *IEEE Transactions on Pattern Analysis and Machine Intelligence* 34, no. 4 (2012): 791–804, https://doi.org/10.1109/TPAMI.2011.156.

32. V. Papyan, Y. Romano, and M. Elad, "Convolutional Neural Networks Analyzed via Convolutional Sparse Coding," *Journal of Machine Learning Research* 18, no. 1 (2017): 1–52, https://doi.org/10.48550/arXiv.1607.08194.

33. V. Papyan, Y. Romano, J. Sulam, and M. Elad, "Theoretical Foundations of Deep Learning via Sparse Representations: A Multilayer Sparse Model and its Connection to Convolutional Neural Networks," *IEEE Signal Processing Magazine* 35, no. 4 (2018): 72–89, https://doi.org/10.1109/MSP.2018.2820224.

34. J. Sulam, A. Aberdam, A. Beck, and M. Elad, "On Multi-Layer Basis Pursuit, Efficient Algorithms and Convolutional Neural Networks," *IEEE Transactions on Pattern Analysis and Machine Intelligence* 42, no. 320649 (2019): 1–13, https://doi.org/10.1109/TPAMI.2019.2904255.

35. D. L. Donoho and M. Elad, "Optimally Sparse Representation in General (Nonorthogonal) Dictionaries via $\ell 1$ Minimization," *Proceedings of the National Academy of Sciences of the United States of America* 100, no. 5 (2003): 2197–2202, https://doi.org/10.1073/pnas.0437847100.

36. J. Sulam, V. Papyan, Y. Romano, and M. Elad, "Multilayer Convolutional Sparse Modeling: Pursuit and Dictionary Learning," *IEEE Transactions on Signal Processing* 66, no. 15 (2018): 4090–4104, https://doi.org/10.1109/TSP.2018.2846226.

37. A. Aberdam, J. Sulam, and M. Elad, "Multi-Layer Sparse Coding: The Holistic Way," *SIAM Journal on Mathematics of Data Science* 1, no. 1 (April 2019): 46–77, https://doi.org/10.1137/18m1183352.

38. S. Chan Wai Tim, M. Rombaut, and D. Pellerin, "Multi-Layer Dictionary Learning for Image Classification," *International Conference on Advanced Concepts for Intelligent Vision Systems* 10016 (2016): 522–533: LNCS, https://doi.org/10.1007/978-3-319-48680-2_46.

39. A. Montazeri, M. Shamsi, and R. Dianat, "MLK-SVD, the New Approach in Deep Dictionary Learning," *Visual Computer* 37, no. 4 (2021): 707–715, https://doi.org/10.1007/s00371-020-01970-x.

40. S. Tariyal, A. Majumdar, R. Singh, and M. Vatsa, "Deep Dictionary Learning," *IEEE Access* 4, no. Dl (2016): 10096–10109, https://doi.org/10.1109/ACCESS.2016.2611583.

41. V. Singhal and A. Majumdar, "Supervised Deep Dictionary Learning for Single Label and Multi-Label Classification," *International Joint Conference on Neural Networks* (2018): 1–7, https://doi.org/10.1109/ijcnn.2018.8489682.

42. S. S. Chen, D. L. Donoho, and M. A. Saunders, "Atomic Decomposition by Basis Pursuit," *SIAM Journal on Scientific Computing* 20, no. 1 (1998): 33–61, https://doi.org/10.1137/s1064827596304010.

43. Y. Romano, A. Aberdam, J. Sulam, and M. Elad, "Adversarial Noise Attacks of Deep Learning Architectures: Stability Analysis via Sparse-Modeled Signals," *Journal of Mathematical Imaging and Vision* 62, no. 3 (2020): 313–327, https://doi.org/10.1007/s10851-019-00913-z.

44. S. Boyd, N. Parikh, E. Chu, B. Peleato, and J. Eckstein, "Distributed Optimization and Statistical Learning via the Alternating Direction Method of Multipliers," *Foundations and Trends in Machine Learning* 3, no. 1 (2010): 1–122, https://doi.org/10.1561/2200000016.

45. M. V Afonso, J. M. Bioucas-Dias, and M. A. T. Figueiredo, "An Augmented Lagrangian Approach to the Constrained Optimization Formulation of Imaging," *Inverse Problems* (December 2009): 1–13, https://doi.org/10.1109/TIP.2010.2076294.

46. B. Wohlberg, "Efficient Algorithms for Convolutional Sparse Representations," *IEEE Transactions on Image Processing* 25, no. 1 (2016): 301–315, https://doi.org/10.1109/TIP.2015.2495260.

47. N. Parikh and S. Boyd, "Proximal Algorithms," *Foundations and Trends in Optimization* 1, no. 3 (2014): 127–239, https://doi.org/10.1561/2400000003.

48. Y. Netzer, T. Wang, A. Coates, A. Bissacco, B. Wu, and A. Y. Ng, "Reading Digits in Natural Images With Unsupervised Feature Learning," in *Proceedings of the International Conference on Neural Information Processing Systems Workshops, 2011*, vol. 2011, no. 5, p. 7 (2011).

49. A. Krizhevsky, Learning Multiple Layers of Features From Tiny Images. (2009).

50. H. Xiao, K. Rasul, and R. Vollgraf. (2017), Fashion-MNIST: A Novel Image Dataset for Benchmarking Machine Learning Algorithms, http://arxiv.org/abs/1708.07747.

51. W. W. CukierskiCukierski, *Dogs Vs. Cats* Kaggle, https://kaggle.com/competitions/dogs-vs-cats.

52. P. O. Hoyer, "Non-Negative Matrix Factorization With Sparseness Constraints," *Journal of Machine Learning Research* 5, no. 9 (2004): 1457–1469, https://openurl.ebsco.com/EPDB%3Agcd%3A1%3A13750657/detailv2?sid=ebsco%3Aplink%3Ascholar&id=ebsco%3Agcd%3A19080031&crl=c&link_origin=scholar.google.com.

53. R. Chalasani, J. C. Principe, N. Ramakrishnan, et al., "A Fast Proximal Method for Convolutional Sparse Coding," *Proceedings of the International Joint Conference on Neural Networks* (2013): 1058–1062, https://doi.org/10.1109/IJCNN.2013.6706854.

54. J. C. Liang, *C LUSTER F ORMER : Clustering as A Universal Visual Learner* (NeurIPS, 2023).

55. X. Ma, "Image as Set of Points," *11th International Conference on Learning Representations* (2023): 1–18.

56. G. Chen, X. Li, Y. Yang, et al., "Neural Clustering Based Visual Representation Learning," *Proceedings of the IEEE conference on computer vision and pattern Recognition* (2024): 5714–5725, https://doi.org/10.1109/CVPR52733.2024.00546.