Attn-HybridNet: Improving Discriminability of Hybrid Features with Attention Fusion

Sunny Verma, Chen Wang, Liming Zhu, *Member, IEEE* and Wei Liu, *Senior Member, IEEE*

***Abstract*—The principal component analysis network (PCANet) is an unsupervised deep network, utilizing principal components as convolution filters in its layers. Albeit powerful, the PCANet suffers from two fundamental problems responsible for its performance degradation. First, the *principal components* transforms the data as column vectors (which we call the amalgamated view) and incurs a loss of spatial information present in the data. Second, the generalized pooling in the PCANet is unable to incorporate spatial statistics of the natural images, and it also induces redundancy among the features. In this research, we first propose a tensor-factorization based deep network called the Tensor Factorization Network (TFNet). The TFNet extracts feature by preserving the spatial view of the data (which we call the minutiae view). We then proposed *HybridNet*, which simultaneously extracts information with the two views of the data as their integration can improve the performance of classification systems. Lastly, to alleviate the feature redundancy among hybrid features, we propose *Attn-HybridNet* to perform attention-based feature selection and fusion to improve their discriminability. Classification results on multiple real-world datasets using features extracted by our proposed *Attn-HybridNet* achieves significantly better performance over other popular baseline methods, demonstrating the effectiveness of the proposed techniques.**

***Index Terms*—Tensor Decomposition, Feature Extraction, Attention Networks, Feature Fusion**

I. Introduction

The deep neural networks perform sophisticated sequential operations to discover effective data representations [1]. However, the training time required to obtain these superior data representations is exponentially large as these networks have an exhaustive hyperparameter search space and usually suffer from various training difficulties [2]. Besides, the deep networks are complex models that require high computational resources for their training and deployment, and therefore, their usability is limited on micro-devices such as cellphones [3], [4]. In this regard, the current research trend on reducing the computational requirements with the deep networks either 1) approximate the network's layers by factorization [5], [6], 2) compress the layers with quantization (or hashing) [4], [7], or, 3) replace the conventional layer's with a tensorized layer and optimize the weights of this layer by retraining [8]. Concludingly, all these techniques can only reduce the computational footprint of a trained deep neural network.The aim of this research is to build deep networks that are computationally inexpensive. In other words, the aim is to develop deep networks that are independent of a) high-performance hardware, and b) exhaustive hyperparameter space, where one promising architecture is the PCANet [9].

The PCANet is an unsupervised-deep-parsimonious network utilizing *principal components* as convolution filters in its cascaded layers. It achieved remarkable performance on several benchmark face datasets but did not achieve competitive performance on challenging object recognition datasets such as CIFAR-10 [10]. There are two major reasons for this performance degradation in natural images datasets: 1) the PCANet incurs a loss of spatial information exhibiting in the data as it vectorizes the data while extracting *principal components* and, 2) the PCANet's output layer (which is spatial-pooling) induces feature redundancy and does not adapt to the structure of the natural images and thus deteriorates classifiers performance [11], [12]. However, the data vectorization is inherent with the *principal components*, and thus requires alternate techniques which alleviates the loss of spatial information. In other words, techniques that can extract information from the untransformed view of the data[1], which is proven to be beneficial in literature [13], [14], [15], [16].

In this research, we first propose an unsupervised tensor factorization based deep network called Tensor Factorization Network (TFNet). *The TFNet, contrary to the PCANet, does not vectorize the data while obtaining weights for its convolution filters.* Therefore, it is able to extract information associated with the spatial structure of the data or the minutiae view of the data. Besides, the information is independently obtained from each mode of the data, providing several degrees of freedom to the information extraction procedure of TFNet. Secondly, we hypothesize that integration of information obtained from the amalgamated view and the minutiae view can enhance the performance of classification systems as the information from these views represent complementary information of the data[2] [17], [14]. In this regard, we propose the *Hybrid Network* (*HybridNet*) that integrates information discovery and feature

Manuscript submitted on June 30, 2019, Revised on 12 Feb 2020, 25 May 2020, and 25 October 2020. Accepted on February 05, 2021.

Sunny Verma is with the Data Science Institute, University of Technology Sydney, Australia (email: Sunny.Verma@uts.edu.au

Wei Liu is with the School of Computer Science, University of Technology Sydney, Australia (email: Wei.Liu@uts.edu.au).

Chen Wang and Liming Zhu are with Data61, Commonwealth Scientific and Industrial Research Organization, CSIRO, Sydney, Australia (email: Chen.Wang@data61.csiro.au, Liming.Zhu@data61.csiro.au).

The source code of proposed technique and extracted features are available at https://github.com/sverma88/Attn-HybridNet-IEEE-TCYB.

[1]Throughout this paper we refer to the vectorized presentation of the data as the amalgamated view where all modes of the data (also called dimension for higher order-matrices, i.e. tensors) are collapsed to obtain a vector. The untransformed view of the data, i.e., when viewed with its multiple modes (e.g., tensors), is referred to as the minutiae view of the data.

[2]Throughout this paper, by two views, we mean the amalgamated view and the minutiae view.

TABLE I: Comparison of different feature extraction models

Methods	Amalgamated View	Minutiae View	Attention Fusion
PCANet [9]	✓	×	×
TFNet [18]	×	✓	×
HybridNet [18]	✓	✓	×
Attn-HybridNet	✓	✓	✓

extraction from the two views of the data simultaneously. However, the *HybridNet* still suffers from feature redundancy arising from the usage of a generalized spatial pooling layer. Thus we propose an attention-based fusion scheme *Attn-HybridNet* that performs feature selection and aggregation, thus enhancing the feature discriminability in hybrid features. The superiority of feature representations obtained with the *Attn-HybridNet* is demonstrated by performing comprehensive experiments on multiple real-world benchmark datasets. The differences and similarities between the PCANet, TFNet, *HybridNet*, and *Attn-HybridNet* from data view perspectives are summarized in Table. I.

Our contributions in this paper are summarized below:

- We propose *Tensor Factorized Network* (TFNet), which extracts features from the minutiae view of the data and hence is able able to preserve the spatial information in the data that is proven beneficial for image classification.
- We propose a Left one Mode Out Orthogonal Iteration (*LoMOI*) algorithm to optimize convolution weights from the minutiae view of the data in the proposed TFNet.
- We introduce the *Hybrid Network* (*HybridNet*), which integrates the feature extraction and information discovery procedure from two views of the data. This integration procedure reduces information loss from the data by combining the merits of the PCANet and TFNet and obtains superior features from both of the two schemes.
- We propose the *Attn-HybridNet* to alleviate the feature redundancy among hybrid features. The *Attn-HybridNet* performs feature selection and aggregation with an attention-based fusion scheme, thus enhancing the discriminability of the feature representations.
- We perform comprehensive evaluations and case studies to demonstrate the effectiveness of features obtained by *Attn-HybridNet* and *HybridNet* on multiple benchmark real-world datasets.

The rest of the paper is organized as follows: in Sec. II we present the literature review including prior works and background on PCANet and tensor preliminaries. We then present the details of our proposed TFNet, *HybridNet*, and *Attn-HybridNet* Sec. III, Sec. IV, and Sec. V respectively. Next we describe our experimental setup, results and discussions in Sec. VI and Sec. VII. Finally, we conclude our work and specify the future directions for its improvement in Sec. VIII.

II. Literature Review

The utilization of CNNs has significantly advanced the field of computer vision, whereby the CNNs have started surpassing human performances, albeit requiring enormous computational resources for this advancement. For example, ResNet with 152-layers [19] achieves a top-5 error rate of 3.57% and performs 2.25×10^{10} flops of the data at inference. This substantial computational cost restricts the applicability of such models, and thus reducing the computational footprint of the CNNs has become a non-trivial task. In this regard, researchers actively pursuit three main active research directions: 1) compressing CNNs weights with quantization, 2) approximating convolution layers with factorization, and 3) replacing fully connected layers with custom-built layers.

In the first category, the size of trained CNNs is reduced by applying either quantization or hashing, as in [4], [7], [20]. These quantized CNNs have significantly less computational requirements and achieve similar recognition accuracy during inference. Similarly, the works in [5], [6] obtain approximations of fully connected and convolution layers by utilizing factorization for compressing the CNN models. However, both the quantization and factorization based methods compress a pre-trained CNN model instead of building a smaller or faster CNN model in the first place. Therefore, these techniques inherit the limitations of the pre-trained CNN models.

In the second and the third categories, the fully connected layers are replaced with customized layers that substantially reduce the size of any CNN model. For example, Kossaifi et. al. [8] proposes a neural tensor layer, and Passalis et. al. [3] proposes a BoF (Bag-of-features) layer as a neural pooling layer, which are lightweight versions of conventional CNNs layers and is trainable in an end-to-end fashion. However, a major limitation of these work is that they can only replace a fully connected layer, and while replacing a convolution layer, they work similarly to the works in the first category.

Contrary to the above, PCANet [9] and TFNN [15] are proposed as lightweight CNNs with lower computational requirements on smaller size images. While the PCANet is a deep unsupervised parsimonious feature extractor, the TFNN is a supervised CNN architecture utilizing neural tensor factorizations on multiway data. Both these networks achieve very high classification performance on handwritten digits dataset but fail to obtain competitive performance on object recognition datasets. This is because the PCANet incurs information loss as it obtains weights of its convolution filters from the amalgamated view of the data. Contrarily, the TFNN extracts information by isolating each view of the multi-view data and fails to efficiently consolidate them, incurring the loss of common information present in the data. Since the information from both the amalgamated view and the minutiae view is essential for classification, their integration is shown beneficial in classification performance [14], [17], [21]. Therefore, we propose *HybridNet*, which integrates the two kinds of information in its deep parsimonious feature extraction architecture. A major difference between *HybridNet* and PCANet is that the earlier obtain information from both views of the data, whereas the latter is restricted with the amalgamated view. It is also notably different than TFNN as the earlier is an unsupervised network, whereas the latter is a supervised deep network.

A. Background

We briefly summarize PCANet's 2-layer architecture and provide background on tensor preliminaries in this section.

1) The First Layer: The procedure begins by extracting overlapping patches of size $k_1 \times k_2$ around each pixel in the image; where patches from image $\boldsymbol{I}_i$ are denoted as $\mathbf{x}_{i,1}, \mathbf{x}_{i,2}, ..., \mathbf{x}_{i,\tilde{m}\tilde{n}} \in \mathbb{R}^{k_1 k_2}$, $\tilde{m} = m - \lceil \frac{k_1}{2} \rceil$[3] and $\tilde{n} = n - \lceil \frac{k_2}{2} \rceil$. Next, the obtained patches are zero-centered by subtracting the mean of the image patches and *vectorized* to obtain the patch matrix $\boldsymbol{X}_i \in \mathbb{R}^{k_1 k_2 \times \tilde{m}\tilde{n}}$. The same procedure is repeated for all the training images to obtain the final patch-matrix as $\boldsymbol{X} \in \mathbb{R}^{k_1 k_2 \times N\tilde{m}\tilde{n}}$ from which the pca filters are obtained. The PCA minimizes the reconstruction error with orthonormal filters as the principal eigenvectors as calculated in Eq. 1

$$\min_{\boldsymbol{V} \in \mathbb{R}^{k_1 k_2 \times L_1}} \|\boldsymbol{X} - \boldsymbol{V}\boldsymbol{V}^T\boldsymbol{X}\|_F, \ s.t. \quad \boldsymbol{V}^T V = \boldsymbol{I}_{L_1} \tag{1}$$

where L_1 is the total number of filters obtained in this layer and $\boldsymbol{I}_{L_1}$ is an identity matrix of size $L_1 \times L_1$. These filters are then expressed as below:

$$\boldsymbol{W}^1_{l_{PCANet}} = mat_{k_1,k_2}(ql(\boldsymbol{X}\boldsymbol{X}^T)) \in \mathbb{R}^{k_1 \times k_2} \tag{2}$$

where $mat_{k_1,k_2}(v)$ is a function that maps $v \in \mathbb{R}^{k_1 k_2}$ to a matrix $\boldsymbol{W} \in \mathbb{R}^{k_1 \times k_2}$, and $ql(\boldsymbol{X}\boldsymbol{X}^T)$ denotes the l-th principal eigenvector of $\boldsymbol{X}\boldsymbol{X}^T$. Next, each training image $\boldsymbol{I}_i$ is convolved with the L_1 filters as in Eq. 3.

$$\boldsymbol{I}^l_{i_{PCANet}} = \boldsymbol{I}_i * \boldsymbol{W}^1_{l_{PCANet}} \tag{3}$$

where $*$ denotes the 2D convolution and i, l are the image and filter indices respectively. Importantly, the boundary of image $\boldsymbol{I}_i$ is padded before convolution to obtain $\boldsymbol{I}^l_{i_{PCANet}}$ with the same dimensions as in $\boldsymbol{I}_i$. From Eq. 3 a total of $N \times L_1$ images are obtained and attributed as the output from the first layer.

2) The Second Layer: The methodology of the second layer is similar to the first layer. We collect overlapping patches from all input images i.e. from $\boldsymbol{I}^l_{i_{PCANet}}$, zero-centre these images patches and vectorize them to obtain the final matrix denoted as $\boldsymbol{Y} \in \mathbb{R}^{k_1 k_2 \times L_1 N\tilde{m}\tilde{n}}$. This patch matrix is then utilized to obtain the convolution filters in layer 2 as in Eq. 4.

$$\boldsymbol{W}^2_{l_{PCANet}} = mat_{k_1,k_2}(ql(\boldsymbol{Y}\boldsymbol{Y}^T)) \in \mathbb{R}^{k_1 \times k_2} \tag{4}$$

where $l = [1, L_2]$ denotes the number of pca filters obtained in this layer. Next, the input images in this layer $\boldsymbol{I}^l_{i_{PCANet}}$ are convolved with the learned filters $\boldsymbol{W}^2_{l_{PCANet}}$ to obtain the output from this layer in Eq. 5. These images are then passed to the feature aggregation phase as in the next subsection.

$$\boldsymbol{O}^l_{i_{PCANet}} = \boldsymbol{I}^l_{i_{PCANet}} * \boldsymbol{W}^2_{l_{PCANet}} \tag{5}$$

3) The Output Layer: The output layer consolidates the output from all the convolution layers of the PCANet to obtain the feature vectors. It first binarizes each of the real-valued outputs from Eq. 5 with a step function $H(\boldsymbol{O}^l_{i_{PCANet}})$ that converts the positive entries to 1 otherwise 0. Next, these outputs are assembled into L_1 batches, where all images in a batch correspond to the same filter in the first layer. These images are combined together by applying weighted sum and partitioned into B blocks and summarized as histograms.

$$\boldsymbol{I}^l_{i_{PCANet}} = \sum_{l=1}^{L_2} 2^{l-1} H(\boldsymbol{O}^2_{l_{PCANet}}) \tag{6}$$

[3]The operator $\lceil z \rceil$ gives the smallest integer greater than or equal to z.

Finally, the histograms from all the B blocks are concatenated to form feature vectors of the images, as in Eq. 7.

$$f_{i_{PCANet}} = [Bhist(\boldsymbol{I}^1_{i_{PCANet}}), ..., Bhist(\boldsymbol{I}^{L_1}_{i_{PCANet}})]^T \in \mathbb{R}^{(2^{L_2})L_1 B} \tag{7}$$

This block-wise encoding process encapsulates the L_1 images from Eq. 6 into a single feature vector which can be utilized for any machine learning task like clustering or classification.

B. Tensor Preliminaries

Tensors are higher-order[4] matrices of dimension > 2 which we write n Euler symbols such as $\boldsymbol{\mathcal{X}}$. A few important multilinear operations are described below.

a) Matriziation: also known as tensor unfolding, is the operation to rearrange the elements of an n-mode tensor $\boldsymbol{\mathcal{X}} \in \mathbb{R}^{i_1 \times i_2 ... \times i_N}$ as matrix $\boldsymbol{X}_{(n)} \in \mathbb{R}^{i_n \times j}$ on the chosen mode *n*, where $j = \big(i_1 \ ... \times i_{n-1} \times i_{n+1} ... \times i_N\big)$.

b) n-mode Product: the product of an n-mode tensor $\boldsymbol{\mathcal{X}} \in \mathbb{R}^{i_1 ... \times i_{m-1} \times i_m \times i_{m+1} ... \times i_n}$ and a matrix $\boldsymbol{A} \in \mathbb{R}^{j \times i_n}$ is denoted as $\boldsymbol{\mathcal{X}} \times_n \boldsymbol{A}$. The resultant of this product is also a tensor $\boldsymbol{\mathcal{Y}} \in \mathbb{R}^{i_1 \times i_2 \times i_{n-1} \times j \times i_{n+1} ... \times i_n}$ which can also be expressed through matricized tensor as $\boldsymbol{Y}_{(n)} = \boldsymbol{A}\boldsymbol{X}_{(n)}$.

c) Tensor Decomposition: Tensor decomposition is a form of generalized matrix factorization for approximating multimode tensors. The factorization an n-mode tensor $\boldsymbol{\mathcal{X}} \in \mathbb{R}^{i_1 \times i_2 ... \times i_n}$ obtains two sub components: 1) $\boldsymbol{\mathcal{G}} \in \mathbb{R}^{r_1 \times r_2 ... \times r_n}$ which is a lower dimensional tensor called the *core-tensor* and, 2) $\boldsymbol{U}^{(j)} \in \mathbb{R}^{r_n \times i_n} \forall j = [1, n]$ which are matrix factors associated with each mode of the tensor. The entries in the *core-tensor* $\boldsymbol{\mathcal{G}}$ signify the interaction level between tensor elements. The factor matrices $\boldsymbol{U}^{(n)}$ are analogous to *principal components* associated with the respective mode-n. This scheme of tensor factorization falls under the *Tucker* family of tensor decomposition [22]. The original tensor $\boldsymbol{\mathcal{X}}$ can be reconstructed by taking the n-mode product of the *core-tensor* and the factor matrices as in Eq. 8.

$$\boldsymbol{\mathcal{G}} \times_1 \boldsymbol{U}^{(1)} \times_2 \boldsymbol{U}^{(2)} ... \times_N \boldsymbol{U}^{(n)} \approx \boldsymbol{\mathcal{X}} \tag{8}$$

The advantages of *Tucker* based factorization methods are already studied in several domains such as computer vision, [23], data mining [24], and signal processing [25], [22]. However, in this research, we factorize tensor to obtain weights of convolution-tensorial filters for TFNet by devising our custom tensor factorization scheme, which we call as Left one Mode Out Orthogonal Iteration (*LoMOI*) presented in Alg. 1.

III. The Tensor Factorization Network

The development of *Tensor Factorization Network* (TFNet) is motivated to reduce the loss of spatial information while vectorizing image patches in the PCANet. This transformation destroys the geometric structure of the data and is inherent to the *principal components* albeit, proven beneficial in many image classification tasks [23], [14], [15]. Furthermore, the vectorization also results in high dimensional data and generally requires more computational resources. Motivated by

[4]Also known as modes (dimensions) of a tensor and are analogous to rows and columns of a matrix.

Algorithm 1: Left One Mode Out Orthogonal Iteration, (*LoMOI*)

Input: n-mode tensor $\mathcal{X} \in \mathbb{R}^{i_1,i_2,\ldots,i_n}$; factorization ranks for each mode of the tensor $[r_1 \ldots r_{m-1}, r_{m+1} \ldots r_n]$, where $r_k \leq i_k \forall\ k \in 1,2,\ldots,n$ and $k \neq m$; factorization error-tolerance ε, and Maximum allowable iterations $= Maxiter$, m = mode to discard while factorizing
for $i = 1,2,\ldots,n$ and $i \neq m$ **do**
  $\boldsymbol{X}_i \leftarrow$ unfold tensor $\mathcal{X}$ on mode-i
  $\boldsymbol{U}^{(i)} \leftarrow r_i$ left singular vectors of $\boldsymbol{X}_i$ ▷ extract leading r_i matrix factors
$\mathcal{G} \leftarrow \mathcal{X} \times_1 (\boldsymbol{U}^{(1)})^T \ldots \times_{m-1} (\boldsymbol{U}^{(m-1)})^T \times_{m+1} (\boldsymbol{U}^{(m+1)})^T \ldots \times_n (\boldsymbol{U}^{(n)})^T$ ▷ Core tensor
$\hat{\mathcal{X}} \leftarrow \mathcal{G} \times_1 \boldsymbol{U}^{(1)} \ldots \times_{m-1} \boldsymbol{U}^{(m-1)} \times_{m+1} \boldsymbol{U}^{(m+1)} \times_n \boldsymbol{U}^{(n)}$ ▷ reconstructed tensor obtained by multilinear product of the core-tensor with the factor-matrices; Eq. 8.
$loss \leftarrow \|\mathcal{X} - \hat{\mathcal{X}}\|$ ▷ decomposition loss
$count \leftarrow 0$
while $[(loss \geq \varepsilon)\ Or\ (Maxiter \leq count)]$ **do** ▷ loop until convergence
  for $i = 1,2,\ldots,n$ and $i \neq m$ **do**
    $\mathcal{Y} \leftarrow \mathcal{X} \times_1 (\boldsymbol{U}^{(1)})^T \ldots \times_{(i-1)} (\boldsymbol{U}^{(i-1)})^T \times_{(i+1)} (\boldsymbol{U}^{(i+1)})^T \ldots \times_n (\boldsymbol{U}^{(n)})^T$ ▷ obtain the variance in mode-i
    $\boldsymbol{Y}_i \leftarrow$ unfold tensor $\mathcal{Y}$ on mode-i
    $\boldsymbol{U}^{(i)} \leftarrow \mathbf{r}_i$ left singular vectors of $\boldsymbol{Y}_i$
  $\mathcal{G} \leftarrow \mathcal{X} \times_1 (\boldsymbol{U}^{(1)})^T \ldots \times_{(m-1)} (\boldsymbol{U}^{(m-1)})^T \times_{(m+1)} (\boldsymbol{U}^{(m+1)})^T \ldots \times_n (\boldsymbol{U}^{(n)})^T$
  $\hat{\mathcal{X}} \leftarrow \mathcal{G} \times_1 \boldsymbol{U}^{(1)} \ldots \times_{(m-1)} \boldsymbol{U}^{(m-1)} \times_{(m+1)} \boldsymbol{U}^{(m+1)} \ldots \times_n \boldsymbol{U}^{(n)}$
  $loss \leftarrow \|\mathcal{X} - \hat{\mathcal{X}}\|$
  $count \leftarrow count + 1$
Output: $\hat{\mathcal{X}}$ the reconstructed tensor and $[\boldsymbol{U}^{(1)} \ldots \boldsymbol{U}^{(m-1)}, \boldsymbol{U}^{(m+1)} \ldots \boldsymbol{U}^{(n)}]$ the factor matrices

the above, we propose the TFNet that preserves the spatial structure of the data (called as the minutiae view) while obtaining weights of its convolution-tensor filters. The unsupervised feature extraction procedure of the TFNet is described in the next subsection.

A. The First Layer

Similar to the PCANet, we begin by collecting all overlapping patches of size $k_1 \times k_2$ from the image $\boldsymbol{I}_i$. However, contrary to PCANet the spatial structure of these patches are preserved and instead of matrix and we obtain a 3-mode tensor $\mathcal{X}_i \in \mathbb{R}^{k_1 \times k_2 \times \tilde{m}\tilde{n}}$. The mode-1 and mode-2 of this tensor represent the row-space, and the column-space spanned by the pixels in the image. Whereas the mode-3 of this tensor represents the total number of image patches obtained from the input image. Iterating this process for all the training images, we obtain $\mathcal{X} \in \mathbb{R}^{k_1 \times k_2 \times N\tilde{m}\tilde{n}}$ as our final patch-tensor. The matrix factors utilized to generate our convolution-tensorial filters for to the first two modes of $\mathcal{X}$ are obtained with our custom-designed *LoMOI* (presented in Alg. 1) in Eq. 9.

$$[\hat{\mathcal{X}}, \boldsymbol{U}^{(1)}, \boldsymbol{U}^{(2)}] \leftarrow LoMOI(\mathcal{X}, r_1, r_2) \tag{9}$$

where $\hat{\mathcal{X}} \in \mathbb{R}^{r_1 \times r_2 \times N\tilde{m}\tilde{n}}$, $\boldsymbol{U}^{(1)} \in \mathbb{R}^{k_1 \times r_1}$, and $\boldsymbol{U}^{(2)} \in \mathbb{R}^{k_2 \times r_2}$. We discard obtaining the matrix factors from mode-3 of tensor $\mathcal{X}$ (which is $\boldsymbol{X}_3$) as this is equivalent to the transpose of the matrix $\boldsymbol{X}$ in the first layer of the PCANet that is not factorized in the PCANet. Moreover, the matrix factors for this mode span the sample space of the data, which is trivial. A total of $L_1 = r_1 \times r_2$ convolution-tensor filters are obtained from the factor matrices $\boldsymbol{U}^{(1)}$ and $\boldsymbol{U}^{(2)}$ as in Eq. 10.

$$\boldsymbol{W}^1_{l_{TFNet}} = \boldsymbol{U}^{(1)}_{(:,i)} \otimes \boldsymbol{U}^{(2)}_{(:,j)} \in \mathbb{R}^{k_1 \times k_2} \tag{10}$$

where '$\otimes$' denotes the *outer*-product and $\boldsymbol{U}^{(m)}_{(:,i)}$ represents 'i^{th}' column of the 'm^{th}' factor matrix. Importantly, our convolution-tensorial filters do not require any explicit reshaping as the *outer*-product naturally results in a matrix. Therefore, we can straightforwardly convolve the input images with our convolution-tensorial filters as described in Eq. 11 where i and l are the image and filter indices respectively.

$$\boldsymbol{I}^l_{i_{TFNet}} = \boldsymbol{I}_i * \boldsymbol{W}^1_{l_{TFNet}} \tag{11}$$

However, whenever the data is an RGB-image, each extracted patch from the image is a 3-order tensor $\mathcal{X} \in \mathbb{R}^{k_1 \times k_2 \times 3}$ (i.e., *RowPixels*×*ColPixels*×Color). After collecting patches from all the training images, we obtain a 4-mode tensor as $\mathcal{X} \in \mathbb{R}^{k_1 \times k_2 \times 3 \times N\tilde{m}\tilde{n}}$ which is decomposed by utilizing *LoMOI* ($[\hat{\mathcal{X}}, \boldsymbol{U}^{(1)}, \boldsymbol{U}^{(2)}, \boldsymbol{U}^{(3)}] \leftarrow LoMOI(\mathcal{X}, r_1, r_2, r_3)$) for obtaining the convolution-tensorial filters in Eq. 12.

$$\boldsymbol{W}^1_{l_{TFNet}} = \boldsymbol{U}^{(1)}_{(:,i)} \otimes \boldsymbol{U}^{(2)}_{(:,j)} \otimes \boldsymbol{U}^{(3)}_{(:,k)} \tag{12}$$

B. The Second Layer

Similar to the first layer, we extract overlapping patches from the input images and zero-center them to build a 3-mode patch-tensor denoted as $\mathcal{Y} \in \mathbb{R}^{k_1 \times k_2 \times NL_1\tilde{m}\tilde{n}}$ which is then decomposed as $[\hat{\mathcal{Y}}, \boldsymbol{V}^{(1)}, \boldsymbol{V}^{(2)}] \leftarrow LoMOI(\mathcal{Y}, r_1, r_2)$ to obtain the convolution-tensor filters for layer 2 in Eq. 13.

$$\boldsymbol{W}^2_{l_{TFNet}} = \boldsymbol{V}^{(1)}_{(:,i)} \otimes \boldsymbol{V}^{(2)}_{(:,j)} \in \mathbb{R}^{k_1 \times k_2} \tag{13}$$

where, $\hat{\mathcal{Y}} \in \mathbb{R}^{r_1 \times r_2 \times NL_1\tilde{m}\tilde{n}}$, $\boldsymbol{V}^{(1)} \in \mathbb{R}^{k_1 \times r_1}$, and $\boldsymbol{V}^{(2)} \in \mathbb{R}^{k_2 \times r_2}$. We, now convolve each of the input images from the first layer with the convolution-tensorial filters as in Eq. 14.

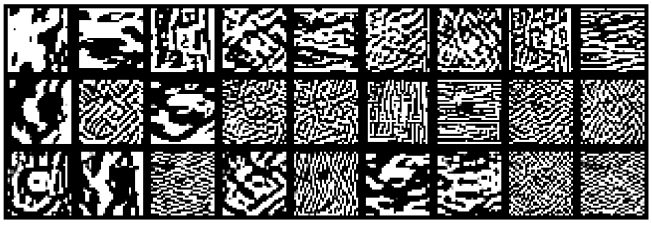
$$\boldsymbol{O}^l_{i_{TFNet}} = \boldsymbol{I}^l_{i_{TFNet}} * \boldsymbol{W}^2_{l_{TFNet}}, \quad l = 1,2,\ldots,L_2 \tag{14}$$

The number of output images obtained here is equal to $L_1 \times L_2$ which is identical to the number of images obtained at layer 2 of PCANet. Finally, we utilize the output layer of PCANet (Sec. II-A3) to obtain the feature vectors from the minutiae view of the image in Eq. 15.

$$\begin{aligned} \boldsymbol{I}^l_{i_{TFNet}} &= \sum_{l=1}^{L_2} 2^{l-1} H(\boldsymbol{O}^2_{l_{TFNet}}) \\ f_{i_{TFNet}} &= [Bhist(\boldsymbol{I}^1_{i_{TFNet}}), \ldots, Bhist(\boldsymbol{I}^{L_1}_{i_{TFNet}})]^T \in \mathbb{R}^{(2^{L_2})L_1 B} \end{aligned} \tag{15}$$

Despite having a close resemblance between the feature extraction mechanism of the PCANet and the TFNet, these two networks capture visibly distinguishable features from the two views of the images, as shown in Fig. 1. These plots are obtained by convolving an image of a *cat* with the first layer's convolution filters of the two networks.

Each of the L_1 convolution responses within the PCANet are visibly distinct, whereas the convolution responses within the TFNet shows visual similarity (the images in a triplet sequence show similarity consecutively). These plots demonstrate that the TFNet emphasizes mining the *common* information from the minutiae view of the data. Whereas the PCANet emphasizes mining the *unique* information from the amalgamated view of the data. Integrating both these kinds of information motivates the development of *HybridNet* explained in the next section.



(a) Convolution responses from the PCANet where each convolution response is visually distinct from the rest of the responses. This illustrates extraction of unique information with the amalgamated view of the data.

(b) Convolution responses from our TFNet where the visual resemblance is observed in a sequence of three responses. This illustrates extraction of common information with minutiae view of the data.

Fig. 1: Comparison of convolution outputs from Layer1 in PCANet and TFNet on CIFAR-10 dataset. These plots demonstrate the contrast between the kinds of information obtained with the amalgamated and the minutiae view of the data.

IV. The Hybrid Network

The complementary information available with the two views of the data is essential but individually insufficient, and their integration can enhance the performance of classification systems. This motivates the development of *HybridNet*, which simultaneously extracts information from both views of the data. The details *HybridNet* are available in the subsections and its feature extraction procedure is illustrated in Fig. 2.

A. The First Layer

Similar to the previous networks, we collect all overlapping patches of size $k_1 \times k_2$ around each pixel from the image I_i. Importantly, the first layer of *HybridNet* consists of image-patches expressed both as tensors $\boldsymbol{\mathcal{X}} \in \mathbb{R}^{k_1 \times k_2 \times 3 \times N\tilde{m}\tilde{n}}$ and matrices $\boldsymbol{X} \in \mathbb{R}^{k_1 k_2 \times N\tilde{m}\tilde{n}}$ which are utilized for obtaining weights of convolution filters in the first layer of *HybridNet*.

This enables the first layer to learn superior filters as it simultaneously receives more information from both views of the data. The weights for the *pca*-filters are obtained as the principal-eigenvectors as $\boldsymbol{W}^1_{l_{PCA}} = mat_{k_1,k_2}(ql(\boldsymbol{X}\boldsymbol{X}^T))$, and the weights for convolution-tensor filters are obtained by utilizing *LoMOI* as $\boldsymbol{W}^1_{l_{TF}} = \boldsymbol{U}^{(1)}_{(:,i)} \otimes \boldsymbol{U}^{(2)}_{(:,j)} \otimes \boldsymbol{U}^{(3)}_{(:,k)}$. Furthermore, the output from this layer is obtained by convolving input images with a) the PCA-filters and b) the convolution-tensorial filters in Eq. 16. This injects more diversity to the output in succeeding layer of *HybridNet*.

$$\begin{aligned} \boldsymbol{I}^l_{i_{PCA}} &= \boldsymbol{I}_i * \boldsymbol{W}^1_{l_{PCA}} \\ \boldsymbol{I}^l_{i_{TF}} &= \boldsymbol{I}_i * \boldsymbol{W}^1_{l_{TF}} \end{aligned} \tag{16}$$

Since we obtain of L_1 *pca* filters and L_1 convolution-tensor filters, a total of $2 \times L_1$ outputs are obtained in this layer.

B. The Second Layer

Similar to the above, we collect all input image patches; however, contrary to the previous layer, the weights of the *pca*-filters $\boldsymbol{W}^2_{l_{PCA}}$ and convolution-tensor filters $\boldsymbol{W}^2_{l_{TF}}$ are learned from the convolvtion output of input images with 1) the *pca* filters and 2) the convolution-tensor filters from the first layer. Hence both the patch-matrix $\boldsymbol{Y} \in \mathbb{R}^{k_1 k_2 \times 2L_1 N\tilde{m}\tilde{n}}$ and the patch-tensor $\boldsymbol{\mathcal{Y}} \in \mathbb{R}^{k_1 \times k_2 \times 2L_1 N\tilde{m}\tilde{n}}$ contain image patches obtained from $[\boldsymbol{I}_{i_{PCA}}, \boldsymbol{I}_{i_{TF}}]$. This enables the hybrid filters to assimilate more variability present in the data while obtaining their convolution filters.

Algorithm 2: The *HybridNet* Algorithm

Input: $\boldsymbol{I}_i, i = 1, 2, ..., n$ n is the total number of training images, $L = [l_1, l_2, ...l_D]$ the number of filters in each layer,k_1 and k_2 the patch-size, B, D = the depth of the network.
for $i = 1, 2, ..., n$ **do** ▷ DO for each image in the first convolution layer.
  $\boldsymbol{X} \leftarrow$ extract patches of size $k_1 \times k_2$ around each pixel of I_i ▷ mean centered and vectorized.
  $\boldsymbol{\mathcal{X}} \leftarrow$ extract patches of size $k_1 \times k_2$ around each pixel of I_i ▷ mean centred but retain their spatial shape.
$\boldsymbol{W}_{PCA} \leftarrow$ obtain PCA filters by factorizing $\boldsymbol{X}$.
$\boldsymbol{W}_{TF} \leftarrow$ obtain tensor filters by factorizing $\boldsymbol{X}$ with LoMOI Algo. 1.
for $i = 1, 2, ..., n$ **do**
  $\boldsymbol{I}^1_{i_{PCA}} \leftarrow \boldsymbol{I}_i * \boldsymbol{W}_{PCA}$ ▷ store convolution with pca filters.
  $\boldsymbol{I}^1_{i_{TF}} \leftarrow \boldsymbol{I}_i * \boldsymbol{W}_{TF}$ ▷ store convolution with tensorial filters.
for $l = 2, ..., D$ **do** ▷ DO for the remaining convolution layers.
  $\boldsymbol{I} \leftarrow [\boldsymbol{I}^{(l-1)}_{PCA}, \boldsymbol{I}^{(l-1)}_{TF}]$ ▷ Utilize both views together.
  for $i = 1, 2, ..., \hat{n}$ **do** ▷ $\hat{n} = 2 \times n \times l_{l-1}$.
    $\boldsymbol{X} \leftarrow$ extract patches of size $k_1 \times k_2$ around each pixel of $\boldsymbol{I}_i$
    $\boldsymbol{\mathcal{X}} \leftarrow$ extract patches of size $k_1 \times k_2$ around each pixel of $\boldsymbol{I}_i$
  $\boldsymbol{W}^l_{PCA} \leftarrow$ obtain PCA filters by factorizing $\boldsymbol{X}$.
  $\boldsymbol{W}^l_{TF} \leftarrow$ obtain tensor filters by factorizing $\boldsymbol{X}$ with LoMOI Alg. 1.
  for $i = 1, 2, ..., \bar{n}$ **do** ▷ $\bar{n} = n \times l_{l-1}$.
    $\boldsymbol{I}^l_{i_{PCA}} = \boldsymbol{I}^{(l-1)}_{i_{PCA}} * \boldsymbol{W}^1_{l_{PCA}}$
    $\boldsymbol{I}^l_{i_{TF}} = \boldsymbol{I}^{l-1}_{i_{TF}} * \boldsymbol{W}^1_{l_{TF}}$
for $i = 1, 2, ..., n$ **do** ▷ DO for each image.
  $\boldsymbol{I}_{i_{PCA}} = \sum_{k=1}^{l_d} 2^{l-1} H(\boldsymbol{I}^d_{k_{PCA}})$ ▷ Binarize and accumulate output from all convolution layers.
  $\boldsymbol{I}_{i_{TF}} = \sum_{k=1}^{l_d} 2^{l-1} H(\boldsymbol{I}^d_{k_{TF}})$
  $f_{i_{PCA}} \leftarrow Bhist(\boldsymbol{I}_{i_{TF}})$ ▷ create block-wise histogram.
  $f_{i_{TF}} \leftarrow Bhist(\boldsymbol{I}_{i_{TF}})$
Output: features from the amalgamated mode as f_{PCA}, and the minutiae mode as f_{TF}.

In the second layer, the weights of *pca* filters are obtained by *principal components* as and the weights for convolution-tensor filters are obtained as $\boldsymbol{W}^2_{l_{TF}}$. Analogous to the previous networks, the output images in this layer are obtained by a) convolving the L_1 images corresponding to the output from the PCA-filters in the first layer with the L_2 *pca* filters obtained in the second layer (Eq. 17), and b) convolving the L_1 images corresponding to the output from the convolution-tensorial filters in the first layer with the L_2 convolution-tensorial filters obtained in the second layer (Eq. 18). This results in a total of $2 \times L_1 \times L_2$ output images in this layer.

$$\boldsymbol{O}^l_{i_{PCA}} = \boldsymbol{I}^l_{i_{PCA}} * \boldsymbol{W}^2_{l_{PCA}} \tag{17}$$

$$\boldsymbol{O}^l_{i_{TF}} = \boldsymbol{I}^l_{i_{TF}} * \boldsymbol{W}^2_{l_{TF}} \tag{18}$$

The output images obtained from the *pca*-filters ($\boldsymbol{O}^l_{i_{PCA}}$) are then processed with the output layer to obtain $f_{i_{PCA}}$ as the information from the amalgamated view and the output images obtained from the convolution-tensor filters ($\boldsymbol{O}^l_{i_{TF}}$)

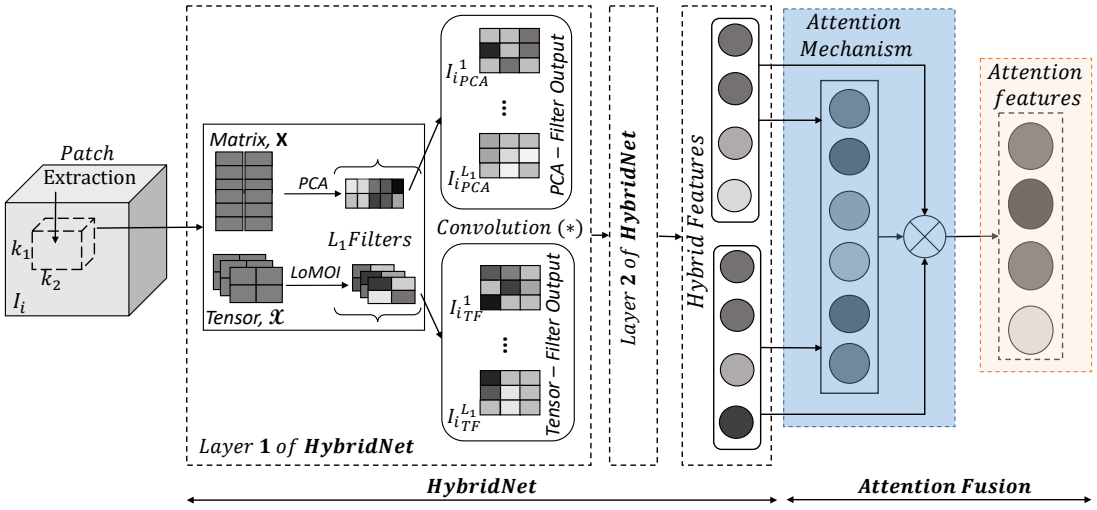


Fig. 2: Workflow of the proposed *Attn-HybridNet* model.

are processed to obtain $f_{i_{TF}}$ as the information from minutiae view of the images. Finally, these two kinds of information are concatenated to obtain the hybrid features; the process of obtaining hybrid features is described in Alg. 2.

$$f_{i_{hybrid}} = [f_{i_{PCA}} \ f_{i_{TF}}] \in \mathbb{R}^{(2^{L_2})2L_1B} \tag{19}$$

Although these hybrid features bring the best of the two views, they still suffer from feature redundancy arising from spatial pooling in the output layer and is alleviated by proposed *Attn-HybridNet* described in the next section.

V. Attention-based fusion - Attn-HybridNet

Our proposed *HybridNet* eradicates the loss of information by integrating the two views of the data, but the hybrid features incorporate feature redundancy due to the encoding scheme in the output layer [11], [26]. Moreover, the generalized spatial pooling operation is unable to accommodate the spatial structure of the natural images, i.e., it is more effective for aligned images dataset like face and handwritten digits than for object dataset. Simply, the output layer's design exacerbates performance degradation on object recognition datasets.

A few numerical optimization based techniques proposed in [11], [12] exist for alleviating the feature redundancy from architectures utilizing generalized spatial pooling layers. However, these techniques require grid search between the dictionary size (number of convolution filters in our case) and the pooling blocks in the output layer while performing optimization. Besides, the transition to prune filters from a single-layer networks to multi-layer network is not smooth in these techniques. A major difference between our proposed *Attn-HybridNet* and the existing proposal in [11], [12] is that we reduce the feature redundancy by performing feature selection with attention-based fusion scheme, whereas the existing techniques prune the filters to eliminate the feature redundancy. Therefore, our proposed *Attn-HybridNet* is superior to these existing techniques as it decouples the two sub-processes, i.e., information discovery with convolution layers and feature aggregation in the pooling layer while alleviating the redundancy exhibiting in the feature representations. Our proposed attention-based fusion scheme is presented in Alg. 3

Algorithm 3: The *Attn-HybridNet* Algorithm

Input: $f_{hybrid} = [f_{PCA};\ f_{TF}] \in \mathbb{R}^{N\times(2^{L_2})L_1B\times 2}$ the hybrid feature vectors from the training images; $y = [0, 1, ..., C]$ ground truth of training images, dimensionality of feature level context vector $w \in \mathbb{R}^d$, where $d << \mathbb{R}^{(2^{L_2})L_1B}$.
randomly initialize $\boldsymbol{W}$, f_c, and w
$loss \leftarrow 1000$ ▷ arbitrary number to start training
do
  $[f_{batch}, y_{batch}] \leftarrow$ sample batch ($[f_{\textbf{hybrid}}, y]$)
  $\boldsymbol{P}_F \leftarrow tanh(W.f_{batch})$ ▷ get the hidden representation of the hybrid features
  $\alpha = softmax(w^T.\boldsymbol{P}_F)$ ▷ measure and normalize the importance
  $F_{attn} = f_{batch}.\alpha^T$ ▷ perform attention fusion
  $\hat{y} \leftarrow f_c(F_{attn})$ ▷ fully connected layer
  $loss \leftarrow LogLoss(y_{batch}, \hat{y}_{batch})$ ▷ compute loss for optimizing parameters
  back-propagate loss for optimizing $\boldsymbol{W}$, f_c, and w.
while $[(loss \geq \varepsilon)]$ ▷ loop until convergence
Output: parameters to perform attention fusion $\boldsymbol{W}$, f_c, and $w \in \mathbb{R}^d$

The discriminative features obtained by *Attn-HybridNet* are utilized with $softmax$-layer for classification, where the parameters in the proposed fusion scheme (i.e., $\boldsymbol{W}$, f_c and w) are optimized via gradient-descent on the classification loss. This simple yet effective scheme substantially enhances the performance by obtaining highly discriminative features whose superiority is demonstrated in Sec. VI.

A. Computational Complexity

To calculate the computational complexity of *Attn-HybridNet*, we assume the *HybridNet* is composed of two-layers with a patch size of $k_1 = k_2 = k$ in each layer followed by our attention-based fusion scheme.

In each layer of the *HybridNet*, we compute the time complexities required to compute the convolution weights of the network. It includes, identical complexities of $k^2(1 + \tilde{m}\tilde{n})$ while forming zero-centered patch-matrix $\boldsymbol{X}$ and zero-centered patch-tensor $\mathcal{X}$. The complexity of eigen-decomposition for patch-matrix and patch-tensor factorization with *LoMOI* are also identical and equal to $\mathcal{O}((k^2)^3)$, where k is a whole number < 7 in our experiments. Further, the

complexity for convolution operations at stage i requires $L_i k^2 mn$ flops and the conversion of L_2 binary bits to a decimal number in the output layer costs $2L_2\tilde{m}\tilde{n}$, where $\tilde{m} = m - \lceil \frac{k}{2} \rceil, \tilde{n} = n - \lceil \frac{k}{2} \rceil$ and the naive histogram computation has complexity equal to $\mathcal{O}(mnBL_2log2)$.

The complexity of performing matrix multiplication in *Attn-HybridNet* is $\mathcal{O}\Big(2L_1B\big(d(1+2^{L_2})+2^{L_2}\big)\Big)$ which can be efficiently handled with modern deep learning packages like Tensorflow [27] for stochastic updates. To optimize the parameters in the attention-based fusion scheme ($\boldsymbol{W}$, f_c, and w), we back-propagate the loss through the attention network until convergence of the error on the training features.

VI. Experiments and Results

A. Experimental Setup

In our experiments, we fixed the depth of the networks as two and optimized the number of convolution filters via cross-validation. We utilized *Linear-SVM* [28] as the classifier incorporating features obtained with the PCANet, TFNet, and the *HybridNet*. Besides, the value of context level feature vector (w) in *Attn-HybridNet* was searched between $[10, 50, 100, 150, 200, 400]$ and is optimized via back-propagation. The obtained attention features i.e., F_{attn}, are then utilized with $softmax$-layer for classification. We observed that the attention-network's optimization took less than 15 epochs to converge on all the datasets utilized in this paper.

B. Datasets and Hyperparameters

The optimal hyperparameters on each dataset are as below:

1 MNIST variations dataset [29] consists of grayscale handwritten digits of size 28×28 with controlled factors of variations such as background noise and rotations. Each variations set consists of $10K$ training and $50K$ testing images. The network's parameters are set as $L_1, L2$ = [9, 8]; k_1, k_2 = [7, 7]. The overlapping block size region (B) was kept as [7, 7] during feature pooling.

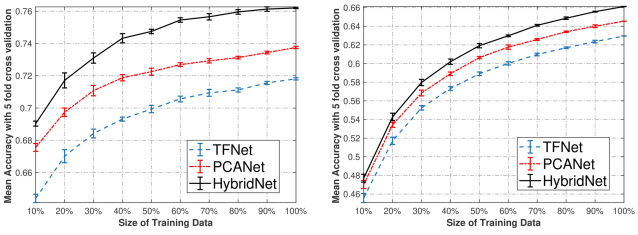
2 ORL[5] and Extended Yale-B [30] are utilized to investigate the performance of the networks on face recognition datasets. The ORL dataset consists of 10 different images of 40 distinct subjects taken by varying the lighting, facial expression, and facial details. The Extended Yale-B dataset consists of face images from 38 individuals under 9 poses and 64 illumination conditions. The images in both the datasets are cropped to size 64×64 pixels followed by unit length normalization. The classification results are averaged over 5 different trails by progressively increasing the number of training examples[6] with hyperparameters as $L_1, L2$ = [9, 8]; k_1, k_2 = [5, 5] with a non-overlapping block of size B as [7, 7].

3 CIFAR-10 [10] object recognition dataset consists of $50K$ training and $10K$ testing color images of size 32×32. These images are distributed among 10 classes and vary significantly in object positions, object scales, colors, and textures within each class. The number of filters is varied between [9, 27] for the first layer and kept as 8 in the second layer. The patch-size k_1, k_2 are kept equal and varied between [5, 7, 9] with a non-overlapping block size B as [8, 8]. Following [9] we also applied spatial pyramid pooling (SPP) [31] to the output layer of all the networks. We also applied PCA to reduce the dimension of pooled features to 100[7] and utilized them for classification.

4 Due to space limitations, we present the results on CIFAR-100 and CUReT dataset in the supplementary[8].

[5]Available online at: http://www.uk.research.att.com/facedatabase.html.

[6]The training and test data split are obtained from http://www.cad.zju.edu.cn/home/dengcai/Data/FaceData.html.



(a) CIFAR-10 (b) MNIST bg-img-rot

Fig. 3: Classification performance of PCANet, TFNet, and proposed HybridNet by varying size of the training data.

C. Baselines

On face recognition datasets, we compare the performance of *Attn-HybridNet* and *HybridNet* against three recent baselines: Deep-NMF [32], PCANet+ [33], and PCANet-II [34][9]. On MNIST variations dataset we select the baselines as in [9].

On CIFAR-10 dataset, we compare our schemes against comparable baselines such as CUDA-Convnet [35], VGG style CNN (VGG-CIFAR-10 reported by [36]), K-means (tri) [37], Spatial Pyramid Pooling for CNN (SPP) [38], Convolution Bag-of-Features (CBoF) [39], and Spatial-CBoF [3]. Note that we do not compare against baselines which aim to either compress deep neural networks or transfer pre-learned CNN filters such as in [40], [41], [42], [43]. Besides, we also report the performances of ResNet [19] and DenseNet [44] datasets, as mentioned in their respective publication.

VII. Results and Discussions

Our two main research contributions in this work are - 1) extracting information from the amalgamated view and the minutiae view in a consolidated framework, and 2) attention-based fusion of information obtained from these two views for supervised classification. We evaluate the significance of these contributions under the following research questions:

Q1: Is the integration of both the minutiae view and the amalgamated view beneficial? Or, does their integration deteriorate the generalization performance of HybridNet?

In order to evaluate this, we varied the size of training data in *HybridNet*, the PCANet, and TFNet and obtained their classification performance on CIFAR-10 and MNIST

[7]Results do not vary significantly on increasing the projection dimensions.

[8]Available online at http://bit.ly/2NxfcHe

[9]The paper did not provide its source code and the results are based on our independent implementation of their second order pooling technique.

TABLE II: Classification performance obtained by varying hyperparameters of PCANet, TFNet, and HybridNet.

Parameters				PCANet [9]	TFNet [18]	*HybridNet*	*Attn-HybridNet*
L_1	L_2	k_1	k_2	Error (%)	Error (%)	Error (%)	Error (%)
8	8	5	5	34.80	32.57	31.39	**28.08**
8	8	7	7	39.92	37.19	35.24	**30.94**
8	8	9	9	43.91	39.65	38.04	**35.33**
27	8	5	5	26.43	29.25	23.84	**18.41**
27	8	7	7	30.08	32.57	28.53	**25.67**
27	8	9	9	33.94	34.79	31.36	**27.70**

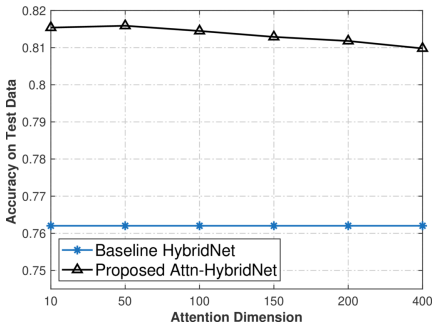


Fig. 4: Accuracy of Attn-HybridNet on CIFAR-10 dataset by varying the dimension of w in Alg. 3.

variations datasets. We cross-validated these schemes for 5 times and present the performances in Fig. 3.

Firstly, these plots suggest that the classification accuracies obtained with the features from *HybridNet* (and from the PCANet and the TFNet) linearly increase with respect to the size of training data. Secondly, these plots also demonstrate that the information obtained from the amalgamated view in PCANet is superior than the information obtained from the minutiae view TFNet on the object-recognition dataset. However, these two kinds of information achieve competitive classification performance on handwritten digits dataset, which contains nearly aligned images. Most importantly, these plots unambiguously demonstrate that integrating both kinds of information enhance the superiority of feature representations in the proposed *HybridNet*.

Q2: How does the hyperparameters affect the discriminability of feature representations? Moreover, how does these affect the performance of HybridNet and Attn-HybridNet?

To address this question, we present a detailed study on how the hyperparameters affect the performance of *HybridNet* and *Attn-HybridNet* and present their classification performance on CIFAR-10 dataset in Table II. The minimum error achieved for each combination of hyperparameters (in each row) is highlighted in bold font. Moreover, we also illustrate the performance of *Attn-HybridNet* by varying the dimension of context level feature vector w in Fig. 4.

A clear trend is visible among all the networks in Table II, where the classification error decreases with increasing the number of filters in the first layer of the networks. It also demonstrates the effect of the factorization rank in the first layer of the networks, signifying an increase in the data variability among subsequent layers. However, this increases the dimensionality of the features vectors but suggests that higher-dimensional features have lower intraclass variability among the objects from the same category.

TABLE III: Classification performance on the ORL dataset.

ORL - Dataset	Number of Training Instances		
	4	6	8
Deep-NMF [32]	9.50 ± 1.94	6.50 ± 2.59	1.75 ± 2.09
PCANet-II [34]	16.16 ± 1.29	7.87 ± 1.29	5.00 ± 2.05
PCANet+ [33]	1.25 ± 0.83	0.50 ± 0.52	0.25 ± 0.55
PCANet [9]	1.75 ± 0.95	0.37 ± 0.34	0.40 ± 0.68
TFNet [18]	1.98 ± 0.54	0.50 ± 0.68	0.25 ± 0.59
HybridNet (proposed)	**1.48 ± 0.72**	**0.25 ± 0.32**	**0.21 ± 0.55**
Attn-HybridNet (proposed)	5.43 ± 0.78	3.11 ± 0.27	1.13 ± 0.31

TABLE IV: Classification performance on the YaleB dataset.

YaleB - Dataset	Number of Training Instances			
	20	30	40	50
Deep-NMF [32]	10.94±0.89	8.03±0.61	5.43±0.94	4.78±0.76
PCANet-II [34]	11.40±0.97	5.54±1.49	2.86±0.35	1.98±0.75
PCANet+ [33]	1.15±0.14	0.28±0.07	0.23±0.14	0.22±0.23
PCANet [9]	1.35±0.17	0.40±0.18	0.38±0.16	0.38±0.13
TFNet [18]	1.97±0.27	0.91±0.28	0.40±0.16	0.42±0.21
HybridNet (proposed)	**1.32±0.35**	**0.55±0.26**	**0.32±0.25**	**0.34±0.21**
Attn-HybridNet (proposed)	5.11±0.65	2.80±0.42	2.12±0.15	1.88±0.40

Another trend is observable, where the classification error increases by increasing the size of the image patches. It might be due to a gradual increase of non-stationarity in data as a larger patch size might contain more background and noise than the actual object in the image patch [9].

Q3: How does the proposed Attn-HybridNet (and HybridNet) perform in comparison to the baseline techniques?

To evaluate this requirement, we compare the performance of our proposed techniques against baselines as detailed in Sec. VI-C. In this regard, we present the performance comparison on face recognition datasets in Table III and Table IV. The performance comparison on handwritten digits and CIFAR-10 dataset in Table V and Table VI, respectively.

We further perform a comparative study of the baselines and our proposed schemes by varying the size of training data and analyzing their classification performances on the CIFAR-10 dataset in Fig. 5. Besides, we perform a qualitative visualization of the feature representation obtained from *HybridNet* and *Attn-HybridNet* with a t-SNE [45] plot in Fig. 6.

a) Performance on face recognition: A similar trend is noticeable among the classification performances on ORL and Extended YaleB datasets, where the classification error decreases with the increase of the number of training examples. This is justifiable as by increasing the amount of training data, all schemes can better estimate the variation in lighting, facial expression, and pose. Secondly, among the baselines, PCANet+ performs substantially better than Deep-NMF and PCANet-II on the face datasets; and performs slightly better than PCANet as it has a better feature encoding scheme. The poor performance for Deep-NMF might be due to less amount of training data available while estimating its parameters. Similarly, in PCANet-II, the requirement for explicit alignment

TABLE V: Classification performance on MNIST variations datasets.

Methods	baisc	bg-rand	bg-img	bg-img-rot	rect-image
CAE-2 [46]	2.48	10.90	15.50	45.23	21.54
PGBM [47]	-	6.08	12.25	36.76	**8.02**
ScatNet-2 [48]	1.27	12.30	18.40	50.48	15.94
PCANet [9]	1.07	6.99	11.16	35.46	13.59
TFNet [18]	1.07	6.96	11.44	37.02	16.87
HybridNet (proposed)	1.01	5.46	10.08	33.87	12.91
Attn-HybridNet (proposed)	**0.94**	**3.73**	**8.68**	**31.33**	10.65

of images can explain its degradation in performance.

Our proposed *HybridNet* outperforms all the baselines on both the datasets validating our hypothesis that utilizing both kinds of information can enhance the classification performance. However, the classification performance of *Attn-HybridNet* is slightly worse than *HybridNet*, which might be due to less amount of data available while learning the attention parameters. However, it still performs better than Deep-NMF as the underlying features are highly discriminative, and therefore it is less strenuous to discover the attention weights in the proposed fusion scheme than in Deep-NMF.

b) Performance on digit recognition: On MNIST variations dataset, the *Attn-HybridNet* outperforms the baselines on four out of five variations. In particular, for *bg-rand* and *bg-img* variations, we decreased the error (compared to [18]) by **31.68**% and **13.80**% respectively. Also on variation *bg-img-rot* we decreased the error by **07.498**%.

c) Quantitative Performance on CIFAR-10 Dataset: We compare the performance of proposed *Attn-HybridNet* and *HybridNet* against the baselines on the CIFAR-10 dataset and report their respective accuracies in Table VI.

The proposed *Attn-HybridNet* achieves the best performance among PCANet, TFNet, and *HybridNet* on the CIFAR-10 dataset and reduces the error by 22.78% compared to the *HybridNet*. Moreover, it achieves 16.70% lower error compared to baseline K-means (tri), which has $2\times$ higher feature dimensionality than our proposed *HybridNet* and utilizes L_2 regularized-SVM instead of $Linear$-SVM for classification. Besides, the proposed *HybridNet* reduces the error by 9.80% on the CIFAR-10 dataset compared to the PCANet.

Additionally, the performance of our proposed *Attn-HybridNet* is still better than VGG-CIFAR-10 [36] and comparable to CUDA-Convnet [35][10], both of which have more depth than *Attn-HybridNet*. In particular, we have reduced the error by 1.63% than VGG-CIFAR-10 with 99.63% less trainable parameters. At the same time, we have performed very competitive to CUDA-Convnet, achieving a marginal 0.41% higher error rate but with 88% less hyperparameters.

Our proposed *Attn-HybridNet* also performs marginally better in compared to deep-quantized networks such as SPP [38], CBoF [39], and Spatial-CBoF [3]. This is because our proposed scheme is a decoupled technique i.e., it has independent feature extraction and pooling schemes, and hence the effort required to estimate the attention parameters is negligible.

[10]We cite the accuracy as published.

TABLE VI: Classification performance on the CIFAR-10 dataset with no data augmentation. The DenseNet achieves the lowest classification error but at the expense of huge depth and substantial computational cost among all techniques.

Methods	**#Depth**	**#Params**	**Error**
K-means (tri.) [37] (1600 dim.)	1	5	22.10
CUDA-Convnet [35]	4	1.06M	18.00
VGG-CIFAR-10 [49]	5	2.07M	20.04
SPP [38]	5	256.5K	19.39
CBoF [39]	5	174.6K	20.47
Spatial-CBoF [3]	5	199.1K	21.37
ResNet reported in-[50]	110	1.7M	13.63
DenseNet-BC reported in-[44]	250	15.3M	**5.2**
PCANet [9]	3	7	26.43
TFNet [18]	3	7	29.25
HybridNet (proposed)	3	7	23.84
Attn-HybridNet (proposed)	3	12.7k	18.41

Besides, the performance gap between *Attn-HybridNet* and state of the art ResNet and DenseNet are not comparable as the depth and the computational complexity of these networks are tremendously huge. In addition, the main bottleneck for these schemes is high computational complexity, which is opposite to the motivation of this work that is the alleviation of such requirements and hence the tradeoff.

d) Qualitative Discussion on CIFAR-10: We now present a qualitative discussion on the performances of various baselines and our proposals by varying the size of the CIFAR-10 training dataset. Although our proposed *Attn-HybridNet* consistently achieved the highest classification performance, a few interesting patterns are noticeable in Fig. 5.An important observation in this regard is the lower classification performance achieved by both CUDA-Convnet [35] and VGG-CIFAR-10 [49] with less amount of training dataset, particularly until 40%. This is justifiable and intuitive since less amount of the training data is insufficient to learn the parameters of these deep networks. However, on increasing the amount of training data, in particular, above 50%, the performance of these networks increases substantially. Simply, it increases with a larger margin compared to the performance of SVM based schemes in *HybridNet* and K-means (tri) [37].

Another observation in this regard is about the classification performances of *HybridNet* and K-means (tri). While both these networks achieve higher classification accuracy compared to the deep networks with significantly less amount of training data (only 10% of the training dataset); particularly, the *HybridNet* has **11.56**$\%$ higher classification rate compared to the second-highest classification accuracy achieved by K-means (tri). The accuracy of these networks does not scale with an increase in the training data, as noticed by deep-network-based schemes.

Finally, since the *Attn-HybridNet* achieved the highest classification performance across different sizes of the training dataset. A possible explanation for this might be the requirement of fewer parameters with proposed attention-fusion while performing feature selection and fusion of the hybrid features. Moreover, the t-SNE plot in Fig. 6 compares the discriminability of features obtained with the *HybridNet* and

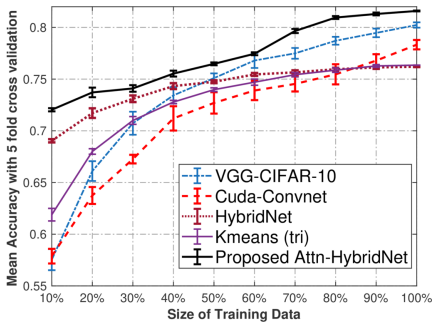
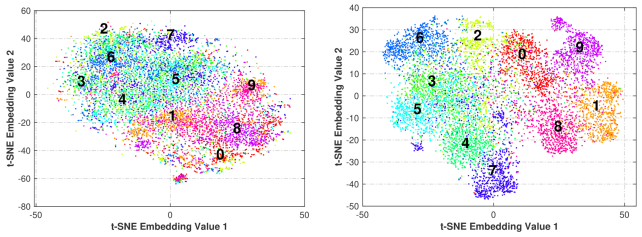


Fig. 5: (Best viewed in color) Accuracy of various methods on CIFAR-10 dataset by varying size of the training data



(a) *HybridNet* (b) *Attn-HybridNet*

Fig. 6: (Best viewed in color) t-SNE visualization of features from *HybridNet* and *Attn-HybridNet* on CIFAR-10 dataset.

Attn-HybridNet. The plot on the features obtained from *Attn-HybridNet* Fig. 6(b) visually achieves better clustering than the plot on features obtained from *HybridNet* Fig. 6(a) and justifies the performance improvement with proposed *Attn-HybridNet*.

VIII. Conclusion and Future Work

In this paper, we have introduced *HybridNet*, which integrates the information discovery procedure from the amalgamated view and the minutiae view of the data. The development of *HybridNet* is motivated by the fact that information obtained from the two views of the data are individually insufficient but necessary for classification. In this regard, we first customized a factorization scheme to obtain information from the minutiae view of the data and called it as *LoMOI*. We then demonstrated that the information obtained with the two views of data are complementary to each other and provided details of *HybridNet*, which simultaneously extracts information from the two views of the data. We then proposed an attention-based fusion scheme to alleviate the feature redundancy among hybrid features as the *Attn-HybridNet*.

We performed comprehensive experiments on multiple real-world datasets to validate the significance of our proposed techniques. The features extracted with *Attn-HybridNet* achieved similar classification performance but required significantly less amount of hyperparameters and training time required among baselines methods. Besides, we also performed case studies to provide qualitative insights regarding the superiority of features extracted by proposed *Attn-HybridNet*.

Furthermore, our research can be further improved with two interesting research directions. The first direction is regarding the design of *HybridNet* filters to accommodate various non-linearities in the data, such as alignments and occlusion. A second research direction can be generalizing the fusion scheme for tasks such as face verification and gait recognition.

References

[1] Y. Bengio, A. Courville, and P. Vincent, "Representation learning: a review and new perspectives," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 35, no. 8, pp. 1798–1828, 2013.

[2] X. Glorot and Y. Bengio, "Understanding the difficulty of training deep feedforward neural networks," in *Proceedings of the thirteenth international conference on artificial intelligence and statistics*, 2010, pp. 249–256.

[3] N. Passalis and A. Tefas, "Training lightweight deep convolutional neural networks using bag-of-features pooling," *IEEE transactions on neural networks and learning systems*, 2018.

[4] S. Han, H. Mao, and W. J. Dally, "Deep compression: compressing deep neural networks with pruning, trained quantization and huffman coding," *ICLR*, 2016.

[5] X. Zhang, J. Zou, X. Ming, K. He, and J. Sun, "Efficient and accurate approximations of nonlinear convolutional networks," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2015, pp. 1984–1992.

[6] V. Lebedev, Y. Ganin, M. Rakhuba, I. Oseledets, and V. Lempitsky, "Speeding-up convolutional neural networks using fine-tuned cp-decomposition," in *ICLR*, 2015.

[7] J. Cheng, J. Wu, C. Leng, Y. Wang, and Q. Hu, "Quantized cnn: a unified approach to accelerate and compress convolutional networks," *IEEE transactions on neural networks and learning systems*, no. 99, pp. 1–14, 2017.

[8] J. Kossaifi, A. Khanna, Z. Lipton, T. Furlanello, and A. Anandkumar, "Tensor contraction layers for parsimonious deep nets," in *Computer Vision and Pattern Recognition Workshops (CVPRW), 2017 IEEE Conference on*. IEEE, 2017, pp. 1940–1946.

[9] T.-H. Chan, K. Jia, S. Gao, J. Lu, Z. Zeng, and Y. Ma, "Pcanet: a simple deep learning baseline for image classification?" *IEEE Transactions on Image Processing*, vol. 24, no. 12, pp. 5017–5032, 2015.

[10] A. Krizhevsky and G. Hinton, "Learning multiple layers of features from tiny images," Master's thesis, Department of Computer Science, University of Toronto, 2009.

[11] Y. Jia, O. Vinyals, and T. Darrell, "On compact codes for spatially pooled features," in *International Conference on Machine Learning*, 2013, pp. 549–557.

[12] Y. Jia, C. Huang, and T. Darrell, "Beyond spatial pyramids: Receptive field learning for pooled image features," in *2012 IEEE Conference on Computer Vision and Pattern Recognition*. IEEE, 2012, pp. 3370–3377.

[13] A. Braytee, W. Liu, and P. Kennedy, "A cost-sensitive learning strategy for feature extraction from imbalanced data," in *International conference on neural information processing*, 2016, pp. 78–86.

[14] S. Verma, W. Liu, C. Wang, and L. Zhu, "Extracting highly effective features for supervised learning via simultaneous tensor factorization," in *Thirty-First AAAI Conference on Artificial Intelligence*, 2017.

[15] J.-T. Chien and Y.-T. Bao, "Tensor-factorized neural networks," *IEEE transactions on neural networks and learning systems*, vol. 29, no. 5, pp. 1998–2011, 2017.

[16] S. Verma, C. Wang, L. Zhu, and W. Liu, "Deepcu: Integrating both common and unique latent information for multimodal sentiment analysis." in *IJCAI*, 2019, pp. 3627–3634.

[17] W. Liu, J. Chan, J. Bailey, C. Leckie, and K. Ramamohanarao, "Mining labelled tensors by discovering both their common and discriminative subspaces," in *Proceedings of the 2013 SIAM International Conference on Data Mining*. SIAM, 2013, pp. 614–622.

[18] S. Verma, W. Liu, C. Wang, and L. Zhu, "Hybrid networks: Improving deep learning networks via integrating two views of images," in *Neural Information Processing - 25th International Conference, ICONIP , Proceedings, Part I*, 2018, pp. 46–58.

[19] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2016, pp. 770–778.

[20] W. Chen, J. Wilson, S. Tyree, K. Weinberger, and Y. Chen, "Compressing neural networks with the hashing trick," in *International Conference on Machine Learning*, 2015, pp. 2285–2294.

[21] W. Liu and S. Chawla, "A quadratic mean based supervised learning model for managing data skewness," in *Proceedings of the 2011 SIAM International Conference on Data Mining*. SIAM, 2011, pp. 188–198.
[22] A. Cichocki, D. Mandic, L. De Lathauwer, G. Zhou, Q. Zhao, C. Caiafa, and H. A. Phan, "Tensor decompositions for signal processing applications: from two-way to multiway component analysis," *IEEE Signal Processing Magazine*, vol. 32, no. 2, pp. 145–163, 2015.
[23] M. A. O. Vasilescu and D. Terzopoulos, "Multilinear analysis of image ensembles: Tensorfaces," in *European Conference on Computer Vision*. Springer, 2002, pp. 447–460.
[24] B. Savas and L. Eldén, "Handwritten digit classification using higher order singular value decomposition," *Pattern Recognition*, vol. 40, no. 3, pp. 993–1003, 2007.
[25] F. Cong, Q.-H. Lin, L.-D. Kuang, X.-F. Gong, P. Astikainen, and T. Ristaniemi, "Tensor decomposition of eeg signals: a brief review," *Journal of Neuroscience Methods*, vol. 248, pp. 59–69, 2015.
[26] A. Coates and A. Y. Ng, "The importance of encoding versus training with sparse coding and vector quantization," in *Proceedings of the 28th international conference on machine learning (ICML-11)*, 2011, pp. 921–928.
[27] M. Abadi, P. Barham, J. Chen, Z. Chen, A. Davis, J. Dean, M. Devin, S. Ghemawat, G. Irving, M. Isard *et al.*, "Tensorflow: A system for large-scale machine learning," in *12th {USENIX} Symposium on Operating Systems Design and Implementation ({OSDI} 16)*, 2016, pp. 265–283.
[28] R.-E. Fan, K.-W. Chang, C.-J. Hsieh, X.-R. Wang, and C.-J. Lin, "Liblinear: a library for large linear classification," *Journal of Machine Learning Research*, vol. 9, no. Aug, pp. 1871–1874, 2008.
[29] H. Larochelle, D. Erhan, A. Courville, J. Bergstra, and Y. Bengio, "An empirical evaluation of deep architectures on problems with many factors of variation," in *Proceedings of the 24th International Conference on Machine Learning*. ACM, 2007, pp. 473–480.
[30] A. S. Georghiades, P. N. Belhumeur, and D. J. Kriegman, "From few to many: Illumination cone models for face recognition under variable lighting and pose," *IEEE transactions on pattern analysis and machine intelligence*, vol. 23, no. 6, pp. 643–660, 2001.
[31] K. Grauman and T. Darrell, "The pyramid match kernel: discriminative classification with sets of image features," in *Computer Vision, 2005. ICCV 2005. Tenth IEEE International Conference on*, vol. 2. IEEE, 2005, pp. 1458–1465.
[32] G. Trigeorgis, K. Bousmalis, S. Zafeiriou, and B. W. Schuller, "A deep matrix factorization method for learning attribute representations," *IEEE transactions on pattern analysis and machine intelligence*, vol. 39, no. 3, pp. 417–429, 2016.
[33] C.-Y. Low, A. B.-J. Teoh, and K.-A. Toh, "Stacking pcanet+: An overly simplified convnets baseline for face recognition," *IEEE Signal Processing Letters*, vol. 24, no. 11, pp. 1581–1585, 2017.
[34] C. Fan, X. Hong, L. Tian, Y. Ming, M. Pietikäinen, and G. Zhao, "Pcanet-ii: When pcanet meets the second order pooling," *IEICE Transactions on Information and Systems*, vol. 101, no. 8, pp. 2159–2162, 2018.
[35] A. Krizhevsky. (2012 (accessed May 20, 2019)) Cuda convnet. [Online]. Available: https://code.google.com/archive/p/cuda-convnet/
[36] S. Chintala, *VGG Style CNN on CIFAR10*, 2017 (accessed September 3, 2017), https://github.com/soumith/DeepLearningFrameworks.
[37] A. Coates, A. Ng, and H. Lee, "An analysis of single-layer networks in unsupervised feature learning," in *Proceedings of the Fourteenth International Conference on Artificial Intelligence and Statistics*, 2011, pp. 215–223.
[38] K. He, X. Zhang, S. Ren, and J. Sun, "Spatial pyramid pooling in deep convolutional networks for visual recognition," *IEEE transactions on pattern analysis and machine intelligence*, vol. 37, no. 9, pp. 1904–1916, 2015.
[39] N. Passalis and A. Tefas, "Learning bag-of-features pooling for deep convolutional neural networks," in *Proceedings of the IEEE international conference on computer vision*, 2017, pp. 5755–5763.
[40] R. Keshari, M. Vatsa, R. Singh, and A. Noore, "Learning structure and strength of cnn filters for small sample size training," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2018, pp. 9349–9358.
[41] X. Yu, T. Liu, X. Wang, and D. Tao, "On compressing deep models by low rank and sparse decomposition," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2017, pp. 7370–7379.
[42] S. Lin, R. Ji, C. Chen, D. Tao, and J. Luo, "Holistic cnn compression via low-rank decomposition with knowledge transfer," *IEEE transactions on pattern analysis and machine intelligence*, vol. 41, no. 12, pp. 2889–2905, 2018.
[43] J.-H. Luo, J. Wu, and W. Lin, "Thinet: A filter level pruning method for deep neural network compression," in *Proceedings of the IEEE international conference on computer vision*, 2017, pp. 5058–5066.
[44] G. Huang, Z. Liu, L. Van Der Maaten, and K. Q. Weinberger, "Densely connected convolutional networks," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2017, pp. 4700–4708.
[45] L. v. d. Maaten and G. Hinton, "Visualizing data using t-sne," *Journal of machine learning research*, vol. 9, no. Nov, pp. 2579–2605, 2008.
[46] S. Rifai, P. Vincent, X. Muller, X. Glorot, and Y. Bengio, "Contractive auto-encoders: explicit invariance during feature extraction," in *Proceedings of the 28th International Conference on International Conference on Machine Learning*. Omnipress, 2011, pp. 833–840.
[47] K. Sohn, G. Zhou, C. Lee, and H. Lee, "Learning and selecting features jointly with point-wise gated boltzmann machines," in *International Conference on Machine Learning*, 2013, pp. 217–225.
[48] J. Bruna and S. Mallat, "Invariant scattering convolution networks," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 35, no. 8, pp. 1872–1886, 2013.
[49] I. Karmanov. (accessed May 20, 2019) Vgg style cnn on cifar10. [Online]. Available: https://github.com/soumith/DeepLearningFrameworks
[50] G. Huang, Y. Sun, Z. Liu, D. Sedra, and K. Q. Weinberger, "Deep networks with stochastic depth," in *European conference on computer vision*. Springer, 2016, pp. 646–661.

Sunny Verma received his Ph.D. degree in Computer Science from University of Technology Sydney in 2020. He is currently working as Postdoctoral Research Fellow at the Data Science Institute, University of Technology Sydney, and as a visiting scientist at Data61, CSIRO. Before joining UTS, he was a Research Assistant at Department of Electrical Engineering, IITD India, and then worked as Senior Research Assistant at Hong Kong Baptist University, Hong Kong. He obtained his Ph.D. from the University of Sydney. His research interests include data mining, FATE, and interpretable deep learning.

Chen Wang is a Principal Research Scientist with Data61, CSIRO. His research is in distributed and parallel computing with recent focus on data analytics systems and deep learning interpretability. He published more than 70 papers in major journals and conferences such as TPDS, TC, WWW, SIGMOD and HPDC. He has industrial experience. He developed a high-throughput event system and a medical 'image archive system used by many hospitals and medical centers in the USA.

Liming Zhu is a Research Director at Data61, CSIRO. He is also a conjoint full professor at University of New South Wales (UNSW). He is the chairperson of Standards Australia's blockchain and distributed ledger committee. His research program has more than 300 people innovating in the area of big data platforms, computational science, blockchain, regulation technology, privacy and cybersecurity. He has published more than 200 academic papers on software architecture, secure systems, data analytics infrastructure and blockchain.

Wei Liu (M'15-SM'20) is an Associate Professor and the Data Science Research Leader at the School of Computer Science, Faculty of Engineering and IT, University of Technology Sydney (UTS). Before joining UTS, he was a Research Fellow at the University of Melbourne and then a Machine Learning Researcher at NICTA. He obtained his PhD from the University of Sydney. He works in the areas of machine learning and data mining and has published more than 90 papers in Q1 journals and world-leading top conferences on topics of tensor factorization, game theory, adversarial learning, graph mining, causal inference, and anomaly detection. He has three best paper awards.