IEEE Access
Multidisciplinary | Rapid Review | Open Access Journal

Received March 20, 2019, accepted April 12, 2019, date of publication April 16, 2019, date of current version April 26, 2019.

Digital Object Identifier 10.1109/ACCESS.2019.2911622

On Learning and Learned Data Representation by Capsule Networks

ANCHENG LIN[1], (Student Member, IEEE), JUN LI[2], (Member, IEEE), AND ZHENYUAN MA[3], (Member, IEEE)

[1]School of Computer Sciences, Guangdong Polytechnic Normal University, Guangzhou 510665, China
[2]School of Software and Centre for Artificial Intelligence, Faculty of Engineering and Information Technology, University of Technology Sydney, Broadway, NSW 2007, Australia
[3]School of Mathematics and System Sciences, Guangdong Polytechnic Normal University, Guangzhou 510665, China

Corresponding author: Zhenyuan Ma (e-mail: mazy@gpnu.edu.cn).

This work was supported in part by the Science and Technology Program of Guangzhou under Grant 201704030133, "A Knowledge-Connection and Cognitive-Style Based Mining System for Massive Open Online Courses and Its Application" (UTS Project Code: PRO16-1300), in part by the China Natural Science Foundation of Guangdong under Grant 61877029, and in part by the Guangdong Innovation Education Program for Graduate under Grant 2016JGXM_MS_47.

ABSTRACT Capsule networks (CapsNet) are recently proposed neural network models containing newly introduced processing layer, which are specialized in entity representation and discovery in images. CapsNet is motivated by a view of parse tree-like information processing mechanism and employs an iterative routing operation dynamically determining connections between layers composed of capsule units, in which the information ascends through different levels of interpretations, from raw sensory observation to semantically meaningful entities represented by active capsules. The CapsNet architecture is plausible and has been proven to be effective in some image data processing tasks, the newly introduced routing operation is mainly required for determining the capsules' activation status during the forward pass. However, its influence on model fitting and the resulted representation is barely understood. In this work, we investigate the following: 1) how the routing affects the CapsNet model fitting; 2) how the representation using capsules helps discover global structures in data distribution, and; 3) how the learned data representation adapts and generalizes to new tasks. Our investigation yielded the results some of which have been mentioned in the original paper of CapsNet, they are: 1) the routing operation determines the certainty with which a layer of capsules pass information to the layer above and the appropriate level of certainty is related to the model fitness; 2) in a designed experiment using data with a known 2D structure, capsule representations enable a more meaningful 2D manifold embedding than neurons do in a standard convolutional neural network (CNN), and; 3) compared with neurons of the standard CNN, capsules of successive layers are less coupled and more adaptive to new data distribution.

INDEX TERMS Capsule network, deep neural network, interpretable learning, representation learning.

I. INTRODUCTION

Deep neural networks (DNNs) have achieved great success in image and video processing tasks [1], [2]. A widely accepted and empirically supported theory on the success of the DNN models states that deep architecture can help discover rich, hierarchical representations of data [3]. However, interpreting the intermediate levels of data representation and how those representations help the analytic task has not been completely understood [4]. Capsule Net (CapsNet) [5] is a recently proposed architecture motivated by a hypothesized information processing hierarchy, it represents an alternative arrangement of the multiple stage processing of image data. Essentially, CapsNet differs from the traditional deep neural networks because i) in each stage, the atomic units of information are vectors rather than scalar values[1] and ii) the output of a processing stage no longer contributes equally to the computation of its successive stage. One can intuitively understand the changes as the introduction of structures in

The associate editor coordinating the review of this manuscript and approving it for publication was Berdakh Abibullaev.

[1]To be more specific, the "atomic units" refer to the basic random variables that we are concerned with. In practical image/video analytic tasks, this concept of lower "convolutional" layers correspond to an element of a channel at a particular image location.

the information and information that no longer flows homogeneously through the processing pipeline. CapsNet attempts to use deep neural networks to construct a parse tree of the images representing the parts-belong-to-whole relationships.

Based on the above understanding, our study is presented based on three aspects of CapsNet, answering three fundamental questions regarding the effectiveness of the new architecture in terms of visual analytics, namely, model fitting, representation learning, and generalization.

- Model fitting: The effect of the routing process on the training of the network is investigated. Routing introduces additional dynamics in the information flow through a network. Different from conventional neural networks, where cross-layer connections are determined completely by network parameters, the connections between two successive layers of capsules in CapsNet are computed at run-time and vary with individual data samples. Our investigation shows that routing determines the (un-)certainty in the information pathway through layers of capsules. The appropriate level of certainty is closely related to the model fitness to data.
- Representation learning: One of the motivations leading to the development of CapsNet is that capsule representation can correspond to interpretable attributes of images, such as the style of writing in the case of handwritten digits. In this work, we test trained models on image data on a known 2D manifold spanned by geometric transformations. The ground-truth data manifold allows us to quantitatively assess the data representations in terms of a meaningful structure discovery. We found that, compared to standard neural networks, CapsNet captured the global manifold structure in the image data more faithfully.
- Representation generalization: if CapsNet could recover parse tree-like structures for images [5], the capsules would correspond to entities at different levels of interpretation. One can then expect the intermediate-level capsules to correspond to cognition ingredients that could be re-adapted to new tasks or contexts. Thus, we verify whether a CapsNet is more versatile than a conventional deep neural network with similar capacity in terms of neuron numbers in the layers by testing how a trained CapsNet can be adapted to perform different tasks. In our comparative study, CapsNet generated mid-level data representations more adaptable to new tasks than conventional neural networks did, which supported the above claim.

It is worth noting that this work is to discuss the characteristics of CapsNet in its learning process and those of the data representation resulted by such nets, while this work is NOT about

- establishing the supremacy of one network structure over the others in certain tasks, or
- rigorous investigation into the optimization problem of CapsNet (however, our findings may indicate interesting direction to explore for the learning of CapsNet), or
- extensive evaluation of different CapsNet architectures for certain tasks.

The contributions of this work are the practical investigation of the training procedure of Capsule networks and the effect on the learned models. More specifically, our empirical assessment of Capsule networks makes contributions by

- showing both the effectiveness of CapsNet for image representation and potential pitfalls in implementation and application of the architecture;
- revealing several interesting characteristics of the model that worth further research, specifically, these can give accessible inspirations to domains needing certain characteristics of CapsNet, such as tasks which need an effective semantic understanding of the object, or those requiring smoother and highly structured image representation.

In the following paragraphs we discuss our investigation and findings in more detail.

The first part of our investigation reveals that routing in CapsNet should be performed strategically: It is beneficial to relax the association between capsules in successive layers until the mapping between layers is sufficiently well learned. Our investigation results show that in the early stage of model training, when the map from value space of low-level capsules to the predictions space has not been well-determined, a level of uncertainty can be beneficial in the posterior inference (routing), which produces the values of high-level capsules with the mapped predictions.

A potential reason for this is as follows: Besides producing values for high-level capsules, the inference process also associates capsules between successive layers and therefore affects the following learning steps of the map from low-level capsules to predictions. Uncertainty in the inference in early training helps avoid capsules in low layer being bound to capsules in high layer prematurely. It has been shown in [6] that, since the gradient direction is not quite the right direction of descent, larger steps of updating can explore more and faster thus result in a more effective stochastic gradient descent. In this CapsNet case, what can be an inference of the statement in [6] is that low-level associations between capsules can benefit the early training, because the weights of associations which are with small gaps ensure an easier exploration in parameter space. To preserve uncertainty in early routing, one can limit the number of EM iterations or employ soft update rules (see Section III-A, Equation 1).

Our second finding is that CapsNet can produce more semantically meaningful intermediate representations in discriminative tasks compared to the standard convolutional neural networks (CNN). It is not uncommon that some natural factors, Z, other than the target Y contribute to the generative process of the image data X. For example, if we allow a hand-written digit shift within the box, the translation of the digit can be such a factor underlying the resulting set of images. Those Z-factors generally remain latent during learning and do not contribute to the discriminative goal Y. However, as the Z-factors represent meaningful physical or

geometric processes in data generation, it is desirable to distinguish the variance in Z from irrelevant noises in the intermediate data representations, i.e., to preserve the former and discard the latter.

Such factor variables are exactly the distributed representations [7] of a model, elements of which are not mutually exclusive and their many configurations correspond to the variations seen in the observed data. We examined the learned intermediate data representation in CapsNet and in a standard deep CNN architecture. CapsNet was proposed to discover hierarchies of features at different interpretation levels given image data [5], more specifically, capsule vectors of real-valued features can generalize across semantically related components of data [8]. The ability to discover entities makes it easier to reveal the Z-factors, which are meaningful in data interpretation; however, they are not directly linked to the discriminative goal. We visually examined the expectation (by using controlled Z-factors of 2D variances, such as translation on an image plane, and placing the intermediate representations on 2D using dimension reduction), and found that CapsNet functioned as expected.

The third finding follows the second finding: Recent studies [9], [10] have proved that deep neural networks are good at separating domain-related factors from variations of data samples, and then grouping features hierarchically according to their relatedness to invariant factors, therefore representations of network are robust to noise. As the data representation is meaningful and capable of catching the semantics that is unbounded to a particular task, it is natural to ask whether the representations are useful for tasks beyond those in which the model is trained, i.e., how transferable the learned representations are.

We assess the transferability by fixing the learned parameters in lower layer capsules and re-adapt the CapsNet model by re-tuning the higher part [9]. Note that this assessment scheme allows both same-task transfer and cross-task transfer. In the same-task setting, after training the model for a discriminative task, the higher part of the model is re-trained from random start for the same task. Transfer is necessary for this setting because the learned intermediate representations need to work with a re-trained higher part of the model. In both same- and cross-task testing, CapsNet demonstrated more transferability than the standard CNN did.

The remaining parts of this paper are as follows. In Section II we review the necessary background. Section III presents the main findings of the research in three aspects. Section IV concludes the paper.

II. BACKGROUND REVIEW

Research in deep neural networks (DNN) has experienced rapid growth in recent years [8]. DNN-powered learning models have become state-of-the-art in a wide range of applications, including image recognition [2], [11]–[13] and voice analysis [14], [15], detection [16], [17], and natural language processing [18], [19]. However, fundamental challenges remain in artificial neural network-based vision systems. The training process is complex and expensive in terms of both computation and training data [8], [20]. The learning is task-oriented and end-to-end, the understanding of intermediate data representation is incomplete, and decision making is obscure [4]. The insufficiently understood model may be prone to peculiar failures or attacks [21], [22]. Generalization and reliability of an existing DNN beyond the training domain can also be problematic [9].

CapsNet [5] represents an alternative visual information processing mechanism that addresses some of the above-mentioned issues. The neurons are divided into small groups in each network layer, known as capsules. The capsules correspond to concepts in different levels of abstraction during the process of parsing visual information. The cross-layer association and the activation status of the capsules represent a semantic analysis of the image data. These techniques have been used in [23], then [5] introduced the concept of dynamic routing in capsules. Recently, CapsNet has undergone some structural developments such as matrix capsules [24] and dense capsule networks [25]. It has also been employed in new application domains such as text classification [26], adversarial images detection [27], and action detection [28]. We list previous works about CapsNet in the following for intuitive summary:

- early use of capsule technique: Hinton *et al.* (2011) [23];
- dynamic routing between capsule: Sabour *et al.* (2017) [5];
- matrix capsules: Hinton *et al.* (2018) [24];
- variants of CapsNet: Phaye *et al.* (2018) [25] and Jaiswal *et al.* (2018) [29] and Zhang *et al.* (2018) [30];
- various tasks: Yang *et al.* (2018) [26] and Nguyen *et al.* (2018) [31] and Frosst *et al.* (2018) [27] and Duarte *et al.* (2018) [28] and Tang *et al.* (2019) [32];

Technical details of CapsNet are described in sub-section CAPSNET MODEL below.

The investigation into CapsNet in this work mostly relates to three topics of research in DNN and broadly in machine learning: model training, data representation, and knowledge transfer. Arguably, an efficient training methodology functions as a midwife for the real-world success of DNN [33]. Rich techniques have been proposed to address different challenges in various DNN structures, including randomly disturbing the cross-layer connections [34], introducing special gate units to retain long-term memory [35], [36] and exploiting computationally affordable structures in the gradients during optimization [37].

Existing studies on network training and optimization are mostly concerned with the standard scheme of computing the gradients using forward-backward propagation. However, the forward pass in CapsNet differs from the standard procedure and involves an iterative routing procedure. Based on this fact, an optimization view [38] was provided to formulate the routing strategy as an optimization problem, which minimizes a combination of clustering-like loss and KL regularization term. Reference [39] reveal the loss of equivariances within the relatively deep CapsNet, and apply sparse

unsupervised learning to restore this quality. Reference [40] makes the coupling coefficients trainable so as to overcome the difficulty of manually setting the routing iteration number. Further study in [30] shows that the orthogonal projection onto capsule subspaces is crucial to yield competitive performance. Nevertheless, the manner in which the routing operation affects the model training still deserves a more extensive investigation.

Data representation is one of the key elements of a successful analysis [41]. It is well known that the learned convolutional neurons in the lower layers of deep networks resemble the primary biological vision processing for discovering low-level features in images [42]. Nevertheless, the roles of intermediate or high-layer neurons in DNNs are not well understood [4], [43], [44]. Karpathy *et al.* [43] firstly explored the predictions of long short-term memory networks (LSTMs) and their learned representations on real-world data. Koh and Liang [4] tried to trance a black-box model's prediction by using a classic technique from robust statistics. To address this issue, CapsNet was proposed and has shown the promise of unveiling meaningful data structures in image populations. In this work, we perform a comprehensive study on CapsNet and propose a systematic, quantitative evaluation protocol.

One advantage of general AI is its supreme adaptability. Every day a two-year-old child learns new skills with a glance, while a state-of-the-art algorithm requires millions of training examples to train. Tremendous research focus has been placed on transfer learning [45]. In particular, a pioneering investigation has revealed characteristics of DNN layers under transfer tasks [9]. Various techniques have been proposed to improve the adaptation of learned data representations under transfer settings such as object recognition, image classification, and human action recognition, as reviewed in [46]. The study of meta-learning has achieved an outstanding generalization performance and it solves transfer tasks using only a small number of samples [47]. Reference [48] proposed a domain adaptation approach and adopted the adversarial network to learn transferable representation. The above-mentioned meaningful capsule data representation indicates that capsules are conducive to knowledge transfer, which is supported by the experiments described in Section III.

A. CAPSNET MODEL

We provide a brief review of CapsNet, with a special focus on the routing operations. Readers can refer to [5] for additional details. In CapsNet, raw images usually pass a preprocessing convolution layer that produces a number of channels, which are then grouped as capsules. For example, if one employs a standard convolutional layer of 256 output channels, the output of this layer can be re-organized as 64 capsules, with each capsule being a 4-D vector. It was argued that the new structure of the channels provides an advantage to image analytics [5]: A vectorial unit, $\mathbf{v}$, can represent both the characterization of an entity (parameterized by the direction, $\frac{\mathbf{v}}{\|\mathbf{v}\|}$) and the likelihood of the presence of the entity (the magnitude, $\|\mathbf{v}\|$).

Algorithm 1 Computation in CapsNet.

input : transform matrices $\mathbf{W}_{ij}$
input : capsule vector $\mathbf{v}_i$, $i \in \{1, \ldots, n\}$
output: capsule vector $\mathbf{v}_j$, $j \in \{1, \ldots, m\}$
begin
  $\hat{\mathbf{u}}_{j|i} \leftarrow \mathbf{W}_{ij}\mathbf{v}_i$; $b_{ij} \leftarrow 0$,
  $i \in \{1, \ldots, n\}, j \in \{1, \ldots, m\}$
  while *not RouteDone* **do**
AC $c_{ij} \leftarrow \frac{e^{b_{ij}}}{\sum_j e^{b_{ij}}}$, w.r.t. Eq. 3 in [5], $i \in \{1, \ldots, n\}$, $j \in \{1, \ldots, m\}$
    for $j \leftarrow 1$ **to** m **do**
WS $\mathbf{s}_j \leftarrow \sum_i c_{ij}\hat{\mathbf{u}}_{j|i}$, w.r.t. Eq. 2 in [5]
SQ $\mathbf{v}_j \leftarrow \frac{\|\mathbf{s}_j\|^2}{1+\|\mathbf{s}_j\|^2}\frac{\mathbf{s}_j}{\|\mathbf{s}_j\|}$, w.r.t. Eq. 1 in [5]
      $b_{ij} \leftarrow b_{ij} + \langle \mathbf{v}_j, \hat{\mathbf{u}}_{j|i} \rangle$, $i \in \{1, \ldots, n\}$
    end
SC RouteDone $\leftarrow$ IsMet(StopConditions)
  end
end

Algorithm 1 have listed the major steps of the forward-pass computation in a CapsNet. The computation of a high-layer capsule $\mathbf{v}_j$ is iterative and the "StopConditions" in Step **SC** corresponds to the predefined number of routing iterations. In one step, a low-layer (input) capsule i, $\mathbf{v}_i$, contributes to a high-layer capsule j via a transformed $\hat{\mathbf{u}}_{j|i}$ using weight matrix W_{ij}. The agreement between i and j is determined by the inner product of $\langle \hat{\mathbf{u}}_{j|i}, \mathbf{v}_j \rangle$ and accumulated to b_{ij}. Then a set of association coefficients c_{ij} are updated according to b_{ij} (as in Step **AC**). The value of $\mathbf{v}_j$ is then updated by non-linear squashing of the weighted sum of the transformations (as in Steps **WS** and **SQ**). Fig. 1 illustrates an example of the computation of three high-layer capsule vectors, $\{\mathbf{v}_j, \mathbf{v}_{j'}, \mathbf{v}_{j''}\}$, from one low-layer capsule vector, $\mathbf{v}_i$.

III. EMPIRICAL STUDY ON CAPSNET LEARNING

For empirical study, we design several experiments to test the three aspects of CapsNet: model fitting, representation learning, and representation generalization. Each relatively independent aspect will be detailed mentioned in the corresponding sub-section, which contains technological details, results, and discussions.

A. MODEL FITTING

This section presents the results of our experiment and the analysis of structural and operational factors that affect the model training process of CapsNet. Multi-layer neural networks are powerful generic function approximation models, while the new family of CapsNet [5] introduces additional versatility via routing in data representation, which is particularly effective in extracting the semantic hierarchy embedded in sensory data. Nevertheless, a data model can only

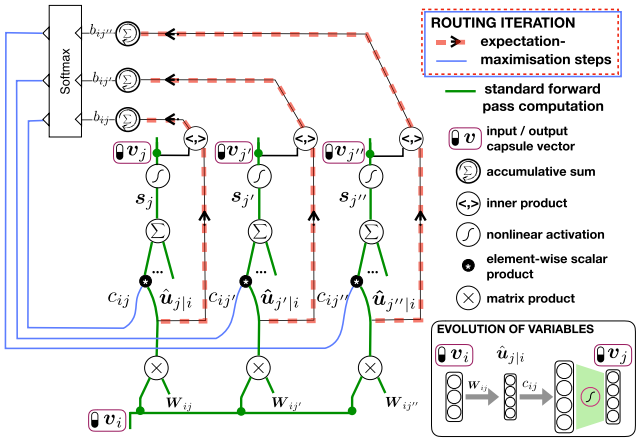

FIGURE 1. Routing iteration in the forward pass computation between consecutive layers in a CapsNet. This figure shows computing high-layer capsules j, j' and j'' using a low-layer capsule i in the routing operation of CapsNet. Line colors roughly indicate operands in different steps in routing. Bold-green denotes the conventional neural network forward operations: Transforming and summation of inputs and outputting to the next layer. Broken-red is for determining the agreement between low- and high-layer capsules. Blue is for updating the association coefficients.

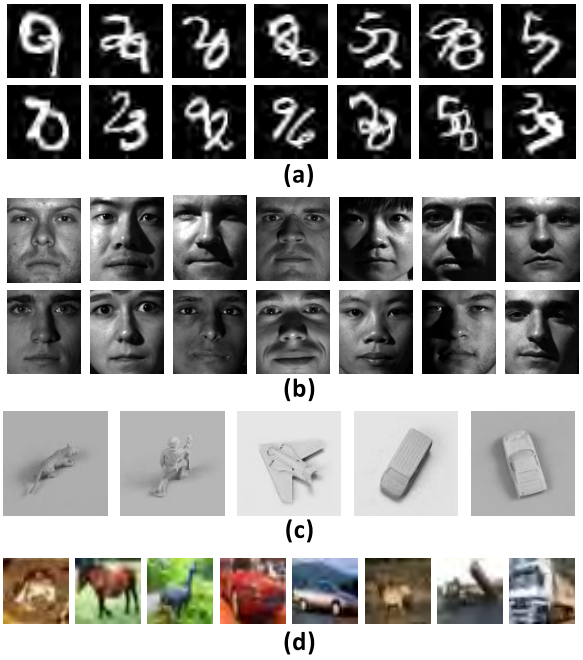

FIGURE 2. Samples images from the multi-MNIST, Extended Yale B, smallNORB and CIFAR-10 datasets.

realize its potential if there is an effective way of fitting the model to data. Multi-layer neural networks have waited decades before undergoing explosive growth thanks to effective training techniques and computational resources. Thus, it is natural to ask what CapsNet has provided considering the trade-off between model capability and training complexity. In particular, data representation routing is realized as EM inference on the association between two layers of capsule units. We test and analyze how the EM operations affect the model training and performance. In the following part we first introduce two types of used data (including four datasets) in subsection **Data** as well as the networks in **Models**. We then indicate the two main control variables used to investigate the effect of routing in **Experiment**. Finally the evaluation result would be showed and corresponding discussions would be given in **Evaluation and discussion**.

1) DATA

We perform this test on four image datasets for classification, where two relatively small datasets of hand-written digits and faces, the other two datasets are more complex and of real objects. For the first dataset, as used in [5], the task is to recognize two handwritten digits in one image. Each sample of data is a 36×36 grey-scale image by superposition of two hand-written digit images from the MNIST dataset [49], with a duplex label of the two digits. We call this dataset multi-MNIST. The dataset contains $30,889$ training samples and $4,738$ test samples. The second dataset is the Extended Yale Face Database B (YaleB) [50], which contains $2,432$ facial images ($2,166$ for training, the rest for testing) of 38 subjects taken under 64 different illumination conditions. In this experiment, we cropped and re-sized these raw images to 168×192. The third dataset is smallNORB [51], which consists of 96×96 stereo grey-scale images belonging to 5 generic categories. There are both $48,600$ images in training set and test set. We resized the images to 48×48 and during training processed random 36×36 crops of them. We passed the central 36×36 patch during test. As for the last dataset, CIFAR-10 [52] has $60,000$ 32×32 color images in 10 classes, with $6,000$ images per class. There are $50,000$ training images and $10,000$ test images. Fig. 2 shows a few examples of images of the four datasets in the (a), (b), (c) and (d) subplots, respectively. It is worth noting that our aim in this test is to investigate the effect of the routing operation on model training, rather than to advance the state-of-the-art for those classification tasks. The classification performance is not the ultimate goal (comparative performance evaluation has been done in [53]), it is used as a gauge of how well the model has been fitted to the data.

2) MODELS

Fig. 3 illustrates the structure of the CapsNet and standard CNN designed for multi-MNIST in the experiment. A convolutional layer receives the input and is followed by three capsule layers. The last capsule layer represents the prediction of classes (specifically, we assumed the indexes of the most active two capsules as predicted digits). As a baseline to assess the training, we have also constructed a conventional neural network with standard convolutional and linear layers (i.e., a standard CNN, as shown in Fig. 3). The standard CNN has the same structure as the CapsNet. In each intermediate layer, we keep the number of total neurons in the standard CNN and CapsNet the same. For example, in Fig. 3, layer Cap1 in CapsNet has 128 capsules that contain

TABLE 1. Accuracies under different n^{EM} and η settings on four datasets.

Datasets	$n^{EM}=1$ $\eta=1.0$	$n^{EM}=3$ $\eta=1.0$	$n^{EM}=6$ $\eta=0.01$	$n^{EM}=6$ $\eta=0.2$	$n^{EM}=6$ $\eta=0.8$	$n^{EM}=6$ $\eta=1.0$	$n^{EM}=7$ $\eta=1.0$	$n^{EM}=15$ $\eta=1.0$	CNN
multi-MNIST	97.0	96.8	96.3	96.8	95.3	94.3	87.7	87.0	92.0
Yale B	99.6	96.7	97.9	95.8	94.2	95.4	93.8	95.0	96.7
smallNORB	88.6	89.6	89.0	87.7	84.4	84.0	83.0	77.9	86.9
CIFAR-10	90.2	88.2	90.1	90.4	87.6	87.9	87.8	88.0	90.3

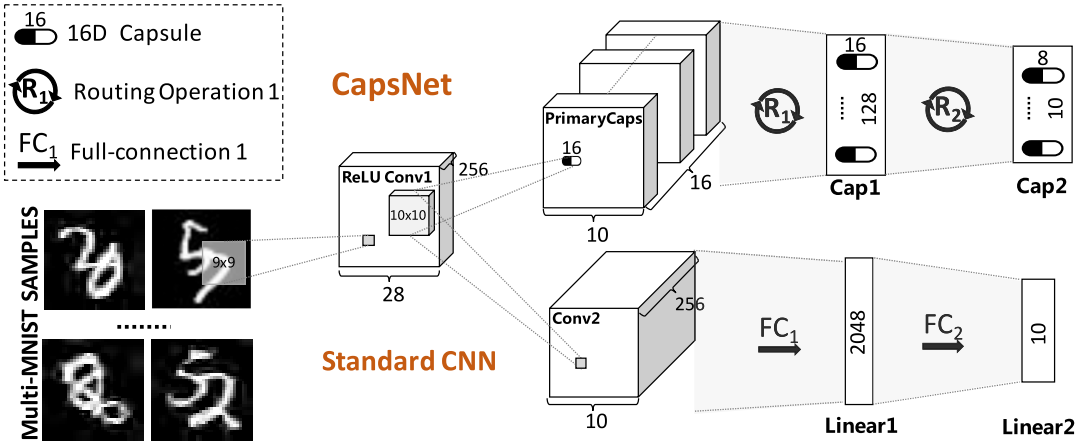

FIGURE 3. CapsNet and standard CNN models. The upper and lower diagrams show a 4-layer CapsNet as in [5] and a similarly structured standard convolutional neural network, respectively. Legends in the figure indicate capsule units of certain dimensions, the routing operation, or the fully connection between layers.

16 dimensions, so the corresponding layer Linear1 in the standard CNN was constructed with 2048 neurons. Note that networks trained on YaleB and smallNORB datasets have almost the same structures. The difference of the network on CIFAR-10 dataset is, we adopt a much deeper model which is constructed by simply replacing $9-10^{th}$ layers of VGG19 [12] (one of the state-of-art models) with capsule layers. Correspondingly, VGG19 is regarded as the standard CNN used on this dataset. And the principle is also to keep the number of parameters in CapsNet the same with VGG19.

3) EXPERIMENT

We want to investigate the effect of the EM operation in forward-pass of the CapsNet during the training process. Naturally, we adopt two schemes to control the effect of the EM operation imposed on the forward pass computation: i) to limit the number of EM iteration and ii) to test different EM update rates in the "soft" update iterations. Both settings refer to the implementation in Algorithm 1. Iteration number n^{EM} indicates the EM loops that compute the coefficients affecting Step **SC** in the algorithm. The update rate is a parameter that we introduce to stabilize the training - we replaced Step **AC** in the algorithm with a soft update rule

$$c_{ij}^{New} \leftarrow \eta \hat{c}_{ij} + (1-\eta) c_{ij}^{Old} \tag{1}$$

where $\hat{c}_{ij}$ is the coefficient computed using the original EM update rule in Algorithm 1, and η is the EM update rate. Note that we would not care much about the extra computational complexity in this investigation experiment for routing operation. By applying various n^{EM} and η, which are in gradually incremental ranges (1, 3, 6, 7, 15) and (0.01, 0.2, 0.8, 1.0) respectively, we evaluate the models being trained periodically to gauge the progress and the quality of training. The evaluation protocol on multi-MNIST follows that in [5], i.e., a correct prediction requires the model to output the identities of both digits correctly. All the training processes share the same optimization settings of Adam [37] and adopt Margin loss defined in [5]. The batch size at each training step is 8. We fine tune the learning rates of models carefully to reach high accuracy rate. Finally, we fix the learning rates of 1e-2 and 2e-5 for CapsNet and standard CNN in our experiments. See the results below for further discussion on the effects of training.

4) EVALUATION AND DISCUSSION

We verify the training progress by plotting the classification performance of the model during training on the four datasets in Fig. 4,5,6 and 7, respectively. Table. 1 shows specific classification accuracies under different settings on four datasets. Fig. 4,5,6,7(a) show the model performance during training under different n^{EM} settings. Fig. 4,5,6,7(b) show the process under different η settings, which we will discuss in the next paragraph. We tested each setting of EM

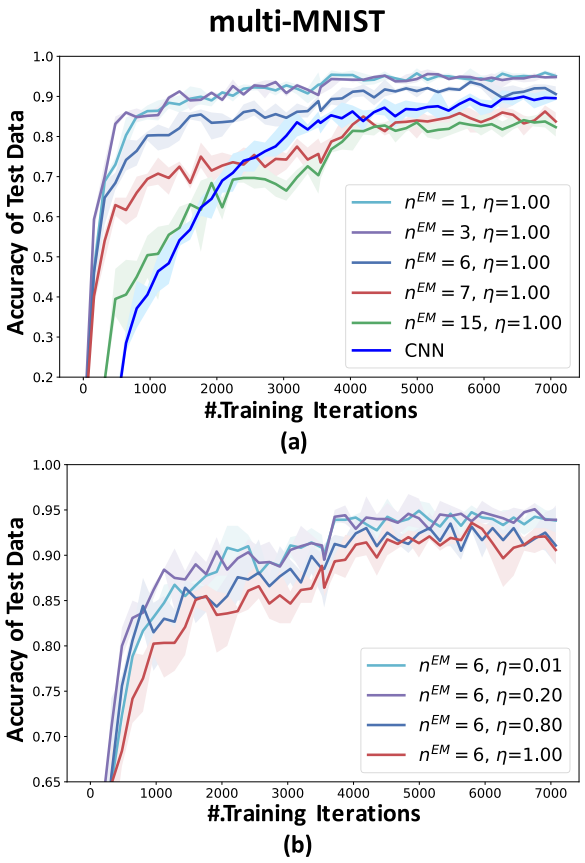

FIGURE 4. Effects of EM iteration number n^{EM} and Update Rate η on CapsNet training on the multi-MNIST dataset. This figure shows the training progress in terms of prediction accuracy on the test dataset against the training steps in mini-batches. The solid curve represents the mean accuracy in five trails using the same n^{EM} and η. Shaded areas represent one standard variance of the model performance in the trails. (The figures in this paper are best viewed in colors.).

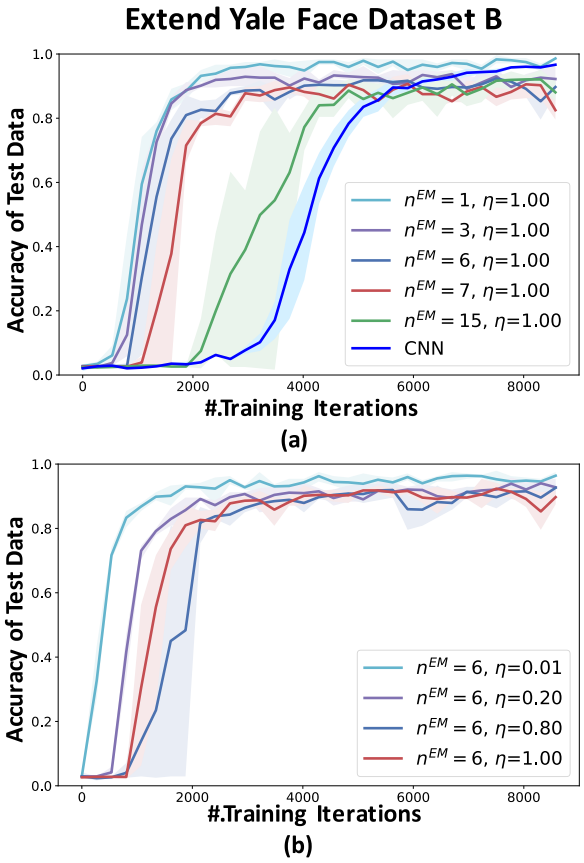

FIGURE 5. Effects of EM iteration number n^{EM} and update rate η on CapsNet training on the Extend Yale Face Database B. See Fig. 4 for additional details.

in five trails and reported the mean and variance of the model performance during the training processes. An unexpected finding was that CapsNet needs fewer training steps than the standard CNN and can achieve superior classification accuracy. However, CapsNet actually yield more time cost because of additional computational complexity in routing procedure, this phenomenon has been shown clearly in [53] and we will not discuss much of it here. It is noteworthy that the training of CapsNet saturates with increasing EM iterations after n^{EM} reaches a small number. In fact, excessive EM iterations (e.g., $n^{EM} \geq 7$ in Fig. 4,5,6,7(a)) impact the effectiveness of the training and deteriorate the performance. A possible explanation is as follows. At the beginning of the training stage, the network weights have not been conditioned to represent meaningful image elements. The routing produced by EM is mostly random. We can eliminate the chance of high-layer capsules being exposed to all the data samples by forcing the high-layer capsules to focus only on a small subset of low-layer inputs. In an early stage, those subsets tend to be randomly assorted without any semantic significance, and the selective representation scheme (low-high layer association with a sparse matrix $\mathbf{C} : \{c_{ij}\}$) is more likely to impair rather than to improve the capsules' ability to discover meaningful attributes of the entities. Moreover, on more complex datasets, such as YaleB, the training process seems to contain a large variance when a strong routing configuration is used, such as the EM iteration $n^{EM} = 15$ in Fig. 5(a). The phenomenon is potentially due to the stochastic sampling process in the complex dataset.

In fact, the observation of the "early over-routing" phenomenon in Fig. 4,5,6,7(a) has motivated the introduction of the soft EM update scheme as in (1). Fig. 4,5,6,7(b) show the model training of the median $n^{EM} = 6$ under different η-values. The results indicate the advantage of soft EM updates, e.g., by comparing the curve for $\eta = 0.01$ and that for $\eta = 1.0$. While routing starts with a slight bearing towards some high-layer capsules, instead of slumping to a random few in the high layer, the network weights are able to represent effectively the semantics in the image. In the later training stage, the routing process helps to divide neuron information into concept entities that can improve the network's performance.

From the viewpoint of a low-layer capsule i, the corresponding association coefficients $\{c_{i1}, c_{i2}, \dots\}$ of the capsules in the layer above can be considered as a probability

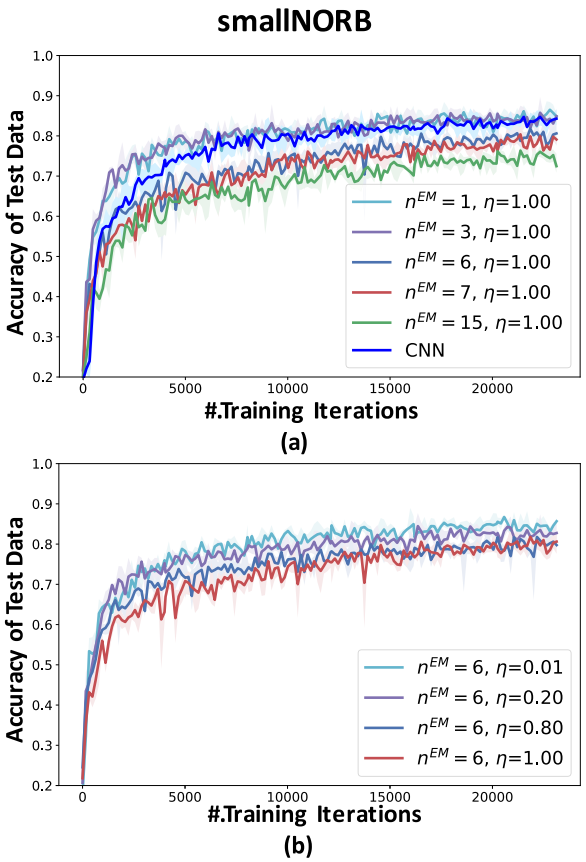

FIGURE 6. Effects of EM iteration number n^{EM} and Update Rate η on CapsNet training on the smallNORB dataset. See Fig. 4 for additional details.

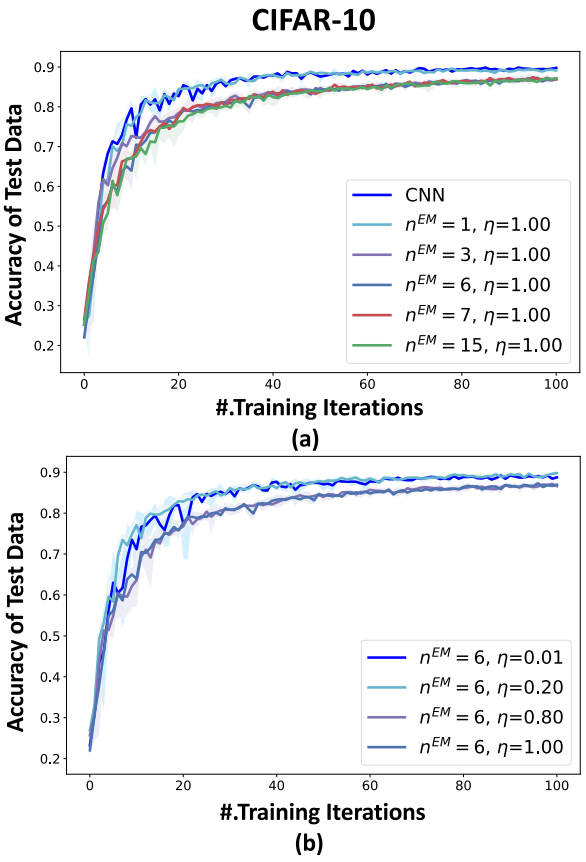

FIGURE 7. Effects of EM iteration number n^{EM} and Update Rate η on CapsNet training on the CIFAR-10 dataset. See Fig. 4 for additional details.

distribution over the high-layer capsules. Let $P_i^c(j) = c_{ij}$ represent the event that "entity i's presence is interpreted by the presence of high-level entity j". In a sense, the entropy of the distribution, $H[P_i^c]$, measures the uncertainty at this step in the simulated cognition process, while forming high-level concepts using low-level information.

Fig. 8,9,10 and 11 show the trends of the average association entropy [54] over the training process. In particular, the association coefficients are from the first EM routing operator $\mathcal{R}_1$, as shown in Fig. 3. Between the two layers, $\mathcal{R}_1$ produces a coefficient matrix $\mathbf{C}^{(n)}$ for each data sample n, and the i-th row of $\mathbf{C}^{(n)}$ realizes the abovementioned distribution, from which we can compute an entropy value $H_i^{(n)}$. We take the average for the entropy over every sample n and every low-layer capsule i,

$$\begin{aligned}\bar{H} &= \frac{1}{N \cdot I} \sum_{n=1}^{N} \sum_{i=1}^{I} H[P_i^{c,n}] \\ &= -\frac{1}{N \cdot I \cdot J} \sum_{n=1}^{N} \sum_{i=1}^{I} \sum_{j=1}^{J} c_{ij}^{(n)} \log c_{ij}^{(n)} \end{aligned} \tag{2}$$

where the superscript $^{(n)}$ indicates the data sample. The trends of the average entropy shown in Fig. 8(a) and Fig. 9(a) on the multi-MNIST and YaleB data reveal the mechanism of routing in two aspects:

1) Reading the plots Fig. 8(a) and Fig. 9(a) vertically: at a certain training iteration (x-axis in the figures), we first compare the average entropy (y-axis in the figures) for different EM settings. As expected, if EM is performed with more stringent updating rules (i.e., high updating rate), the less uncertainty remains in the resultant association distribution and the entropy reduces.
2) More interestingly, we can also read the plots Fig. 8(a) and Fig. 9(a) horizontally: we verify how the entropy reached using a certain EM setting varies along the model training. The plot shows that the entropy reduces when the model becomes more completely trained.

A possible interpretation of observation 2 is as follows: When the model fits well to the images, the parts to which the response of individual capsule units become clearer and the connections between layers (coefficients W, not to be confused with association $\mathbf{C}$) become more relevant. In general, we can be more certain about whether a low-layer capsule should contribute to the activation of a high-layer capsule. As an intuitive example, one can usually determine with more confidence whether a body part belongs to some creature than whether an amorphous blob of pixels belongs to some blurry assortment.

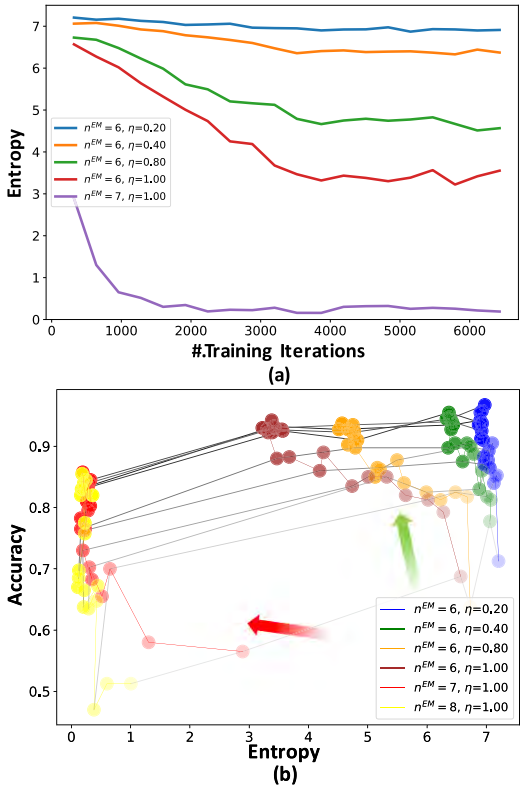

FIGURE 8. Trends of entropy of the stochastic association between successive capsule layers during training on multi-MNIST. (a) shows the change in entropy with training iterations; the top four lines are under different η-values of the fixed EM-iteration 6. The last line shows one η-value setting of the fixed EM-iteration 7. (b) illustrates the trend in the change of entropy and classification accuracy. The curve marker colors represent different EM-settings. Transparency indicates training progress. Markers in one gray line represent models that have undergone the same number of training iterations. The red and green arrows show appropriate and excessive reductions of the associated entropy, respectively. See the text for further details.

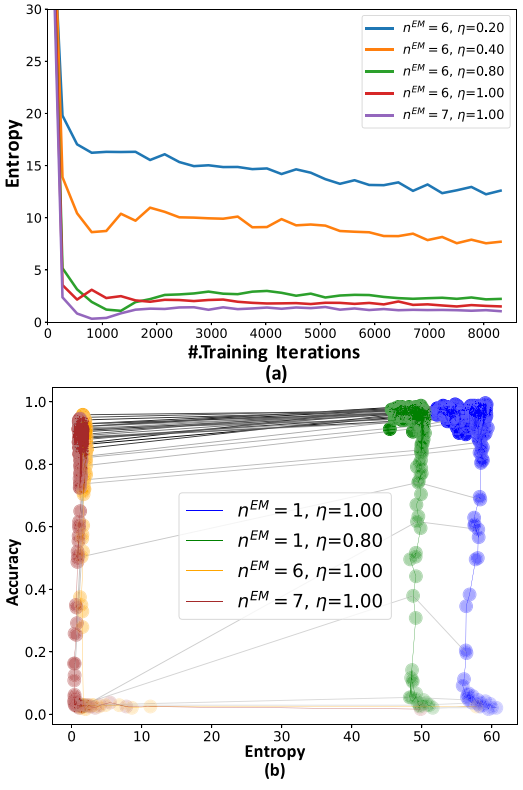

FIGURE 9. Trends of entropy during the training on the Extend Yale Face Database B. The interpretation of this figure is similar to that of Fig. 8.

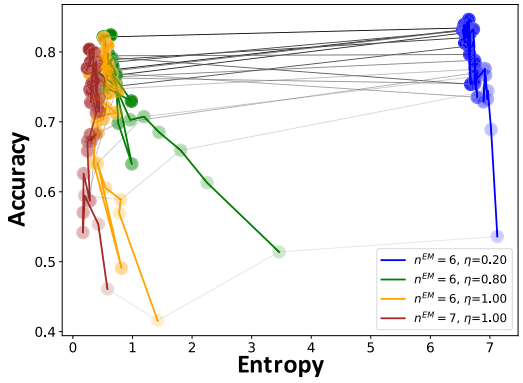

FIGURE 10. Trends of entropy during the training on smallNORB dataset. See Fig. 8 for additional details.

The above understanding leads to a heuristic training strategy: The EM operation should be modulated so that the certainty of the resultant association matches the model fitness to the data (reflected by accuracy). In Fig. 8(b),9(b),10,11, we plot the average association entropy against the model performance for models at different training stages and EM settings on the four datasets. Each colored curve represents one EM operation setting. Each grey line links models tested after the same number of training steps. The general tendency in the plot is consistent with our observation in both Fig. 8(a) and Fig. 4: during training, the model accuracy increases with the association certainty. The green arrows in Fig. 8(b) intuitively illustrate the phenomenon. Curves that correspond to this phenomenon in Fig. 9(b),10,11, can also reach better final accuracy than the others. The red arrows, corresponding to deeply reduced entropy in the early training stage indicate negative impacts on training. It indicates that CapsNet overestimates the confidence without an appropriate fitting, and thereby it may encounter the "early over routing" issue as discussed above.

A possible interpretation of the relationship between the entropy in the association distribution and the training progress is as follows. A key of CapsNet is that the higher-layer capsules summarize the statistics of the lower-layer ones, and routing calculates the low-to-high association. If the view holds, we argue that *Using the same inference operation, routing in the networks that are better fitted to represent the data would produce a more certain low-to-high association than routing in a less-fitted network.*

We further elaborate on this statement as follows: As an example, a fixed routing procedure produces two association relationships $\mathbf{C_1}$ and $\mathbf{C_2}$ between the corresponding capsule layers in two networks of the same structure but different parameters, Net[θ_1] and Net[θ_2] (see Fig. 12 for illustration).

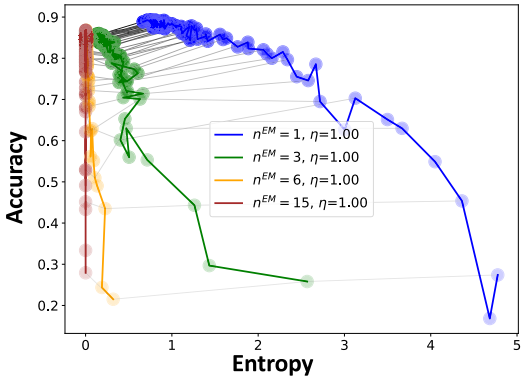

FIGURE 11. Trends of entropy during the training on CIFAR-10 dataset. See Fig. 8 for additional details.

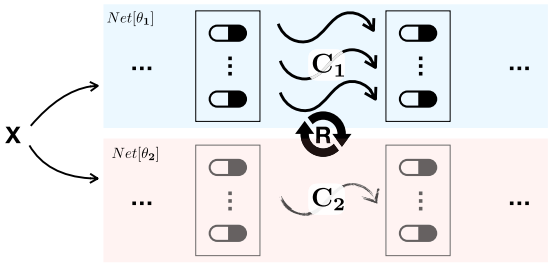

FIGURE 12. Association relationships between two layers using the same routing procedure in two nets.

Without losing generality, let Net[θ_1] be well fitted to the data and Net[θ_2] be less well fitted. Then capsules in Net[θ_1] tend to correspond to entities with stronger expression in the data than those of Net[θ_2] do. In such a context, if the routing operation ("R" in Fig. 12) successfully discovers the association from the low-level entities to the high-level entities as proposed in [5], the association $\mathbf{C_1}$ should be more certain than $\mathbf{C_2}$. This is what we have observed in the previous experiment.

We have got a consistent conclusion from experiments on both simple and complex data. However, comparing to the significant advantage of CapsNet on multi-MNIST and YaleB datasets, there are only a small gap between Standard CNN and the CapsNet with the best routing setting. It reveals that CapsNet needs to be improved. The following sections will show the significant potential for CapsNet to reach better performance.

B. REPRESENTATION LEARNING

In practical data analytics, the data population often has intrinsic structures that are different from the main target but are semantically meaningful. The two previously used datasets are taken as an example. Each image in multi-MNIST dataset contains two handwritten digits as described in III-A, the relative location of the two digits is a geometric attribute of the image content. In the YaleB dataset, the illumination condition represents an important factor underlying the appearance of the faces. This section presents

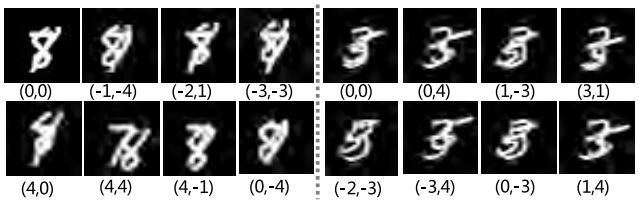

FIGURE 13. Example images of two shifted and overlapping hand-written digit images. The left panels show images of fixed "7" and moving "8"; the right panels are of fixed "3" and moving "5". The label below each image indicates the offset (±rows, ±cols) of the moving part with respect to the fixed part. Taking the (7,8) images as example, (0,0) means that the image "8" is located with the center overlapping that of "7" and (−1,−4) means that the "8" has been shifted 1 unit up and 4 units to the left.

the experimental results on how the network models' learned data representation corresponds to the intrinsic structures in the population distribution of the data while being trained for the discriminative tasks. Beyond assessing the representation by intuition and visual plausibility, we have constructed a dataset based on the multi-MNIST with internal structures induced by geometric transformations, which is introduced in the following subsection **Data**. In subsection **Experiment** we perform both quantitative and qualitative assessment of the learned data representations by CapsNet and standard CNN introduced in subsection **Models**, finally show and analyze the results in **Results and discussion**.

1) DATA

In this experiment, we use both hand-written digits and YaleB faces. For the hand-written digits, we use the multi-MNIST dataset for training, as in the previous section. However, the test is conducted on a specially constructed image datasets with known underlying 2D manifold spanned by the geometric translations. More specifically, we generated images of two digits by moving one digit while keeping the other digit fixed. The population of such a dataset is naturally distributed on a 2D manifold. Fig. 13 shows two example test datasets of digits (fixed-7, moving-8) and (fixed-3, moving-5), respectively. We shifted the moving digit horizontally and vertically by $[-4, +4]$ units, resulting in $9 \times 9 = 81$ sample images per test dataset. Note that we discussed above the test datasets. The models for hand-written digits are trained on the multi-MNIST data as in the last experiment. Notably, we evaluate the intermediate data representations in networks trained for classification (toward hand-written and YaleB face images), rather than optimizing the network deliberately for discovering the intrinsic manifold in the test data.

2) MODELS

In this experiment, we use the same CapsNet and standard CNN models as described in the last Subsection III-A. Note that the result is not to be compared with a representation learned with unsupervised settings. All the results are based on the representation generated by previously trained networks for both the moving digit data and the YaleB face data.

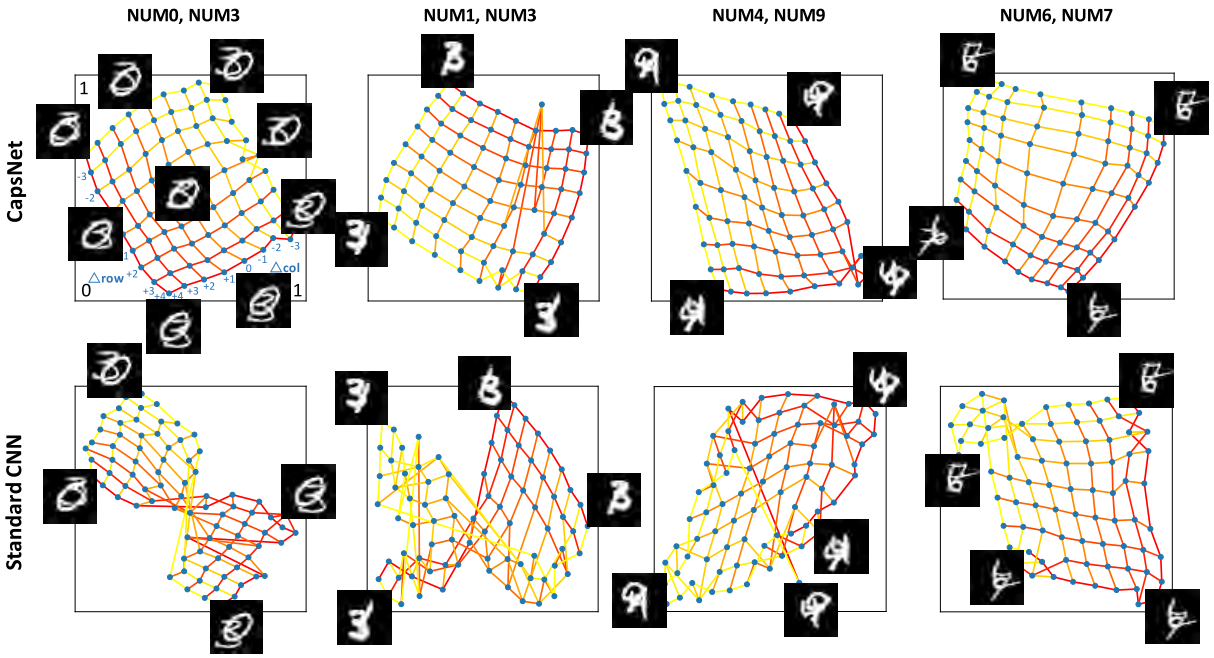

FIGURE 14. Embedding of learned representations. This figure shows the 2D embedding result of CapsNet and standard CNN on four different test datasets. The points in the subplot correspond to samples in a dataset. Digits being shifted by the same number of units and direction have been connected by a line with certain color.

3) EXPERIMENT

Using trained networks to process the test datasets (moving digit data and YaleB face data), we collect the data representations at an intermediate layer of neurons/capsules (the capsule layer after the first routing operation and the counterpart layer in the standard CNN, see Fig. 3 for "Cap1" and "Linear1", respectively). Then, we apply the manifold embedding algorithm t-SNE [55] to render the learned representations into $\mathbb{R}^2$.

4) RESULTS AND DISCUSSION

Fig. 14 and Fig. 15 illustrate the t-SNE $\mathbb{R}^2$ embedding of the samples in several test datasets of moving digit and the dataset of YaleB faces. The resultant embedding of the moving digits (shown in Fig. 14) indicates that the data representation and the corresponding embedding from CapsNet are smoother and better aligned with the internal 2D manifold than those from the standard CNN are. The intuitive results reveal that semantically related components of data can be represented by Z-factors which are not directly linked to the discriminative goal. In this case, capsules make it easier to discover such factors, result in producing semantically meaningful intermediate representations. Note that these results and arguments are concluded from empirical study, which haven't been rigorously guaranteed yet in theory. The quantitative validation is as follows. We take a regular grid on $\mathbb{R}^2$, corresponding to the movements $G := \{(-4,-4),(-3,-4),\ldots,(+4,-4),\ldots,(+4,+4)\}$. We then minimize the chamfer distance (CD) [56] by transforming the $\mathbb{R}^2$ embedding and compute the CD to the regular grid

$$d_{cd}(S_1,S_2) = \sum_{x\in S_1}\min_{y\in S_2}\|x-y\|_2^2 + \sum_{y\in S_2}\min_{x\in S_1}\|x-y\|_2^2 \quad (3)$$

$$\text{minimize}_T\, d_{cd}(T(Z),G) \quad (4)$$

where S_1 and S_2 are two sets, T represents isotropic transformations on $\mathbb{R}^2$, and Z represents the embedded data in $\mathbb{R}^2$. The minimization in (4) is over all possible transformations and can be understood as the best attempt to align the embedded samples and the regular grid G. Table 2 shows the mean and variance of the chamfer distances resulted from CapsNet and CNN embedding of the moving digit dataset. The mean value of CapsNet is over 1.25 times bigger than that of CNN, and, the variance value is 2.5 times bigger. The difference of values in Table. 2 is significant since they are computed under a normalized point space (as shown in Fig. 14).

TABLE 2. Chamfer distance between regular grid of movements and $\mathbb{R}^2$ embedding.

	CapsNet	Standard CNN
mean	0.313	0.394
variance	0.010	0.025

We visually inspect the representation embedding of the YaleB faces, as shown in Fig. 15. The locations of the face images correspond to the embedded sample in $\mathbb{R}^2$. The embedded arrangement of the samples according to CapsNet representations are more plausible than that according to CNN representations. We can find that photographs taken

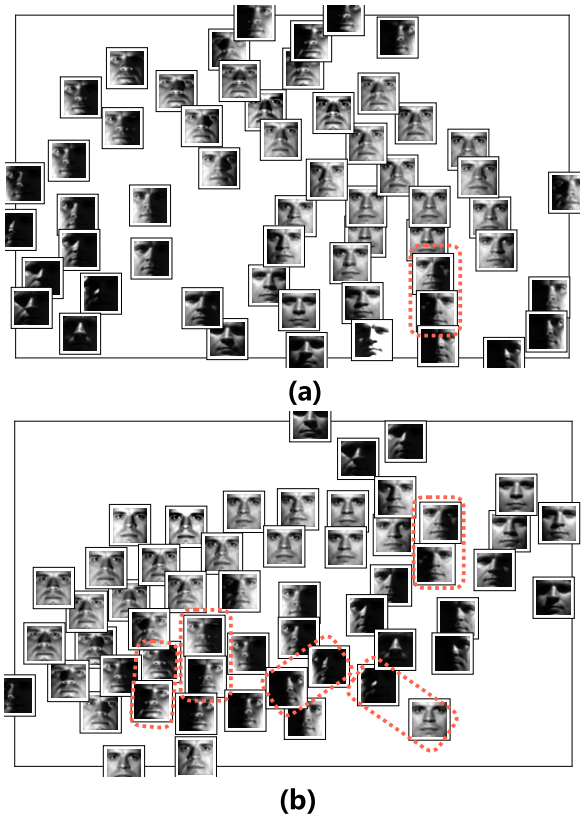

FIGURE 15. Embedding of the presentation learned in the Extended Yale Face Database B. This figure shows the 2D embedding result of CapsNet and the standard CNN on images of certain person under 64 illumination conditions. The small images on the embedding indicate the raw images of the representations there. The areas circled by red dashed denote those obviously not in line with human vision in light and shadow.

with light from the left mostly appeared in the lower left region of subplot (a), while photographs taken with light from the right appeared in the lower right region. More generally, photos of the position from-bottom-to-top correspond to the light angles changing from top to bottom. The embedding shown in subplot (b) has a less meaningful interpretation (as marked with red dashed in Fig. 15, subplot (b) has more areas unnatural in the transformation of light and shadow).

C. REPRESENTATION GENERALIZATION

Meaningful data representation can facilitate the knowledge transfer or generalization to distinctive cognition tasks. In this experiment, we further investigate how a trained CapsNet generalizes beyond the original task. This issue is also known as transfer learning. It refers to generic artificial intelligence: The model can be used or easily adapted to tasks that are different from the original task for which the model has been trained. This assumes that the new tasks share certain structures at the cognitive or knowledge level with the original one, and thus, the model can take advantage of the shared characteristics of the tasks. We have carefully designed two similar tasks and employed an intuitive scheme to evaluate the transferability of CapsNet and the standard CNN.

Similar to previous section, we describe the tasks and data in subsection **Data**, as well as the used models and experiment settings in **Models** and **Experiment**, respectively, finally we show and analyze the results in **Results and discussion**.

1) DATA

The multi-MNIST dataset is naturally suitable for this experiment. We split the set of images and targets into tasks A and B with a little technique: the image is composed by a digital pair sequentially divided; one of the digits is at the left-top and another at the right-bottom, so the number of these targets is $10 \times 9 = 90$, where 45 samples were distributed to task A and 45 to task B. The two subsets, namely S_A and S_B, are constructed so that i) each two-digit class of images are exclusively within S_A or S_B, and ii) S_A and S_B both contain a complete set of digits. In particular, we have yielded $33,497$ task A samples ($31,049$ training, $2,448$ testing samples) and $30,440$ task B samples ($28,148$ training, $2,292$ testing samples).

The motivation for using S_A and S_B is as follows. Condition i) ensures that during training, the model does not use the images of the same two-digit label on which it will be tested, and Condition ii) ensures that the model has seen all the necessary concepts, i.e., the individual digits, in order to successfully perform the new task. For example, if the model is to be tested on images containing digits $(7, 8)$, then we do not use images of $(7, 8)$ in the training stage. Instead, the training data contain the appearance of both 7 and 8 in images such as $(7, 9)$ and $(1, 8)$, as shown in Fig. 16(a).

2) MODELS

The CapsNet and standard CNN models are similar to those used in Subsection III-A with one additional capsule/CNN layer, respectively. The extra layer is for testing the transferability of the neurons at different layers.

3) EXPERIMENT

We follow a similar test protocol to that in [9]: The models are trained in domain S_A and tested in domain S_B. We re-adjust the last one or two layers using the training data of domain S_B, while retaining the remaining net parameters as trained in S_A. Such models are called A1B (1 layer fixed to S_A-training, 2 layers adjusted on S_B) or A2B (2 layers fixed to S_A-training, and 1 layer adjusted on S_B). As a control test, the experiment also includes networks prepared following the above protocol with the difference that the first training pass is on the target domain S_B. We call such control set of models as B1B (fixing 1 layer and re-adjusting 2) and B2B (fixing 2, re-adjusting 1). Fig. 16(b) illustrates the test schemes of B1B and A2B.

Note that the parameters in the last 1 or 2 layers of B1B/B2B have been randomly initialized before the re-adjustment stage, in order to distinguish data representation in different layers and test if the information in previous layers can be decoupled from the original learned model. If the data representation of a certain layer interact

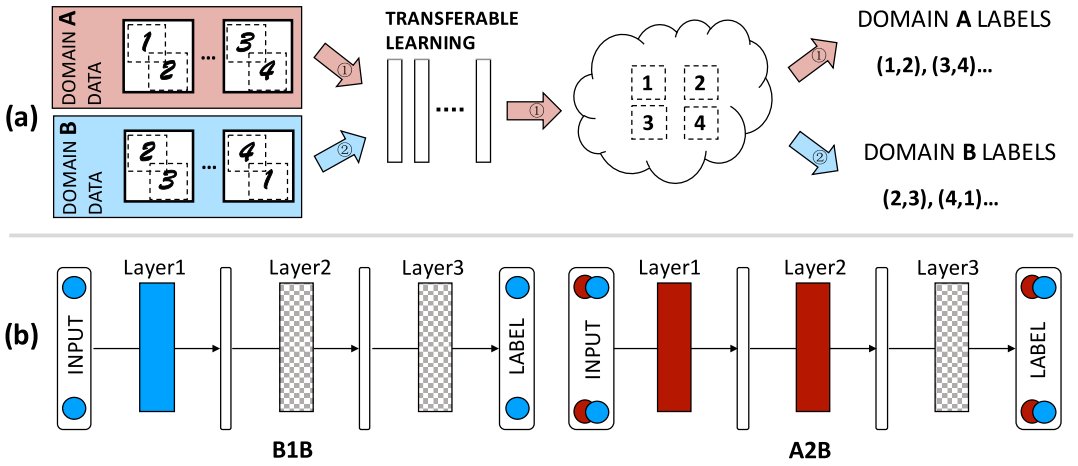

FIGURE 16. Example testing schemes of domain transfer. The colors represent domain, red for A and blue for B. (a) Domain data are the superposition of 2 hand-written digits. Domain A and B contain different combinations of digits. (b) The tests are to determine how representations produced by pre-trained layers can help new cross-/same-domain tasks. The chessboard pattern stands for layers to re-adjust. B1B represents keeping 1 pre-trained layer on domain B and re-adjusting the top 2 layers also on domain B. A2B represents keeping 2 pre-trained layers on domain A and re-adjusting the top 1 layer on domain B. See the text of this article and refer to [9] for more detailed discussions of the testing protocol.

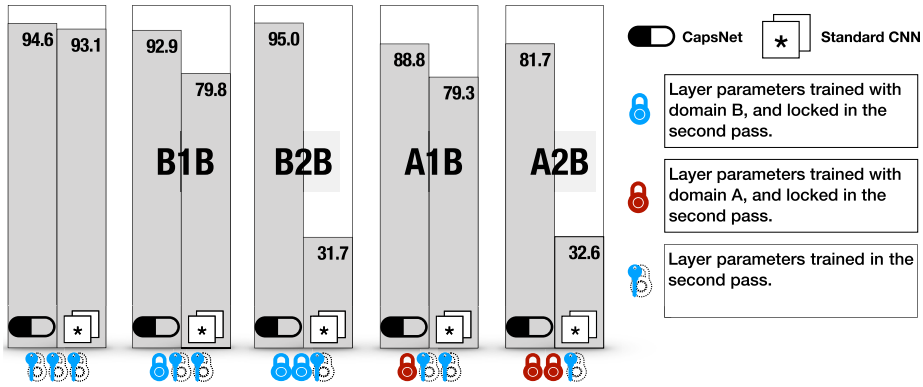

FIGURE 17. Transfer Performance of CapsNet and Standard CNN. The figure shows model performance on different testing schemes. The first plot shows the baseline model performance of standard supervised learning on domain B.

with other layers in a complex or fragile way, then such co-adaptation could not be relearned by the upper layers alone [9]. Keeping lower layers fixed and re-adjusting the higher ones breaks the coupling between the layers formed during training. As A1B/B1B have the same process, AnB schemes tend to be more challenging, as the representation fully learned by lower layers would suffer cross-domain readaptation. Cross-domain readaptation can benefit from a representation that contains intermediate-level knowledge relevant to both tasks, e.g., the appearance of individual digits in this experiment.

4) RESULTS AND DISCUSSION

Fig. 17 displays the results of CapsNet and the standard CNN with different transfer test schemes. We have observed from the B1B/B2B results that breaking the coupling between layers significantly reduced the fitness of the standard CNN. It is evident that CapsNet has less "co-adapted" features on successive layers than the standard CNN. Co-adapted means that features were learned across multiple layers with complex processing, which cannot be relearned in a single layer.

The CapsNet representation can be successfully used by newly learned higher layers. When the new task is cross-domain, CapsNet has a minor performance drop. Nevertheless, the CapsNet representations remain satisfactorily relevant on AnB tasks, which supports the claim that the intermediate level capsules can capture knowledge on the appearance of meaningful object parts [5].

IV. CONCLUSION

In this paper, we investigated several important aspects of CapsNet, including model learning, attributes of learned data representations, and generality of the representations. Our tests demonstrate that appropriate routing plays a significant

role in CapsNet training. In the early stage of training, the routing between capsules should contain a level of uncertainty. Early over-confidence regarding the routing tends to impose excessive limits on the training process, which leads to suboptimal models. CapsNet can produce data representation with interesting attributes. To explore such attributes, we especially designed a test using image data on a 2D manifold spanned by geometric transformations. The test shows that, compared to the standard CNN, CapsNet can capture more faithfully the global manifold structures in data. Moreover, following the test protocol in [9], we show that the representation by CapsNet is more transferrable than that by the standard CNN.

REFERENCES

[1] Y. Han, Y. Yang, Y. Yan, Z. Ma, N. Sebe, and X. Zhou, "Semisupervised feature selection via spline regression for video semantic recognition," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 26, no. 2, pp. 252–264, Feb. 2015.

[2] K. He, X. Zhang, S. Ren, and J. Sun. "Deep residual learning for image recognition," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, Jun. 2016, pp. 770–778.

[3] Y. Bengio, "Learning deep architectures for AI," *Found. Trends Mach. Learn.*, vol. 2, no. 1, pp. 120–127, 2009.

[4] P. W. Koh and P. Liang. "Understanding black-box predictions via influence functions," in *Proc. Int. Conf. Mach. Learn.*, Aug. 2017, pp. 1885–1894.

[5] S. Sabour, N. Frosst, and G. E. Hinton "Dynamic routing between capsules," in *Proc. NIPS*, Mar. 2017, pp. 3859–3869.

[6] Y. Bengio, "Practical recommendations for gradient-based training of deep architectures," in *Proc. Neural Netw.: Tricks Trade*, May 2012, pp. 437–478.

[7] A. Paccanaro and G. E. Hinton, "Learning distributed representations of concepts using linear relational embedding," *IEEE Trans. Knowl. Data Eng.*, vol. 13, no. 2, pp. 232–244, Mar. 2001.

[8] Y. LeCun, Y. Bengio, and G. Hinton, "Deep learning," *Nature*, vol. 521, pp. 436–444, May 2015.

[9] J. Yosinski, J. Clune, Y. Bengio, and H. Lipson, "How transferable are features in deep neural networks?" in *Proc. Adv. Neural Inf. Process. Syst.*, Nov. 2014, pp. 3320–3328.

[10] M. Long, Y. Cao, J. Wang, and M. I. Jordan, "Deep transfer learning with joint adaptation networks," in *Proc. ICML*, May 2015, pp. 97–105.

[11] A. Krizhevsky, I. Sutskever, and G. E. Hinton "ImageNet classification with deep convolutional neural networks," in *Proc. NIPS*, Dec. 2012, pp. 1106–1114.

[12] K. Simonyan and A. Zisserman. (2014). "Very deep convolutional networks for large-scale image recognition." [Online]. Available: https://arxiv.org/abs/1409.1556

[13] C. Szegedy, S. Ioffe, V. Vanhoucke, and A. A. Alemi. "Inception-v4, inception-ResNet and the impact of residual connections on learning," in *Proc. AAAI*, Feb. 2017, pp. 4278–4284.

[14] S. Garimella and H. Hermansky, "Factor analysis of auto-associative neural networks with application in speaker verification," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 24, no. 4, pp. 522–528, Apr. 2013.

[15] R. Khosrowabadi, C. Quek, K. K. Ang, and A. Wahab, "ERNN: A biologically inspired feedforward neural network to discriminate emotion from EEG signal," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 25, no. 3, pp. 609–620, Mar. 2014.

[16] R. Girshick, "Fast R-CNN," in *Proc. ICCV*, Dec. 2015, pp. 1440–1448.

[17] W. Liu *et al.*, "SSD: Single shot MultiBox detector," in *Proc. ECCV*, Sep. 2016, pp. 21–37.

[18] J.-T. Chien and Y.-C. Ku, "Bayesian recurrent neural network for language modeling," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 27, no. 2, pp. 361–374, Feb. 2016.

[19] D. P. Barrett, S. A. Bronikowski, H. Yu, and J. M. Siskind, "Driving under the influence (of language)," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 29, no. 7, pp. 2668–2683, Jul. 2018.

[20] K. Jia, D. Tao, S. Gao, and X. Xu, "Improving training of deep neural networks via singular value bounding," in *Proc. CVPR*, Jul. 2017, pp. 4344–4352.

[21] M. Sharif, S. Bhagavatula, L. Bauer, and M. K. Reiter. "Accessorize to a crime: Real and stealthy attacks on state-of-the-art face recognition," in *Proc. ACM SIGSAC*, New York, NY, USA, Oct. 2016, pp. 1528–1540.

[22] N. Papernot, P. McDaniel, I. Goodfellow, S. Jha, Z. B. Celik, and A. Swam, "Practical black-box attacks against machine learning," in *Proc. ACM Asia Conf. Comput. Commun. Secur.*, New York, NY, USA, Apr. 2017, pp. 506–519.

[23] G. E. Hinton, A. Krizhevsky, and S. D. Wang. "Transforming auto-encoders," in *Proc. Int. Conf. Artif. Neural Netw.*, Jun. 2011, pp. 44–51.

[24] S. Sabour, N. Frosst, and G. Hinton, "Matrix capsules with EM routing," in *Proc. ICLR*, Feb. 2018, pp. 1–29.

[25] R. S. S. Phaye, A. Sikka, A. Dhall, and D. Bathula. (2018). "Dense and diverse capsule networks: Making the capsules learn better." [Online]. Available: https://arxiv.org/abs/1805.04001

[26] M. Yang, W. Zhao, J. Ye, Z. Lei, Z. Zhao, and S. Zhang. "Investigating capsule networks with dynamic routing for text classification," in *Proc. EMNLP*, Mar. 2018, pp. 3110–3119.

[27] N. Frosst, S. Sabour, and G. Hinton. (2018). "DARCCC: Detecting adversaries by reconstruction from class conditional capsules." [Online]. Available: https://arxiv.org/abs/1811.06969

[28] K. Duarte, Y. S. Rawat, and M. Shah. "VideoCapsuleNet: A simplified network for action detection," in *Proc. Adv. Neural Inf. Process. Syst.*, May 2018, pp. 7610–7619.

[29] A. Jaiswal, W. AbdAlmageed, Y. Wu, and P. Natarajan, "CapsuleGAN: Generative adversarial capsule network," in *Proc. ECCV*, Jan. 2018, pp. 526–535.

[30] L. Zhang, M. Edraki, and G. J. Qi. "CapProNet: Deep feature learning via orthogonal projections onto capsule subspaces," in *Proc. Adv. Neural Inf. Process. Syst.*, May 2018, pp. 5819–5828.

[31] D. Q. Nguyen, T. Vu, T. D. Nguyen, D. Q. Nguyen, and D. Phung. (2018). "A capsule network-based embedding model for knowledge graph completion and search personalization." [Online]. Available: https://arxiv.org/abs/1808.04122

[32] B. Tang, A. Li, B. Li, and M. Wang, "CapSurv: Capsule network for survival analysis with whole slide pathological images," *IEEE Access*, vol. 7, pp. 26022–26030, 2019.

[33] G. E. Hinton, S. Osindero, and Y.-W. Teh, "A fast learning algorithm for deep belief nets," *Neural Comput.*, vol. 18, no. 7, pp. 1527–1554, 2006.

[34] N. Srivastava, G. Hinton, A. Krizhevsky, I. Sutskever, and R. Salakhutdinov, "Dropout: A simple way to prevent neural networks from overfitting," *J. Mach. Learn. Res.*, vol. 15, no. 1, pp. 1929–1958, 2014.

[35] S. Hochreiter and J. Schmidhuber, "Long short-term memory," *Neural Comput.*, vol. 9, no. 8, pp. 1735–1780, 1997.

[36] H. Kataoka, Y. Miyashita, M. Hayashi, K. Iwata, and Y. Satoh. "Recognition of transitional action for short-term action prediction using discriminative temporal CNN feature," in *Proc. BMVC*, 2016, pp. 1–12.

[37] D. P. Kingma and J. Ba. (2014). "Adam: A method for stochastic optimization." [Online]. Available: https://arxiv.org/abs/1412.6980

[38] D. Wang and Q. Liu, "An optimization view on dynamic routing between capsules," in *Proc. ICLR*, Jun. 2018, pp. 1–4.

[39] D. Rawlinson, A. Ahmed, and G. Kowadlo. (2018). "Sparse unsupervised capsules generalize better." [Online]. Available: https://arxiv.org/abs/1804.06094

[40] Z. Chen and D. Crandall. (2018). "Generalized capsule networks with trainable routing procedure." [Online]. Available: https://arxiv.org/abs/1808.08692

[41] T. Hastie, R. Tibshirani, and H. J. Friedman, *The Elements Statistical Learning: Data Mining, Inference, and Prediction*, 2nd ed. New York, NY, USA: Springer, 2009.

[42] S. R. Kheradpisheh, M. Ghodrati, M. Ganjtabesh, and T. Masquelier, "Deep networks can resemble human feed-forward vision in invariant object recognition," *Sci. Rep.*, vol. 7, p. 32672, Sep. 2015.

[43] A. Karpathy, J. Johnson, and L. F. Fei. (2015). "Visualizing and understanding recurrent networks." [Online]. Available: https://arxiv.org/abs/1506.02078

[44] J. Li, W. Monroe, and D. Jurafsky. (2016). "Understanding neural networks through representation erasure." [Online]. Available: https://arxiv.org/abs/1612.08220

[45] S. J. Pan and Q. Yang, "A survey on transfer learning," *IEEE Trans. Knowl. Data Eng.*, vol. 22, no. 10, pp. 1345–1359, Oct. 2010.

[46] L. Shao, F. Zhu, and X. Li, "Transfer learning for visual categorization: A survey," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 26, no. 5, pp. 1019–1034, May 2014.
[47] C. Finn, P. Abbeel, and S. Levine. "Model-agnostic meta-learning for fast adaptation of deep networks," in *Proc. 34th Int. Conf. Mach. Learn.*, Aug. 2017, pp. 1126–1135.
[48] J. Shen, Y. Qu, W. Zhang, and Y. Yu. (2017). "Adversarial representation learning for domain adaptation." [Online]. Available: http://arxiv.org/abs/1707.01217
[49] C. Cortes, Y. LeCun, and C. J. C. Burges. (1998). *The MNIST Database of Hand-Written Digits*. [Online]. Available: http://yann.lecun.com/exdb/mnist/
[50] A. S. Georghiades, P. N. Belhumeur, and D. Kriegman, "From few to many: Illumination cone models for face recognition under variable lighting and pose," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 23, no. 6, pp. 643–660, Jun. 2001.
[51] Y. LeCun, F. J. Huang, and L. Bottou, "Learning methods for generic object recognition with invariance to pose and lighting," in *Proc. CVPR*, Jun. 2004, pp. 97–104.
[52] A. Krizhevsky, "Learning multiple layers of features from tiny images," Univ. Toronto, Toronto, ON, Canada, Tech. Rep., vol. 1, no. 4, 2009.
[53] R. Mukhometzianov and J. Carrillo. (2018). "CapsNet comparative performance evaluation for image classification." [Online]. Available: https://arxiv.org/abs/1805.11195
[54] C. E. Shannon, "A mathematical theory of communication," *Bell Syst. Tech. J.*, vol. 27, no. 3, pp. 379–423, Jul./Oct. 1948.
[55] L. V. D. Maaten and G. Hinton, "Visualizing data using t-SNE," *J. Mach. Learn. Res.*, vol. 9, pp. 2579–2605, Nov. 2008.
[56] H. Fan, H. Su, and L. J. Guibas, " A point set generation network for 3D object reconstruction from a single image," in *Proc. CVPR*, Jul. 2017, pp. 605–613.

ANCHENG LIN is currently pursuing the B.Eng. degree with the Department of Computer Science, Guangdong Polytechnic Normal University. His research interests include deep neural networks, probabilistic models, machine learning, and computer vision.

JUN LI received the B.S. degree in computer science and technology from Shandong University, Jinan, China, in 2003, the M.Sc. degree in information and signal processing from Peking University, Beijing, China, in 2006, and the Ph.D. degree in computer science from Queen Mary, University of London, London, U.K., in 2009. He is currently a Lecturer with the School of Software and the Centre for Artificial Intelligence, Faculty of Engineering and Information Technology, University of Technology Sydney.

ZHENYUAN MA received the B.S. degree from Jinan University, Shandong, China, in 2002, the M.S. degree from Jinan University, Guangdong, China, in 2005, and the Ph.D. degree from the South China University of Technology, Guangdong, China, in 2009, all in computer science. He is currently an Associate Professor with the School of Mathematics and System Sciences, Guangdong Polytechnic Normal University, China. He is also a MOOC Provider and a MOOC Theory Researcher. His current research interests include deep learning, evolutionary algorithm, and intelligent control.