An Interpretable Deep Architecture for Similarity Learning Built Upon Hierarchical Concepts

Xinjian Gao, Tingting Mu, *Member, IEEE,* John Y. Goulermas, *Senior Member, IEEE,*
Jeyarajan Thiyagalingam *Member, IEEE,* Meng Wang, *Senior Member, IEEE*

***Abstract*—In general, development of adequately complex mathematical models, such as deep neural networks, can be an effective way to improve the accuracy of learning models. However, this is achieved at the cost of reduced post-hoc model interpretability, because what is learned by the model can become less intelligible and tractable to humans as the model complexity increases. In this paper, we target a similarity learning task in the context of image retrieval, with a focus on the model interpretability issue. An effective similarity neural network (SNN) is proposed to offer not only to seek robust retrieval performance but also to achieve satisfactory post-hoc interpretability. The network is designed by linking the neuron architecture with the organization of a concept tree and by formulating neuron operations to pass similarity information between concepts. Various ways of understanding and visualizing what is learned by the SNN neurons are proposed. We also exhaustively evaluate the proposed approach using a number of relevant datasets against a number of state-of-the-art approaches to demonstrate the effectiveness of the proposed network. Our results show that the proposed approach can offer superior performance when compared against state-of-the-art approaches. Neuron visualization results are demonstrated to support the understanding of the trained neurons.**

***Index Terms*—Similarity learning, neural networks, clustering, image retrieval, model interpretability.**

I. Introduction

In many applications that leverage artificial intelligence, such as person (re)identification [1], object detection [2], super-resolution imaging [3], image/video retrieval [4], learning an accurate similarity for quantifying the relevance between objects is very crucial. This is usually referred to as (dis)similarity learning [5]. Conventional machine learning approaches usually attempt to formulate the similarity measure using the Mahalanobis distance [6] or a kernel function [7], which is parameterized on a set of variables such as covariance matrix or the kernel parameters. However, the expressive power of such approaches can be limited when processing complex relations and data patterns, and it is therefore necessary to seek techniques to construct more robust similarity learning models.

Neural networks offer a range of powerful modeling techniques for representation and similarity learning [8]–[10]. They are effective at processing real-world data that contains complex patterns, redundant information and noise. Most of the deep learning models used in computer vision for learning image representations [11] and for modeling relevance information [12] are based on the convolutional neural networks (CNNs) [13]. To take the heterogeneous properties of the given objects in account, multi-view techniques can be employed to combine different feature extraction methods and fuse information collected from multiple resources or views [14]. Among these, assuming complementary information across different views is proven to be a viable avenue for improving the performance of machine learning [15]. This motivates various deep learning models to generate multi-view representations [16]–[18]. A brief review on this is provided in Section II-A.

There are cases where two objects are connected under different modalities corresponding to different relation types. For instance, in an image retrieval task, the given query image is an apple in the format of fruit. In the retrieved images, the users may like to find not only the images of apple as a fruit, but also the images of apple juice, apple trees, or even the eponymous company. These images are similar to each other under the root relation “apple”, but they are dissimilar in terms of their specific content. Therefore, these images are deemed to be connected under the different modalities of “juice”, “tree” and “company”. To accommodate this phenomenon, various multimodal similarity learning algorithms have been developed, e.g., by using different kernel functions [19], base metrics [20], transformation functions [21] or distributed relation measures across multiple dimensions [22] in order to model such diverse relation modalities. A brief review on this is provided in Section II-B.

As can be found in the literature, by developing adequately complex (yet elegant) mathematical models, the performance of the underlying model can be improved, e.g., in terms of accuracy, precision and recall related measures. Examples of such technique include deep neural networks and multimodal architectures. However, as the model becomes more complex, what is learned often becomes less tractable and intelligible to humans. In other words, the post-hoc interpretability, the ability to explain the predictions made by a model without elucidating its internal mechanisms [23], becomes limited. For instance, it is not straightforward to explain the meaning of the different modalities learned and represented by the different functions or metrics in an image retrieval task, as a multiple kernel similarity learning algorithm does not necessarily relate

X. Gao and M. Wang are with the School of Computer Science and Information Engineering, Hefei University of Technology, Hefei, 230009, China. Email: gao_xinjian@outlook.com, eric.mengwang@ gmail.com.

T. Mu is with the School of Computer Science, The University of Manchester, Kilburn Building, Oxford Road, Manchester, UK, M13 9PL. Email: tingting.mu@manchester.ac.uk.

J. Y. Goulermas is with the School of Electrical Engineering, Electronics and Computer Science, The University of Liverpool, Liverpool, UK, L69 3BX. Email: j.y.goulermas@liverpool.ac.uk.

J. Thiyagalingam is with the Science and Technology Facilities Council (STFC), Rutherford Appleton Laboratory, Harwell Campus, Didcot, OX11 0QX. Email: t.jeyan@stfc.ac.uk

a "juice" modality to a kernel instance. Moreover, the post-hoc rationalization of a model (e.g., what else can a model reveal) can become important in issues related to model transparency, trust and debugging. This can not only promote the use of machine learning models in real-world applications but can also improve the overall performance. As such, this has been recognized as a major research challenge in the machine learning community and related areas in data analytics.

In this work, focusing on the image retrieval task, we attempt to improve the interpretability of a similarity learning system in addition to improving its accuracy. We propose a similarity neural network (SNN) built upon novel designs of interpretable network architecture and neuron operations, and propose various approaches for visualizing and understanding the SNN neurons. In each layer, each hidden neuron corresponds to a relevance modality. This modality can be interpreted by a semantic concept that is either verbally explainable or visually observable, or both. In this way, what is learned by the network, like how two images are related, can effectively be communicated to the end user. The concepts that the neurons are bounded to either exist in an already known knowledge-base or are automatically extracted from an image collection through clustering. The output of the system exhibits an accumulation of the relevance messages passed from the leaf neurons to the parent neurons, which can be viewed as a distributed similarity over a semantic hierarchical concept tree. The effectiveness of the proposed method is demonstrated through an evaluation and comparative analysis against various state-of-the-art methods using a number of datasets. Our evaluations show that the proposed model can offer improved model interpretability compared to a number of different state-of-the-art approaches. We also make a number of observations at the intermediate neuron level which confirm these improvements.

The remaining part of this paper is organized as follows. Section II briefly reviews some representative works on representation learning, similarity learning and the interpretability of deep learning in the context of image retrieval. The research motivation, proposed model and various neuron visualization approaches are discussed in Section III. Finally, Section IV evaluates, compares and analyzes the proposed methods, while Section V concludes the work.

II. Related Work

A. Deep Representation Learning

In the past decade, deep learning has achieved notable success in data representation learning [8]–[10]. In particular, CNNs have been shown to be effective in a number of computer vision tasks [13], [24]–[27]. A typical approach for image representation learning is to first pre-train a CNN using a large labeled image corpus, and then transfer the learned representations to facilitate other visual recognition tasks that may be lacking of training data through network weight initialization [24]. The CNN can also be used in unsupervised settings [28] or by generating surrogate tasks [25]. A CNN can also be adapted to learn video representations, for example, by forcing the image representation of a frame patch to be close enough to that of a patch from the same track in a video [26]. An alternative way of applying CNNs to learn video representations is to treat the video as a sequence of image frames and process it using hybrid networks containing both CNNs and recurrent neural networks (RNNs) [29]. To capture visual saliency, deep CNNs can be used to extract multi-scale features and generate saliency maps from image regions [27].

The task of mining complementary information across different views is often referred to as multi-view representation learning. This has facilitated complex data analysis in areas such as video surveillance, multimedia, and image classification [16]–[18]. A thorough survey on multi-view representation learning is provided in [30], and it covers both shallow and deep architectures.

B. Multimodal Similarity Learning

In general, multimodal similarity (or distance metric) learning refers to the methodology of measuring the (dis)similarity between objects from different perspectives. A classical approach is the online multiple kernel similarity learning [19]. This takes into account the multimodal image connections using multiple kernels that correspond to different relation modalities. Another typical approach is the transfer distance metric learning [20], which encodes the multimodal connections between objects using multiple base metrics. An alternative way for establishing multimodality is through the direct use of multiple feature spaces [31]. By treating different metrics as different modalities, [32] proposes the cross-diffusion method to compute an enhanced metric from multiple given metrics in an unsupervised way. Lately, various multimodal neural networks have been developed to model interactions between objects. For instance, the multi-manifold distance metric learning for image set classification [33] trains different neural networks for different manifolds corresponding to different classes, then computes a combined distance between two image sets over these manifolds.

There are also works on learning from multimodal data where the different data modalities refer to different data types, such as image, video, audio and text, used to describe the given objects [34]–[37]. A survey on cross-modal retrieval is provided in [38].

C. Interpretability of Deep Learning

Given the recent success of deep learning techniques in AI tasks, it has become more and more important to understand what is going on inside a black-box deep learning model, and this leads to the development of the relatively new research topic on interpretability of deep learning. The goal is to provide straightforward explanations to the modeling processes, intermediate results or the final output of deep learning algorithms. These explanations should be easily understood by algorithm users who are not experts in mathematical modeling. In computer vision, most interpretability works are focused on the alignment between image features and understandable semantic topics, aiming at explaining the meaning of the learned image representations by these topics. For instance, [39] aligns the hidden units in each intermediate layer to human-interpretable concepts. It shows that interpretable units indicate a partially disentangled representation and confirms that representation at different layers disentangle different categories of meaning. The work in [40] attempts to show how human-defined concepts are related to the prediction by automatically collecting image segments that form the human concepts. It validates the learned segments from concepts as coherent as human-labeled concepts and these concepts are often carry sufficient information to be predicted as the corresponding class. In [41], semantic topics are extracted from human descriptions that cover a wide range of visual concepts. An interpretive loss is proposed to integrate these topics into the model. Focusing on visual question answering, [42] proposes a network that is able to quantitatively evaluate the interpretability of the visual attention mechanisms and examines whether the intermediate outputs visually highlight the correct regions of the input images.

There are also works proposed to improve the interpretability of the model architecture. For instance, the input gradient regularization in [43] changes the shape of the decision boundaries that improves the interpretability of adversarial perturbations and leads to interpretable errors rated by humans. In [44], a self-explaining model is proposed, providing explanations to the prediction as a part of the linear classifier models. The input features are anchored by available observations, while a quantitative contribution measure is provided by the model parameters for the corresponding feature and predicted value. A tree-regularization technique is proposed to approximate complex decision boundaries of any differentiable models by human based simulation functions [45]. This allows the domain experts to understand the role of a complex model. Some relevant survey and overview works on interpretability in deep learning can be found in [23], [46], [47].

III. Proposed Method

A. Background and Motivation

Given a collection of n images $\{I_i\}_{i=1}^n$, each represented by matrices of pixel values collected from multiple channels (e.g., colors R, G, B), a general way to establish the similarity formulation between two images is

$$s_{ij} = f\left(\boldsymbol{\phi}\left(I_i, \boldsymbol{\eta}\right), \boldsymbol{\phi}\left(I_j, \boldsymbol{\eta}\right), \boldsymbol{\theta}\right), \tag{1}$$

where $\boldsymbol{\phi}(\cdot)$ denotes a representation function used to extract the high-level features for characterizing the input image, $f(\cdot,\cdot)$ a function for computing the similarity score between two image representations, $\boldsymbol{\eta}$ and $\boldsymbol{\theta}$ are the function parameters. This formulation can be used to unify many existing similarity learning models. Classical ways for obtaining $\boldsymbol{\phi}(\cdot)$ are based on image feature extractors such as local binary pattern, colour histogram, bag of visual words, etc. Typical mechanisms for modelling $f(\cdot,\cdot)$ include the construction of suitable kernel or distance functions as in kernel or distance metric learning.

A more powerful way for modeling $\boldsymbol{\phi}(\cdot)$ and $f(\cdot,\cdot)$ is neural networks. For instance, a CNN can be used to learn the image representation vector. By taking the generated vectors as the input, another neural network can be used to model $f(\cdot,\cdot)$. Past studies have shown that competitive performance gains can be obtained by using neural networks to model functions, but the common criticism is that what is learned in this way is often not interpretable [48]. Recent works attempt to interpret and understand a trained CNN by, for example, displaying the weights of the learned convolutional filters [49], observing images that maximally activate a neuron [50], or reconstructing the images based on learned representations [51]. More generally, embedding methods such as t-SNE [52] can be used to visualize representation vectors returned by any type of neural network in a two-dimensional space. However, there has not been work on improving the interpretability of a similarity learning network typically used for encoding the relevance between objects, or on visualizing and understanding such a network.

We start from some straightforward similarity network construction. The simplified notation $\boldsymbol{\phi}_i \in R^k$ is used to denote the k-dimensional high-level feature vector $\boldsymbol{\phi}_{\text{CNN}}\left(I_i, \boldsymbol{\eta}\right)$ generated by a CNN for the i-th image, which is referred to as the *neural content code*. All the vectors in this paper are column vectors unless stated otherwise. The content codes $\boldsymbol{\phi}_i$ and $\boldsymbol{\phi}_j$ are fed into another neural network to produce the image relevance information, which is encoded by a real-valued d-dimensional vector $\boldsymbol{r}_{ij} \in R^d$, referred to as the *neural relevance code*. The most straightforward way for computing the relevance codes is probably to employ a multilayer perceptron (MLP) to map the concatenated column vector $\boldsymbol{\phi}_{ij} = [\boldsymbol{\phi}_i^T, \boldsymbol{\phi}_j^T]^T$ to the relevance vector $\boldsymbol{r}_{ij}$, where the MLP weights are stored in the vector $\boldsymbol{\varpi}$. Usually, such an MLP can be trained by minimizing a linear ranking loss function as follows

$$L(\boldsymbol{\varpi}, \boldsymbol{w}) = \sum_{\substack{ij_+ \in I_+ \\ ij_- \in I_-}} \max\left(\boldsymbol{w}^T \boldsymbol{r}_{ij_-}(\boldsymbol{\varpi}) - \boldsymbol{w}^T \boldsymbol{r}_{ij_+}(\boldsymbol{\varpi}) + 1, 0\right), \tag{2}$$

with respect to the MLP weights $\boldsymbol{\varpi}$ and the prediction weights $\boldsymbol{w}$[1]. The index sets I_+ and I_- contain respectively the truly related and unrelated image pairs in the training set (referred to as the positive and negative training pairs). After completing this training process, the only understanding gained is that there exists a powerful nonlinear mapping from $\boldsymbol{\phi}_{ij}$ to $\boldsymbol{r}_{ij}$ which creates a high-level d-dimensional relevance space, where the relevant and irrelevant image pairs can be well separated. The linear ranking loss based training results in a linearly separable relevance space. Other than this, deeper interpretation of what is learned by the MLP is not possible. For instance, the meaning of the relevance dimension and the meaning of the hidden neuron output are both unknown.

We attempt to improve this process, and our core strategy is to link the network architecture with human understandable concepts. In general, the human understanding of concepts is naturally hierarchical. For instance, the appearance of Maybach in an image could trigger a series of concepts like Maybach → car → motor vehicle → vehicle → artifact → physical object. This has laid the foundation for some early

[1] To match the similarity template as in Eq. (1), $s_{ij} = \boldsymbol{w}^T \boldsymbol{r}_{ij}(\boldsymbol{\varpi})$ and $\boldsymbol{\theta}^T = \left[\boldsymbol{w}^T, \boldsymbol{\varpi}^T\right]$.

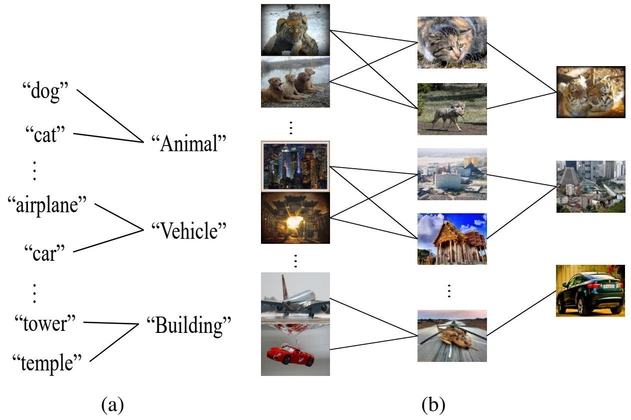


Fig. 1. Examples of concept hierarchies: (a) word-based concepts and (b) image cluster-based concepts.

developments in visual object recognition [53] which improves the model by encoding hierarchical semantic knowledge in a lexical network (e.g., WordNet). Inspired by this, we propose to equip the network with interpretability by matching its neuron connection with a hierarchical semantic knowledge tree structure. The relevance vector computed in each hidden layer is a vector distributed over different relation modalities, each corresponding to a hidden neuron. In this way, the meaning of each relation modality can be naturally explained by the semantic concept that the corresponding hidden neuron is attached to. It is worth to mention that, when the semantic knowledge tree is constructed using an information resource external to the training image collection, the network finally infuses information offered by both the image content and the external knowledge. This results in an alternative modality view, where the two data types are treated as the two data modalities to be combined by the model.

B. Model Construction

As mentioned above, by adopting a concept hierarchy suitable for a given image collection, we can associate each concept in the hierarchy with a neuron in the hidden layers and force the neuron connectivity to match the concept organization hierarchy. As such, the information flow between layers of neurons can be explained as a message passing process from low-level to high-level concepts, where each concept is interpretable with known characteristics. However, this instigates various research questions, including (1) how to define a meaningful concept and a concept hierarchy, (2) how to associate a neuron with a specific concept, (3) how to encode a message to carry the relevance information, and also (4) how to model the message passing action between concepts. We will answer these questions below.

1) Concept Hierarchy: A straightforward way to obtain a concept hierarchy is to seek support from existing knowledge bases that are relevant to the image collection. For instance, WordNet provides a semantic network of general concepts in language, and it can be used to generate a concept hierarchy suitable for general image collections by exploiting the hypernym-hyponym relations between nouns [54], [55]. These concepts correspond to descriptive words and are naturally interpretable. Fig. 1(a) demonstrates a simple two-level concept hierarchy where each concept is described by a single word.

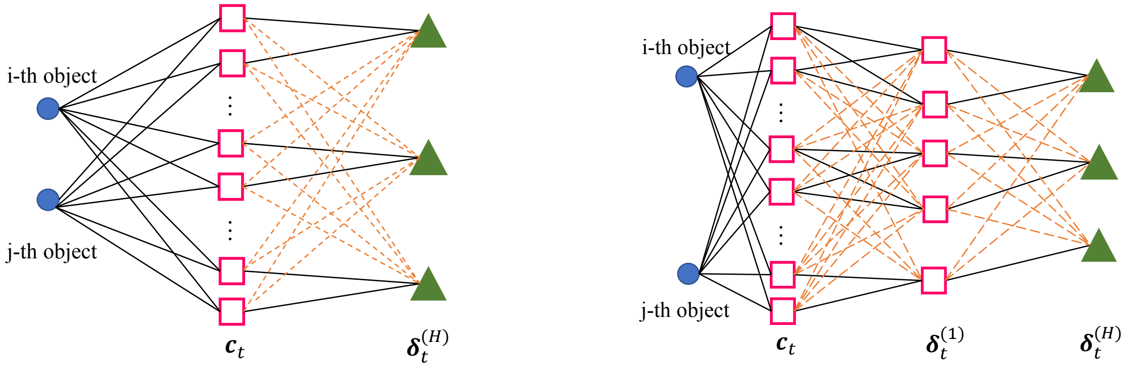
When there is no explicit external knowledge available to help building a concept hierarchy, an alternative way is to seek latent topics contained within the images by exploring their visual content, which usually corresponds to clusters of images. Such an approach is based on the assumption that interactions between images exhibit the same underlying topic structure as that revealed by individual images. A similar strategy has been pursued in [56], which assumes that both individual documents and document pairs are generated from the same set of topic distributions. The cluster-based concept hierarchy can be obtained by applying a hierarchical clustering algorithm [57] over the high-level representations of images $\{\boldsymbol{\phi}_i\}_{i=1}^{n}$ returned by the CNN network. The resulting concepts correspond to image clusters and are therefore naturally observable. Fig. 1(b) demonstrates a three-level concept hierarchy based on image clusters, where each concept is represented by an image that is closest to the cluster center in the CNN feature space.

The hierarchical organization of the extracted concepts can be modelled by a tree structure. Let l_0 denote the number of leaf concepts, corresponding to level 0. The h-th level contains the parents of the concepts from the previous level, and the number of parent concepts in the h-th level ($h = 1, 2, \ldots, H$) is denoted by l_h. The number of concepts in the last level controls the dimensionality of the computed relevance vector such as $l_H = d$. Each leaf concept is characterized by a *leaf concept representation vector* denoted by $\boldsymbol{c}_t \in R^k$ (for $t = 1, 2, \ldots, l_0$). Its dimensionality is set to be the same as that of the image representation $\boldsymbol{\phi}_i \in R^k$ for the convenience of modelling the relevance message as in Eqs. (5) -(7) later. When there exists a one-to-one mapping between a concept and a word, $\boldsymbol{c}_t$ can be set as the corresponding word embedding vector computed using a neural language model trained with a large text corpus, such as GloVe word embeddings trained using Wikipedia text [58]. This vector uniquely encodes the semantic meaning of the word. When the image clusters are used as the concepts, $\boldsymbol{c}_t$ can be set as the averaged CNN representation of the member images. The parent concepts from the same level are uniquely distinguishable by their connections to the concepts from the previous level. For instance, the t-th concept from the h-th level can be characterized by the binary row vector $\boldsymbol{\delta}_t^{(h)} = [\delta_{t1}, \ldots, \delta_{tq}, \ldots, \delta_{tl_{h-1}}]$, where each element indicates whether it is the parent of the concept q from the previous level $h-1$; we refer to this as the *parent concept vector*. It has to be mentioned that the leaf concepts correspond to specific patterns characterized by explicit feature representations as in $\{\boldsymbol{c}_t\}_1^{l_0}$, while the parent concepts are more abstract patterns characterized by links to their child concepts.

2) Concept Neuron Association and Message Passing: Our goal is to design a neural network so that its information flow between neurons can be explained as a message passing process between either verbally explainable (or visually observable) concepts. Therefore, we force the neuron organization hierarchy to be identical to a concept organization hierarchy. Each object pair is connected to all leaf neurons. Specifically,

TABLE I
A LIST OF MAIN NOTATIONS USED IN THE PROPOSED SNN MODEL.

Variable Name	Variable Description
$\boldsymbol{\phi}_i \in R^k$	The ith image's representation vector generated by CNN, $i = 1, 2, 3 \ldots, n$.
$\boldsymbol{r}_{ij} \in R^d$	The output relevance vector between the ith and jth images, where $\boldsymbol{r}_{ij} = \left[r_{ij}^{(1,H)}, r_{ij}^{(2,H)}, \ldots, r_{ij}^{(l_H,H)}\right]^T$ by Eq. (9) and $l_H = d$.
$\boldsymbol{c}_t \in R^k$	The tth real-valued leaf concept vector at level 0 (1st hidden layer) , $t = 1, 2, \ldots, l_0$.
$\boldsymbol{\delta}_t^{(h)} \in [0, 1]^{l_{h-1}}$	The tth binary parent concept vector at level h ($h + 1$-th hidden layer), $t = 1, 2, \ldots, l_h$ and $h = 1, 2 \ldots H$.
$\mathbf{F}_t \in R^{k \times k_F}$	Weights of the hidden neuron corresponding to the tth leaf concept, to be optimized, $t = 1, 2, \ldots, l_0$.
$\boldsymbol{\alpha}, \boldsymbol{\beta} \in R^k, b \in R$	Weights shared by hidden neurons corresponding to all the leaf concepts, to be optimized.
$\boldsymbol{w}_t^{(h)} \in R^{l_{h-1}}, b_t^{(h)} \in R$	Weights of the hidden neurons corresponding to the tth parent concept at level h, to be optimized, $t = 1, 2, \ldots, l_h$ and $h = 1, 2 \ldots H$.


i-th object
j-th object
$\boldsymbol{c}_t$
$\boldsymbol{\delta}_t^{(H)}$
i-th object
j-th object
$\boldsymbol{c}_t$
$\boldsymbol{\delta}_t^{(1)}$
$\boldsymbol{\delta}_t^{(H)}$

(a) The case from Fig. 1(a).

(b) The case from Fig. 1(b).

Fig. 2. Illustration of the network architectures built upon the two example cases as shown in Fig. 1. The dashed lines correspond to the eliminated neuron connections due to the concept hierarchy matching.

the t-th concept in the h-th level corresponds to the t-th neuron in the $(h + 1)$-th hidden layer, where $h = 0, 1, \ldots, H$. In Fig. 2, we demonstrate the network architectures built upon the two example cases in Fig. 1. It can be seen that the concept hierarchy results in sparse connections between the hidden layers, where neuron connections in dashed lines are removed to ensure alignment between the neurons and the pre-constructed concepts.

Given $\{\boldsymbol{c}_t\}_{t=1}^{l_0}$ and $\{\boldsymbol{\delta}_t^{(h)}\}_{t=1}^{l_h}$ $(h = 1, 2, \ldots, H)$ characterizing the leaf and parent concepts, we continue to explain how to model a message carrying the relevance information between two images and how it is communicated between two concepts. The input layer of our SNN contains a pair of image vectors $\boldsymbol{\phi}_i$ and $\boldsymbol{\phi}_j$ that are computed by a CNN. A similarity score between an input image and the leaf concept can be simply modelled by $\boldsymbol{\phi}_i^T \boldsymbol{c}_t + b$, where b is a bias parameter. Two similar images with their relevance triggered specifically by the concept $\boldsymbol{c}_t$ would be expected to have a high score of

$$s = \left(\boldsymbol{\phi}_i^T \boldsymbol{c}_t + b\right) \left(\boldsymbol{\phi}_j^T \boldsymbol{c}_t + b\right) \tag{3}$$

$$= \left(\boldsymbol{\phi}_i^T \boldsymbol{c}_t\right) \left(\boldsymbol{\phi}_j^T \boldsymbol{c}_t\right) + b\boldsymbol{c}_t^T \boldsymbol{\phi}_i + b\boldsymbol{c}_t^T \boldsymbol{\phi}_j + b^2. \tag{4}$$

Therefore, a naive way of formulating the relevance message received by a leaf concept $\boldsymbol{c}_t$ can be

$$r_{ij}^{(t,0)} = \ln\left(1 + \exp\left[\left(\boldsymbol{\phi}_i^T \boldsymbol{c}_t\right) \left(\boldsymbol{\phi}_j^T \boldsymbol{c}_t\right) + b\boldsymbol{c}_t^T \boldsymbol{\phi}_i + b\boldsymbol{c}_t^T \boldsymbol{\phi}_j + b^2\right]\right), \tag{5}$$

for $t = 1, 2, \ldots, l_0$. Here, a smoothed version of the rectifier function is used to avoid negative relevance messages which are unnatural to explain. The leaf concept $\boldsymbol{c}_t$ is fixed as either a word embedding vector or a cluster center vector. As a result, only when two images are similar to $\boldsymbol{c}_t$ at the same time, Eq. (5) will return a high relevance score. This obviously does not encourage the model to recognize a wider range of relevance patterns.

To improve the model expressive power, we equip the model with more parameters but still maintain a similar formulation template to Eq. (5). This results in the following modification:

$$r_{ij}^{(t,0)} = \ln\left(1 + \exp\left[\left(\boldsymbol{\phi}_i^T \boldsymbol{c}_t\right) \left(\boldsymbol{\phi}_j^T \boldsymbol{c}_t\right) + \boldsymbol{\alpha}^T \boldsymbol{\phi}_i + \boldsymbol{\beta}^T \boldsymbol{\phi}_j + b\right]\right). \tag{6}$$

The usage of the weight vectors $\boldsymbol{\alpha}, \boldsymbol{\beta} \in R^k$ allows to take into account the characteristic of each individual image that is not necessarily tied to $\boldsymbol{c}_t$. However, the fixed quantity $\boldsymbol{\phi}_i^T \left(\boldsymbol{c}_t \boldsymbol{c}_t^T\right) \boldsymbol{\phi}_j$ still limits the diversity of the relevance patterns that could be captured by the model. Building upon Eq. (6), a further modification is applied, resulting in

$$r_{ij}^{(t,0)} = \ln\left(1 + \exp[\boldsymbol{\phi}_i^T \mathbf{F}_t \mathbf{F}_t^T \boldsymbol{\phi}_j + \boldsymbol{\alpha}^T \boldsymbol{\phi}_i + \boldsymbol{\beta}^T \boldsymbol{\phi}_j + b]\right), \tag{7}$$

where the elements of the $k \times k_F$ matrix $\mathbf{F}_t$ are treated as the model variables to be optimized, but its initialization is controlled by the center vector $\boldsymbol{c}_t$. Specifically, the first column of $\mathbf{F}_t$ is initialized by the normalized center unit length vector, set as $\mathbf{F}_t^{(0)}(:, 1) = \frac{\boldsymbol{c}_t}{\|\boldsymbol{c}_t\|}$, and the remaining $k_F - 1$ columns of $\mathbf{F}_t$ are initialized by $k_F - 1$ randomly generated orthonormal vectors that span the nullspace of $\boldsymbol{c}_t \boldsymbol{c}_t^T$. As a result, the hidden neurons corresponding to the leaf concept are finally characterized by the optimized matrix $\mathbf{F}_t$. Such a relaxation allows a deviation from $\boldsymbol{c}_t$ with k_F controlling the amount of the allowed deviation. The full parametric formulation in Eq. (7) is expected to model a wider range of similarity patterns between images, but meanwhile the initialization by $\boldsymbol{c}_t$ attempts to maintain a connection between the neurons and the pre-constructed concepts. Specifically, the pre-constructed concepts largely affect which local optimum to be arrived during the training.

Regarding to the messages received by a parent concept, it is natural to assume that a parent concept can receive only the messages passed by its child concepts. Based on this, an accumulated message received by the t-th concept at level h can be formulated as

$$r_{ij}^{(t,h)} = \frac{1}{1 + \exp\left[-\sum_{q=1}^{l_{h-1}} w_{tq}^{(h)} \delta_{tq}^{(h)} r_{ij}^{(q,h-1)}\right]}, \tag{8}$$

where $w_{tq}^{(h)}$ is the composition weight with $t = 1, 2, \ldots, l_h$ and $h = 1, 2, \ldots, H$. Employing the vector notations $\boldsymbol{w}_t^{(h)} = [w_{t1}^{(h)}, w_{t2}^{(h)}, \ldots, w_{tl_{h-1}}^{(h)}]$ and $\boldsymbol{r}_{ij}^{(h)} = [r_{ij}^{(1,h)}, r_{ij}^{(2,h)}, \ldots, r_{ij}^{(l_h,h)}]$, Eq. (8) can be expressed as

$$r_{ij}^{(t,h)} = \frac{1}{1 + \exp\left[-\boldsymbol{r}_{ij}^{(h-1)}(\boldsymbol{w}_t^{(h)} \circ \boldsymbol{\delta}_t^{(h)})^T - b_t^{(h)}\right]}, \tag{9}$$

where $b_t^{(h)}$ is the bias parameter. The sigmoid function is used to ensure non-negativity of the message.

Here, we employ the rectifier function to encode the message passed from the input images to a leaf concept as in Eq. (7), while the sigmoid function from the leaf concepts to a parent concept as in Eq. (9). This is because we attempt to introduce higher sparsity in the leaf messages than in the parent messages, as it is more meaningful to select fewer but more distinctive leaf concepts to represent the low-level similarity patterns between images.

3) Layer-wise Learning: The complete similarity learning process includes the training of a CNN to obtain the neural content code for characterizing an image, the training of the proposed SNN to obtain the neural relevance code characterizing the image relevance, and finally a fine-tuning process for optimizing the two connected CNN-SNN networks together. Existing works [59] have shown that, nowadays, although deep neural networks are gradually replacing hand-crafted feature extraction, they do not encourage the use of expert knowledge, for instance, provided by existing feature extraction methods, to enhance the learning. To improve this, we have previously proposed an unsupervised multi-view training algorithm in [22], [60]. It pre-trains a CNN to preserve knowledge offered by multiple image feature extraction methods that characterize heterogeneous properties of the image content. To facilitate the practitioners, we explain in Appendix A how the multi-view pre-training works. By taking the image representation vector $\boldsymbol{\phi}_i$ computed by the CNN as the input, we further optimize the SNN model variables by minimizing the loss function in Eq. (2) through stochastic gradient descent, where $\boldsymbol{\varpi}$ is used to store the SNN variables instead. So far, the separate training of CNN and SNN divides the model architecture into two independent components: (1) the unsupervised feature learning, and (2) the supervised relation learning. A fine-tuning procedure is conducted based on the ranking loss, to further jointly optimize all the model variables, including $\boldsymbol{\eta}$ for CNN, $\boldsymbol{\varpi}$ for SNN and $\boldsymbol{w}$ for relevance prediction.

When training using the ranking loss, the derivatives are computed based on a positive pair and a negative pair in each update. The two pairs are denoted by $(\text{obj}_i, \text{obj}_{j+})$ and $(\text{obj}_i, \text{obj}_{j-})$, where $_{ij+} \in I_+$ and $_{ij-} \in I_-$. In general, I_+ can be set as a collection of image pairs where each pair contains a query image and one of its correct images to be retrieved. While, I_- can be set as the collection where each pair includes a query image and an image that should not be retrieved given this query. Given a training set containing n candidate images for retrieval and m query images, we let n_i denote the number of images that are related to the ith query image according to the ground truth information. This subsequently results in n_i positive pairs and $n - n_i$ negative pairs for the ith query, and in total $\sum_{i=1}^m n_i$ positive examples and $nm - \sum_{i=1}^m n_i$ negative examples for training the similarity score. Given large n and m, the amount of positive and negative example pairs can become extremely large. It is not practical to use all the available pairs for training and a random pair selection is usually applied to improve the learning speed. However, this is not effective. Since the goal of the training procedure is to move the truly related objects closer and push the unrelated ones farther away, we attempt to pay more attention to those object pairs that are more challenging to learn. These include pairs containing objects that appear proximate (distant) but are actually unrelated (related). One convenient way to achieve this, is to first seek the nearest neighbors for each object; for instance, to search the neighbors of each query image among the n candidate images. Then, the two objects that are directed neighbors and related to each other are treated as a friend pair, while those that are directed neighbors but are unrelated are treated as an enemy pair. Subsequently, a random subset from the friend pairs are used as I_+, while a random subset from the enemy pairs as I_-. The ratio between the sizes of I_+ and I_- is introduced as a user-set parameter to control the learning.

C. Visualizing What is Learned by the SNN

The proposed SNN attempts to improve the model interpretability by linking the concept hierarchy and the neuron hierarchy (see Fig. 2). The neuron activations are viewed as the message passing operations between concepts as defined in Eqs. (9,5). When the concepts are already associated with descriptive words, they are naturally interpretable. When the concepts are associated with latent patterns corresponding to image clusters, we propose various ways to observe these patterns.

1) Maximal Neuron Activation: A straightforward way to observe the characteristics of each trained neuron in the SNN is through observing the image that maximally activates that neuron given a query image, based on Eqs. (7) and (9). Specifically, given a query image I_q, we seek and identify

$$I = \arg \max_{j \in 1,2,\ldots,n} r_{qj}^{(t,h)}, \tag{10}$$

for the hidden neuron corresponding to the the t-th concept at level h, where $t = 1, 2, \ldots, l_h$ and $h = 0, 1, \ldots, H$.

2) Visualizing Leaf Concepts: The neurons that correspond to the leaf concepts are characterized by the weight matrix $\mathbf{F}_t$. Each column is associated with a k-dimensional embedding vector such that $\mathbf{F}_t = [\boldsymbol{f}_t^{(1)}, \boldsymbol{f}_t^{(2)}, \ldots, \boldsymbol{f}_t^{(k_F)}]$. Since the first term in Eq. (7) can be re-written as

$$\boldsymbol{\phi}_i^T \mathbf{F}_t \mathbf{F}_t^T \boldsymbol{\phi}_j = \sum_{m=1}^{k_F} (\boldsymbol{\phi}_i^T \boldsymbol{f}_t^{(m)})(\boldsymbol{\phi}_j^T \boldsymbol{f}_t^{(m)}), \qquad (11)$$

the embedding vectors $\{\boldsymbol{f}_t^{(m)}\}_{m=1}^{k_F}$ actually encode the potential common patterns shared between two related images. Therefore, we propose to observe $\boldsymbol{f}_t^{(m)}$ by treating it as a representation vector produced by the trained CNN that is used to generate $\boldsymbol{\phi}_i$, and then inverting it by following the same method proposed in [51]. This results in the optimization problem of finding the optimal pixel values of an image, such that its neural content code is as close to $\boldsymbol{f}_t^{(m)}$ as possible. One way to formulate this problem is

$$\mathbf{X}_{t,m}^* = \arg\min_{\mathbf{X}} \left\| \boldsymbol{\phi}_{\text{CNN}}(\mathbf{X}) - \boldsymbol{f}_t^{(m)} \right\|_2 + \lambda R(\mathbf{X}), \qquad (12)$$

where

$$R(\mathbf{X}) = \sum_{i,j} \left((x_{i,j+1} - x_{i,j})^2 + (x_{i+1,j} - x_{i,j})^2 \right)^{\frac{\beta}{2}}. \qquad (13)$$

The variable $\mathbf{X}$ stores the image pixels to be optimized. The term $R(\mathbf{X})$ is known as the total variance regularizer, which encourages the reconstructed image to contain piecewise constant patches. The user-defined parameter $\lambda > 0$ controls the preference degree over the regularization term and β is usually set to 1. Subsequently, a set of k_F images $\left\{\mathbf{X}_{t,m}^*\right\}_{m=1}^{k_F}$ are recovered for the k_F embedding vectors used by the t-th leaf neuron in the first hidden layer of SNN.

3) Neuron Image Reconstruction: Here, we attempt to propose a unified approach for observing neurons corresponding to both leaf and parent concepts. The activation computed over each neuron $r_{ij}^{(t,h)}$ is a function of the neuron content codes of the input two images, and can thus be written as $r^{(t,h)}(\boldsymbol{\phi}_i, \boldsymbol{\phi}_j)$. Given a query image I_q with its neural content code $\boldsymbol{\phi}_q$ computed by the trained CNN, we propose to generate an optimal neural content code that results in the maximal activation over each neuron by solving the following optimization problem

$$\boldsymbol{\phi}_{t,h}^* = \arg\max_{\boldsymbol{\phi} \in R^k} r^{(t,h)}(\boldsymbol{\phi}_q, \boldsymbol{\phi}). \qquad (14)$$

Then, an inverted image that is visually observable is computed from $\boldsymbol{\phi}_{t,h}^*$ by following

$$\mathbf{X}_{t,h}^* = \arg\min_{\mathbf{X}} \left\| \boldsymbol{\phi}_{\text{CNN}}(\mathbf{X}) - \boldsymbol{\phi}_{t,h}^* \right\|_2 + \lambda R(\mathbf{X}). \qquad (15)$$

This image $\mathbf{X}_{t,h}^*$ reveals what is considered the most similar to the input image by the t-th neuron at the h-th hidden layer, and reflects the characteristic of the target neuron. As compared to the leaf concepts, the parent concepts represent more abstract patterns. They can be understood as an accumulation of the selected leaf patterns and may therefore not appear visually meaningful when observing their computed visual patterns. As an alternative, trees showing connections between their descendant concepts can be a good way to visualize the parent neurons. Discussions on this are provided in Section IV-B.

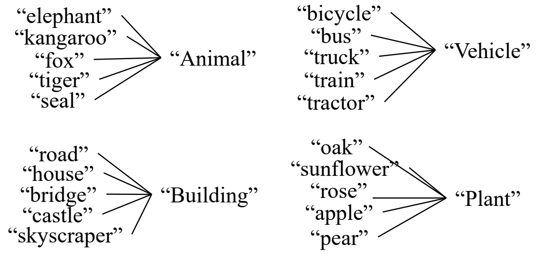


Fig. 4. Illustration of the word-based concept hierarchy used for CIFAR-20, which is manually developed by following a similar class hierarchy as provided in NUS-WIDE.

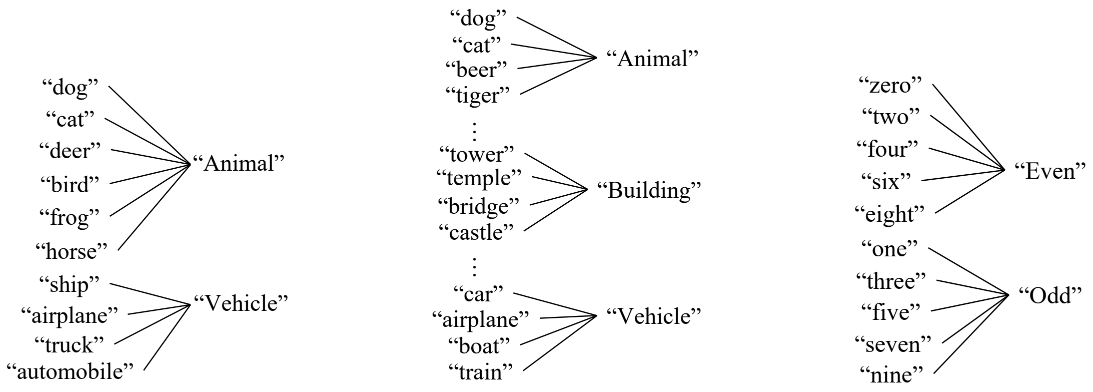
IV. Experimental Results and Analysis

We evaluate the proposed model using a number of different image retrieval tasks. Comparative analysis is performed against various state-of-the-art similarity learning models and popular retrieval techniques in terms of the retrieval performance, which is assessed by the precision of the top 500 retrieved images (500AP) and the mean average precision (MAP). Meanwhile, we provide various examples to demonstrate what is learned by the proposed SNN and how it can be interpreted.

A. Image Retrieval Tasks

1) Datasets and Experimental Setup: Image retrieval is the task of enabling a computer system to search relevant images from a database given a query image, where the retrieval accuracy relies on the quality of the similarity computation between images. In this task, six benchmark datasets are used for evaluation, each containing multiple image classes. The goal is to retrieve all the images belonging to the same class as the given query. The six datasets are:

- CIFAR-10 [62] contains 60,000 color images belonging to 10 object classes such as airplane, truck, bird, cat, deer, horse, etc. Each class contains 6,000 images, among which 1,000 images per class are randomly selected as query images, 1,000 images per class as training images, and all the remaining ones as testing images.
- CIFAR-20 [62] contains 60,000 color images belonging to 20 super classes. Each class contains 3,000 images, among which 200 images per class are randomly selected as query images, 1,000 images per class as training images, while the rest ones as testing images.
- NUS-WIDE [61] is a large collection of flicker web images containing 269,648 images from 81 concepts such as temple, tiger, tower, airplane, boat, etc. In the experiments, 2,000 randomly selected images are used as queries, 5,000 as training images and 260,000 as testing ones to evaluate the retrieval performance.
- MNIST [13] is a handwritten digits dataset containing 60,000 grayscale images representing the ten digits from 0 to 9. We randomly selected 10,000 images as queries, 10,000 images for training and the remaining images as testing ones. Another handwritten digits collection USPS [63], which contains 11,000 grayscale images of 0 to 9, is used to conduct experiment in a transfer learning setup.



(a) CIFAR-10 (b) NUS-WIDE (c) MNIST

Fig. 3. Illustration of the word-based concept hierarchies used for the CIFAR-10, NUS-WIDE and MNIST datasets. We adopt the object class hierarchy provided in the NUS-WIDE image set (see Figure. 3 in [61] for example) as its concept hierarchy, and manually develop a similar concept hierarchy for CIFAR-10. A simple hierarchy based on integer parity in mathematics is used for MNIST images.

- Caltech-UCSD Bird-200-2011 [64] is a bird image collection containing 11,788 images corresponding to 200 kinds of birds, such as sooty albatross, fish crow, gadwall, etc. (referred to as the Caltech-Bird dataset). All bird categories are selected to perform the experiments, where 20 samples per class are randomly selected as the query images, 20 as the training images, and the remaining ones for testing.
- Stanford Dog-120 [65] is a dog image collection containing 20,580 images belong to 120 dog categories, such as dingo, chihuahua, boxer, etc.(referred to as the Stanford-Dog dataset). All categories are selected to perform the experiments, where 20 samples per class are randomly selected as the query images, 100 as the training images, and the remaining ones for testing.

To implement the CNN, we employ two convolutional layers each followed by a max-pooling layer, and finally a fully connected layer[2]. In the first convolutional layer, 5×5 kernel size and $m = 5$ maps are used to construct the convolutional maps, while the same kernel size and $m = 16$ maps in the second convolutional layer. To perform the multi-view pretraining of CNN, for CIFAR-10 and CIFAR-20, 900-D local binary pattern (LBP), 256-D color histogram (CH), 324-D histogram of gradient (HoG) and 1024-D wavelet texture (WT) are used. For NUS-WIDE, 64-D CH,144-D color correlogram (CORR), 73-D edge direction histogram (EDH), 128-D WT, 225-D block-wise color moments (CM) and the 500-D bag-of-word model based on SIFT descriptions are used. For Caltech-Bird and Stanford-Dog, the LBP, CH, HoG and WT features are used. Because images in the MNIST and USPS datasets are grayscale, three grayscale feature extraction methods of 676-D LBP, 784-D WT and 144-D HoG are used. Mini-batch gradient descent is applied for training, where we adopt the setting of batchsize $= 50$ and learning rate $= 0.1$.

TABLE II
COMPARISON OF DIFFERENT METHODS OF FORMULATING LEAF MESSAGES, INCLUDING THE USE OF EQ. (6), THE PROPOSED EQ. (7) INITIALIZED BASED ON $\boldsymbol{c}_t$ FOR DIFFERENT VALUES OF k_F, AS WELL AS EQ. (7) WITH RANDOM INITIALIZATION FOR $k_F = 50$.

Concept	Message	NUS-WIDE		CIFAR-10		CIFAR-20		MNIST	
		500AP (%)	mAP (%)	500AP (%)	mAP	500AP (%)	mAP (%)	500AP(%)	mAP(%)
Word	Eq. (6)	0.68	0.66	0.60	0.70	0.28	0.30	0.92	0.92
Word	Eq. (7), random ini, $k_F = 50$	0.70	0.68	0.66	0.68	0.31	0.30	0.95	0.95
Word	Eq. (7), $\boldsymbol{c}_t$ ini, $k_F = 50$	0.71	0.71	0.72	0.70	0.31	0.33	0.97	0.97
Word	Eq. (7), $\boldsymbol{c}_t$ ini, $k_F = 100$	0.71	0.69	0.72	0.70	0.31	0.31	0.97	0.97
Word	Eq. (7), $\boldsymbol{c}_t$ ini, $k_F = 150$	**0.73**	**0.75**	**0.73**	**0.72**	**0.32**	**0.33**	**0.97**	**0.98**
Word	Eq. (7), $\boldsymbol{c}_t$ ini, $k_F = 200$	0.72	0.72	0.70	0.69	0.30	0.30	0.96	0.96
Word	Eq. (7), $\boldsymbol{c}_t$ ini, $k_F = 250$	0.70	0.70	0.70	0.71	0.28	0.30	0.96	0.96
Cluster	Eq. (6)	0.53	0.68	0.52	0.62	0.22	0.22	0.88	0.85
Cluster	Eq. (7), random ini, $k_F = 50$	0.63	0.70	0.58	0.69	0.24	0.27	0.92	0.94
Cluster	Eq. (7), $\boldsymbol{c}_t$ ini, $k_F = 50$	0.69	0.71	0.69	0.70	0.25	0.27	0.92	0.95
Cluster	Eq. (7), $\boldsymbol{c}_t$ ini, $k_F = 100$	0.71	0.67	0.70	0.70	0.26	0.27	0.91	0.95
Cluster	Eq. (7), $\boldsymbol{c}_t$ ini, $k_F = 150$	0.71	0.71	0.71	0.70	0.27	0.27	0.92	0.96
Cluster	Eq. (7), $\boldsymbol{c}_t$ ini, $k_F = 200$	0.70	0.70	0.70	0.69	0.25	0.25	0.90	0.93
Cluster	Eq. (7), $\boldsymbol{c}_t$ ini, $k_F = 250$	0.68	0.70	0.69	0.69	0.23	0.25	0.89	0.90

For the NUS-WIDE, CIFAR-10, CIFAR-20 and MNIST datasets, we experiment with two ways for constructing the concept hierarchy. This includes a pre-defined concept hierarchy where each concept is described by a word, and a computed concept hierarchy where each concept corresponds to an image cluster. The word-based concept hierarchies used for are displayed in Figs. 3 and 4. These are constructed by considering the class list available in each dataset and the relevant common knowledge. The agglomerative hierarchical cluster tree [68] is used to generate the cluster-based concept hierarchy, where the same value of l_H as that in the word-based concept hierarchy is used to keep consistency. In the case of word-based hierarchy, $\boldsymbol{c}_t$ is set as the GloVe word embeddings, while in the case of cluster-based hierarchy, $\boldsymbol{c}_t$ is set as the cluster centers computed in the CNN feature space. Both the Caltech-Bird and Stanford-Dog datasets are developed for fine-grained image classification. For instance, a single parent concept of "bird" is associated with multiple leaf concepts corresponding to different bird species. This does not make it straightforward to develop a word-based concept hierarchy with layers of parent concepts, so the image cluster based concept hierarchies are used, with $l_H = 4$. In general, when the concept hierarchy is constructed using external knowledge, its depth and concept number is informed by the external knowledge. When the concept hierarchy is constructed using [68], the cluster number (concept number) is determined by the clustering algorithm given a user-specified hierarchy depth l_H.

[2] We aim at testing the proposed SNN model and therefore a simple CNN architecture is employed. In practice, the users can employ any state-of-the-art CNN architectures suitable for their datasets, e.g., ResNet [66], DenseNet [67], etc.

TABLE IV
COMPARISON OF DIFFERENT NETWORK HIERARCHIES FOR SIMILARITY LEARNING.

	NUS-WIDE		CIFAR-10		CIFAR-20		MNIST		MNIST-USPS	
	500AP (%)	mAP (%)	500AP (%)	mAP (%)	500AP(%)	mAP(%)	500AP(%)	mAP(%)	500AP(%)	mAP(%)
cluster-flat	0.71	0.72	0.70	0.68	0.29	0.32	0.96	0.95	0.90	0.87
cluster-fc	0.71	0.72	0.70	0.70	0.29	0.32	0.97	**0.98**	0.95	0.89
cluster-hier	0.73	0.72	0.72	0.72	0.29	0.32	**0.98**	**0.98**	0.95	0.96
word-flat	0.73	**0.75**	0.73	0.72	0.32	0.33	0.97	**0.98**	**0.96**	0.97
word-fc	0.72	**0.75**	0.73	0.74	0.33	**0.35**	0.97	**0.98**	**0.96**	0.96
word-hier	**0.75**	**0.75**	**0.74**	**0.76**	**0.35**	**0.35**	**0.98**	**0.98**	**0.96**	**0.98**

TABLE V
PERFORMANCE COMPARISON FOR DIFFERENT METHODS.

Methods	NUS-WIDE		CIFAR-10		CIFAR-20		MNIST		MNIST-USPS		Caltech-Bird		Stanford-Dog	
	500AP (%)	mAP (%)	500AP (%)	mAP (%)	500AP(%)	mAP(%)	500AP (%)	mAP (%)	500AP (%)	mAP (%)	20AP (%)	mAP (%)	20AP (%)	mAP (%)
OMKS [19]	0.60	0.62	0.58	0.55	0.19	0.23	0.82	0.86	0.77	0.80	0.01	0.03	0.06	0.07
ITQ [69]	0.28	0.28	0.22	0.25	0.15	0.18	0.46	0.52	0.39	0.46	0.03	0.01	0.05	0.07
MAH [70]	0.35	0.32	0.38	0.40	0.16	0.20	0.58	0.64	0.52	0.60	0.02	0.02	0.06	0.10
DRSCH [71]	0.63	0.64	0.65	0.63	0.26	0.27	0.96	0.98	0.93	0.95	0.07	0.06	0.16	0.09
DSRH [72]	0.62	0.63	0.64	0.63	0.26	0.25	0.95	0.96	0.91	0.92	0.06	0.03	0.11	0.11
KSH-CNN [73]	0.62	0.62	0.52	0.47	0.25	0.24	0.85	0.90	0.82	0.87	0.08	0.05	0.18	0.11
NDH [74]	0.30	0.32	0.26	0.32	0.18	0.20	0.52	0.56	0.49	0.51	0.06	0.05	0.08	0.07
MVC-MS [60]	0.68	0.68	0.70	0.67	0.27	0.30	0.94	0.94	0.90	0.92	0.09	0.10	0.20	0.21
CMCQ [75]	0.71	0.72	0.68	0.70	0.29	0.32	0.97	0.96	0.95	0.92	0.10	0.11	0.19	0.18
DPLM [76]	0.71	0.73	0.64	0.70	0.28	0.28	0.97	0.95	0.94	0.95	0.10	0.08	0.17	0.20
SNN	0.75	0.75	0.74	0.76	0.35	0.35	0.98	0.98	0.96	0.98	0.14	0.11	0.22	0.23
ResNet18-SNN	0.88	0.83	**0.90**	0.89	0.77	0.79	**0.99**	**0.99**	**0.99**	**0.99**	0.18	0.18	**0.26**	**0.25**
DenseNet34-SNN	**0.91**	**0.89**	**0.90**	**0.91**	**0.82**	**0.80**	**0.99**	**0.99**	**0.99**	**0.99**	**0.20**	**0.20**	0.25	**0.25**

TABLE III
COMPARING THE RECTIFIER ACTIVATION USED BY LEAF NEURONS AGAINST THE SIGMOID ACTIVATION, UNDER THE CLUSTER-BASED CONCEPT HIERARCHY.

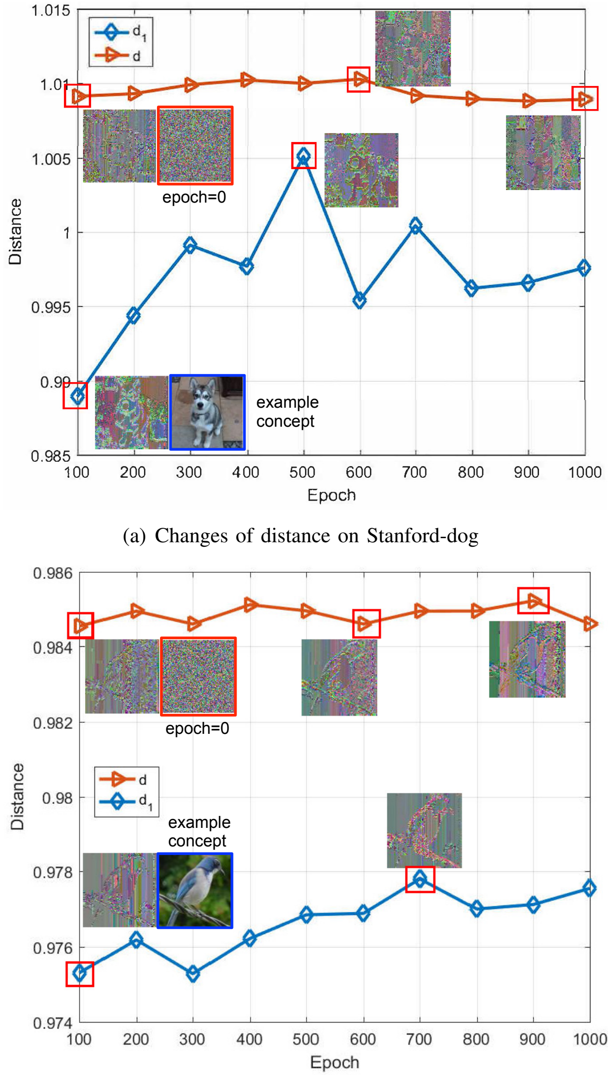
	NUS-WIDE		CIFAR-10		CIFAR-20		MNIST	
	500AP (%)	mAP (%)	500AP (%)	mAP	500AP (%)	mAP (%)	500AP(%)	mAP(%)
Sigmoid ($d_0 = 5$)	0.62	0.63	0.52	0.53	0.24	0.29	0.68	0.70
Sigmoid ($d_0 = 10$)	0.63	0.64	0.51	0.51	0.27	0.30	0.70	0.73
Sigmoid ($d_0 = 15$)	0.65	0.64	0.56	0.56	0.27	0.30	0.70	0.75
Sigmoid ($d_0 = 20$)	0.66	0.66	0.59	0.60	0.28	0.31	0.72	0.79
Rectifier ($d_0 = 5$)	0.67	0.66	0.57	0.62	0.27	0.30	0.92	0.93
Rectifier ($d_0 = 10$)	0.68	0.70	0.62	0.65	0.28	0.31	0.95	0.93
Rectifier ($d_0 = 15$)	0.70	0.70	**0.70**	0.65	**0.29**	0.31	**0.96**	0.94
Rectifier ($d_0 = 20$)	**0.71**	**0.72**	**0.70**	**0.68**	**0.29**	**0.32**	**0.96**	**0.95**

2) Empirical Analysis of SNN: In this section, we conduct a series of experiments to assess the effectiveness of our system design using the NUS-WIDE, CIFAR-10, CIFAR-20 and MNIST datasets. Firstly, Eq. (6) that uses fixed leaf neuron representation vector $\boldsymbol{c}_t$ is compared to the more expressive modification $\mathbf{F}_t$ in Eq. (7) under different settings of $k_F \in \{50, 100, 150, 200, 250\}$. Additionally, we experiment with a random initialization of columns of $\mathbf{F}_t$ under $k_F = 50$. Table II summarizes the performance. It can be seen that Eq. (7) provides better retrieval performance than both Eq. (6) and itself under a random initialization. This indicates the effectiveness of the proposed design. Also, the word-based concepts seem to perform better than the cluster-based concepts. The reason for this can be that, in the case of word-based concepts, the leaf neuron representation matrix $\mathbf{F}_t$ is initialized by the word embedding vectors. These contribute more distinctive features for characterizing the differences between the leaf concepts.

As seen in Table II, the proposed $\boldsymbol{c}_t$-based initialization performs better than a random initialization for optimizing $\mathbf{F}_t$. To investigate further the effect of $\boldsymbol{c}_t$, we observe the changes of the Euclidean distance between the first column of the trained $\mathbf{F}_t$ (denoted by $\boldsymbol{f}_1$) and $\boldsymbol{c}_t$ at different training epochs. We also observe such averaged distance changes between the remaining columns of the trained $\mathbf{F}_t$ (the ith column is denoted by $\boldsymbol{f}_i$) and $\boldsymbol{c}_t$. Particularly, Fig. 5 displays changes of the two quantities $d_1 = \frac{\|\boldsymbol{f}_1 - \boldsymbol{c}_t\|_F}{\|\boldsymbol{c}_t\|_F}$ and $d = \sum_{i=2}^{k_F} \frac{\|\boldsymbol{f}_i - \boldsymbol{c}_t\|_F}{(k_F - 1)\|\boldsymbol{c}_t\|_F}$, using the Stanford-Dog and CIFAR-20 datasets. The experiment is performed under the setting of $k_F = 50$. Additionally, visual patterns captured by reconstructing images using Eq. (12) from the first and the averaged remaining columns of $\mathbf{F}_t$ that corresponds to a randomly selected example concept at different example epochs are illustrated in the figure. It can be seen that the overall change is within a narrow range given both d_1 and d varying around 1 over the training, though there is more significant changes in d_1 than d, also the change pattern is data dependent. The reconstructed images show that the abstract patterns finally learned by $\mathbf{F}_t$ are related to $\boldsymbol{c}_t$.

Next, we investigate the impact of the rectifier activation function used to create sparsity in $\boldsymbol{r}_{ij}^{(0)}$, by comparing it against the sigmoid function. Meanwhile, we investigate the impact of different numbers of leaf concept neurons $l_0 \in \{5, 10, 15, 20\}$ based on the cluster-based concepts. The results are summarized in Table III. It can be seen that the rectifier activation with larger number of hidden neurons offers significantly better performance than the sigmoid function. By setting the threshold for classifying the active and inactive relation neurons as the mean value of the elements in all the computed similarity vectors, the percentage of the active neurons can be computed. We observe that the rectifier activation outputs 35%, 33%, 30% and 30% active relation neurons given a total of $d_0 = 5, 10, 15$ and 20 neurons, while the sigmoid function outputs higher percentage of active neurons, e.g., 45%, 45%, 45% and 44% for the same settings of d_0. This shows that the use of a rectifier activation function to enforce sparsity not only provides higher retrieval performance, but also returns a more highlighted picture of the active relation types. This can potentially lead to improved model interpretability.

We also compare different settings of the network architecture for learning the similarity scores. These include a single-layer network which is equivalent to using only one flat group



(a) Changes of distance on Stanford-dog

(b) Changes of distance on CIFAR-20

Fig. 5. Illustration of the changes of the two quantities $d_1 = \frac{\|\boldsymbol{f}_1 - \boldsymbol{c}_t\|_F}{\|\boldsymbol{c}_t\|_F}$ and $d = \sum_{i=2}^{k_F} \frac{\|\boldsymbol{f}_i - \boldsymbol{c}_t\|_F}{(k_F - 1)\|\boldsymbol{c}_t\|_F}$ using the Stanford-dog and CIFAR-20 datasets. Example images reconstructed by Eq. (12) from the first and the averaged remaining columns of the $\mathbf{F}_t$ corresponding to an example concept are illustrated for different example epochs (highlighted in square boxes).

of concepts, a fully connected network which is equivalent to assuming all the concepts from adjacent levels are connected, and a sparsely connected network with the neuron connections controlled by the concept hierarchies as proposed in this work. We experiment with these three architecture settings by initializing the neurons in the first layer using: (1) the word embedding vectors, and (2) the cluster center vectors. fc is short for fully-connected. hier is the short for hierarchical. The performance is compared in Table IV, from which it can be seen that by enforcing the network architecture to match a pre-identified concept hierarchy, the retrieval performance is improved compared to the single-layer and fully-connected multi-layer architectures in most cases. In addition to offering improved interpretability, the proposed method also has the potential of improving retrieval accuracy.

Finally, to examine how the proposed model performs under an inductive setting, we replace the query images in the MNIST dataset with randomly selected images from the unseen dataset USPS. The comparison is performed between models with different network architecture as used above. The performance is reported in Table IV under the column of MNIST-USPS. It can be seen that the proposed method suffers the least performance drop when changing from the transductive setting (being trained on MNIST and tested on MNIST, as shown in the MNIST column) to the inductive setting (being trained on MNIST and tested on USPS, as shown in the MNIST-USPS column).

3) Comparison with Existing Methods: We compare the proposed SNN guided by the word-based concept hierarchy for NUS-WIDE, CIFAR-10, CIFAR-20 and MNIST/USPS datasets and by the cluster-based concept hierarchy for Caltech-Bird and Stanford-Dog datasets, against ten existing algorithms including online multiple kernel similarity learning (OMKS) [19], the conventional approach of iterative quantization (ITQ) [69], mutliview alignment hashing (MAH) [70], deep regularized similarity comparison hashing (DRSCH) [71], deep semantic ranking hashing (DSRH) [72], kernel based supervised hashing (KSH-CNN) [73], neighborhood discriminant hashing (NDH) [74], multi-modal similarity learning with convolutional features (MVC-MS) [60], cross-modal collaborative quantization (CMCQ) [75] and discrete proximal linearized minimization hashing (DPLM) [76]. Some of these methods focus on multi-view or deep representation learning, some on multimodal similarity learning and some are specialized in image retrieval. For the competing methods, the same parameters as used in the corresponding published papers are adopted. In the digit recognition case, both the transductive setting (being trained on MNIST and tested on MNIST, as shown in the MNIST column) and the inductive setting (being trained on MNIST and tested on USPS, as shown in the MNIST-USPS column) are assessed. The performance is reported in Table V. It can be seen that the proposed method outperforms the existing ones for all the datasets.

We investigate the performance of the proposed similarity learning model when using state-of-the-art deep networks to obtain the image presentations $\{\phi_i\}_{i=1}^n$. The model is connected with ResNet-18 [66] and DenseNet-34 [67], where ResNet-18 includes 18 layers with a configuration of 1 Conv layer, 4x4 Conv layers and 1 fully connected layer, while DenseNet-34 includes 3 transition layers and 4 dense blocks resulting in a total of 34 layers. The performance is reported in Table V, showing very promising performance improvement.

In addition to directly comparing the performance as in Table V, we also examine the statistical significance of the performance difference between the proposed and existing methods on various datasets using the F-distribution as employed in [77]. Originally, F-distribution [78] serves as a non-parametric analysis of variance test for detecting differences in treatments across multiple test attempts. It is computed by

$$F_F = \frac{(N-1)F^2}{N(k-1) - F^2}, \tag{16}$$

with

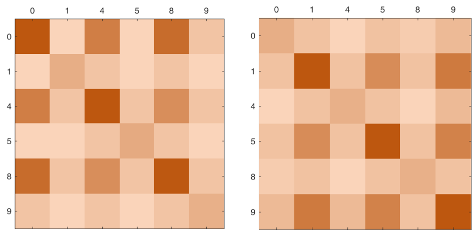
$$F^2 = \frac{12N}{k(k+1)}\left[\sum_{j}^{N} R_j^2 - \frac{k(k+1)^2}{4}\right], \tag{17}$$

where N denotes the number of used datasets, k the number of compared algorithms, and R_j the sum of the rank of the jth algorithm over all the used datasets. The computed F_F is checked against the table of the critical values [79] to determine whether the difference is significant. We apply this method to analyze performance reported in Table V, for which $N = 7$ and $k = 11$. These algorithms are ranked for each dataset separately according to their performance, where the best performing one is ranked as the 1st. The computed F^2 for our experiments is 4.09 and the F_F is 0.37. According to the critical value table, $F^2 = 4.09, F_F = 0.37$ indicates that the proposed method provides relative significant improvement over the 10 competing algorithms assessed by the 7 benchmark datasets.

B. Visualizing and Understanding SNN

1) Cases with Word-based Concept Hierarchy: When the word-based concept hierarchy is used, the semantic meaning of each concept word, as shown in Figs. 3 and 4, naturally interprets the role of its corresponding hidden neuron in a trained SNN. To verify this, we take a close look at the trained SNN for the MNIST data which employs an odd/even number based concept hierarchy as shown in Fig. 3. Randomly choosing 1,000 examples of image pairs corresponding to a digit pair, we compute the relevance scores for each image pair by Eq. (9) for the trained "even" and "odd" neurons. Between randomly chosen MNIST digits, we examine and illustrate in Fig. 6 the averaged relevance scores over their corresponding example image pairs. It can be seen that two digits from the same odd (or even) group possess higher relevance scores on the neuron corresponding to their correct group. For instance, both the pairs ("0", "4") and ("4", "4") possess higher relevance score over the "even" neuron than the "odd" neuron. Among these two pairs, ("4", "4") possesses higher score as they represent the same digit. In general, the relevance scores between the same digits that are computed on the correct odd (or even) group that they belong to are among the highest ones. These correspond to those darkest diagonal blocks in Fig. 6, e.g., ("5", "5") on the "odd" neuron and ("8", "8") on the "even" neuron. This shows that the behaviour of the two neurons are compatible with their semantic interpretation "odd" and "even".

We perform another experiment using the NUS-WIDE dataset, to examine the connection between the word description and the visual pattern of the hidden neuron. The images that maximally activate the neurons corresponding to different concept words are displayed in Fig. 7 for different query images. It can be seen that the query image and its matching image obtained by Eq. (18) over each neuron constitute an image pair that also matches the semantic meaning of the target neuron's descriptive word.



(a) "even" neuron. (b) "odd" neuron.

Fig. 6. Illustration of the averaged relevance scores between MNIST digits, computed on the "even" and "odd" parent neurons, where darker colour indicates higher score.

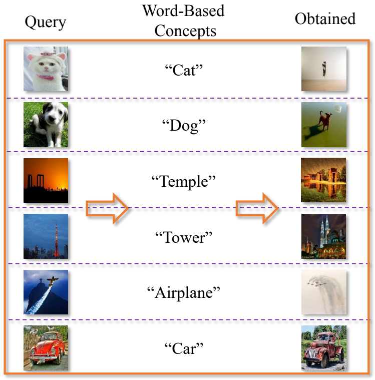


Fig. 7. Illustration of the images obtained by Eq. (18) for neurons corresponding to different word-based concepts and different query images, using the NUS-WIDE dataset.

2) Cases with Cluster-based Concept Hierarchy: When the cluster-based concept hierarchy is used, although there is no direct semantic description available for each hidden neuron, their characteristics can be revealed by their corresponding visual patterns through approaches proposed in Section III-C. Examining a trained SNN for the NUS-WIDE data where l_H=3, Fig. 8 displays the top images that maximally activate each neuron based on Eq. (18). The maximum activation of a parent neuron is determined by its child neurons. With regard to the leaf neurons, when the query image is irrelevant to the corresponding concept, the elements in $\phi_i^T \mathbf{F}_t$ possess low values. The returned images have to offer high values of

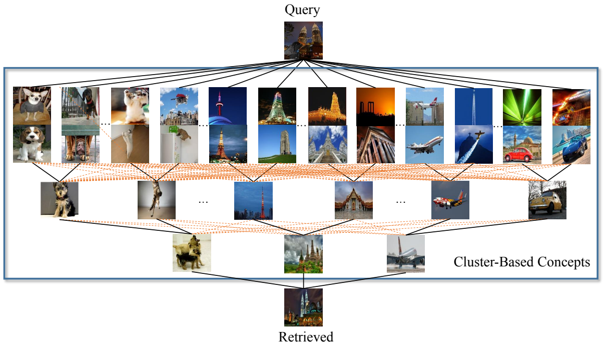


Fig. 8. Illustration of the top two images that maximally activate the leaf neurons and the top images that maximally activate the parent neurons for the NUS-WIDE dataset. Each neuron corresponds to a cluster-based concept. The used query image and its top retrieved image are also displayed.

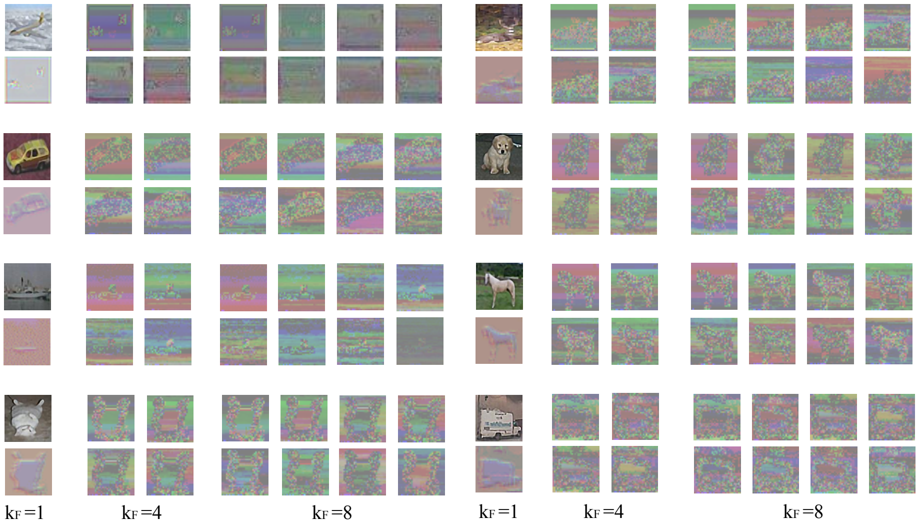
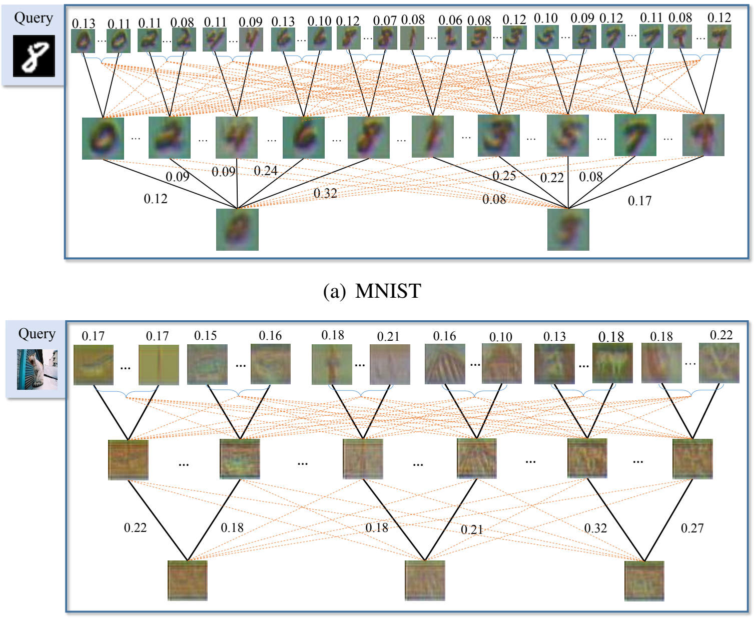


Fig. 9. Illustration of the reconstructed images according to Eq. (12) for eight example leaf neurons using the CIFAR-20 dataset. In the $k_F = 1$ columns, the top image in each example corresponds to the original image that is closet to the cluster center in the CNN feature space, while the bottom image is reconstructed from the optimized $\mathbf{F}_t$. In the $k_F = 4$ (and 8) columns, all the four (and eight) images in each example are reconstructed from the four (and eight) columns of $\mathbf{F}_t$.



(a) MNIST

(b) NUS-WIDE

Fig. 10. Illustration of the reconstructed images for both leaf and parent neurons by Eqs. (14,15), where the cluster-based concept hierarchies are used. The neuron connections are shown in solid lines and the learned connection weights are displayed. The dashed lines indicate the eliminated neuron connections due to the concept hierarchy match.

elements in $\phi_j^T \mathbf{F}_t$, and therefore they are supposed to resemble similar visual patterns to those carried by the leaf neurons. Because of this, Fig. 8 serves as an effective interpretation of the trained network.

Additionally, we illustrate in Fig. 9 the reconstructed images from columns of the representation matrix $\mathbf{F}_t$ of the leaf neurons using Eq. (12). It seems that the reconstructed images capture the main shape patterns possessed by the images that are close to the cluster centers, and can thus be used as an alternative way of understanding what type of visual patterns the leaf neurons capture.

As a matching experiment to that in Fig. 6, we illustrate the visual patterns captured by the SNN for different MNIST digits and the odd/even concepts. This is achieved by employing an image cluster based concept hierarchy with matching parent-level concepts to the word based one in Fig. 3. Specifically, the 1st-layer parent concepts are image clusters corresponding to the different digits, the leaf concepts correspond to sub image clusters of different digits. To enable to observe the visual patterns of the odd/even concepts, two 2nd-layer parent

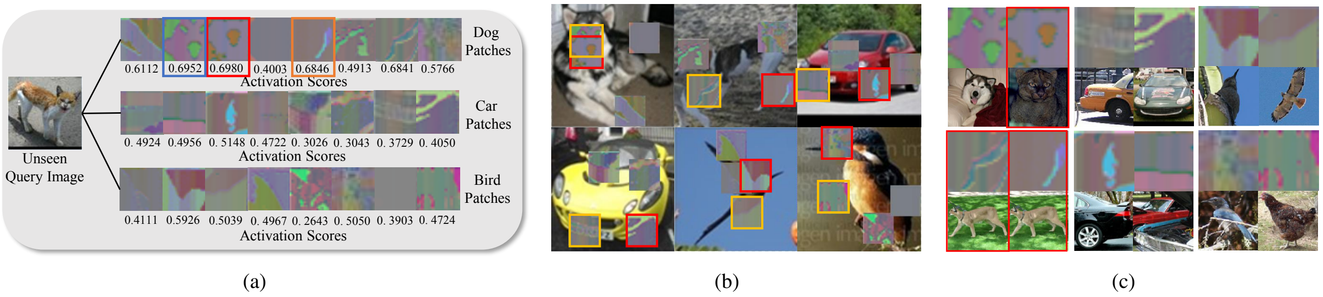


Fig. 11. (a) Reconstructed image and the activation score over each patch-based concept, where the concepts with top three activation scores are highlighted in red box, blue box and orange box, respectively. (b) Reconstructed images of the patch-based concepts and the selected cluster center images, where the concepts with the highest and 2nd highest activation scores are highlighted in red and yellow boxes. (c) Top retrieved images over each concept by Eq. (18). The accurately retrieved images are highlighted by red boxes.

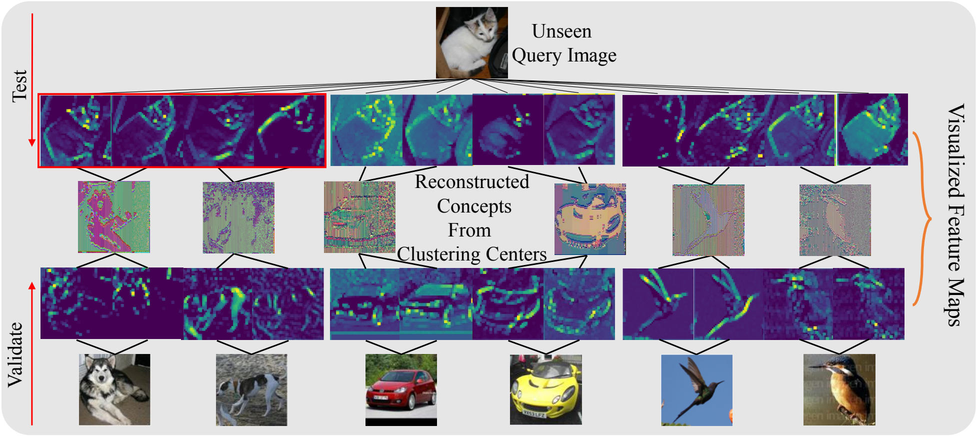


Fig. 12. Illustration of the $m = 2$ feature maps computed over each concept for the query and cluster center images, as well as the reconstructed image for each concept. The red arrow from the bottom to the middle indicates the validation of the pre-trained kernels to demonstrate how these kernels process the samples from the seen concepts. In comparison, the red arrow from the top to the middle indicates how these kernels transfer the pre-trained information to unseen concept and extract feature maps from the unseen query image.

concepts are introduced, each being connected to the image clusters representing the odd (or even) digits. Using Eqs. (14,15), the reconstructed images representing the trained leaf and parent neurons are illustrated in Fig. 10(a). It can be seen that when a neuron is associated with an image cluster corresponding to a single digit, its visual pattern matches that specific digit. However, for the ”odd” and ”even” neurons that are associated with image clusters containing multiple digits, the reconstructed image seems to be a blurred mixture of the visual patterns of the member digits.

The same experiment as above for observing the visual patterns of parent and leaf concepts is also conducted using the NUS-WIDE dataset, and the results are demonstrated in Fig. 10(b). Similarly, it can be seen that the leaf neurons possess more distinguishable visual patterns, whereas the parent neurons correspond to more abstract concepts and their patterns can be viewed as an accumulation of their child neuron patterns which makes them less distinguishable.

3) Cases with Unseen Categories: We conduct two experiments to illustrate how the proposed model interprets its results when dealing with unseen category in a zero-shot retrieval setting. In the first experiment, we employ image patch information as a mean of transferring information to unseen categories, for which the representation vectors of 24 image patches are used to initialize an SNN with one hidden layer. This results in a flat arrangement of 24 concept neurons. Specifically, we extract images from the 3 categories of “dog”, “car” and “bird” in CIFAR-10 dataset, and group these images to 6 clusters using k-means clustering based on their pre-trained image representation vectors. The 6 images that are the closest to the clustering centers are selected. Then, we randomly select 4 patches from each image and resize these patches to the same as that of the original image. The pre-trained image representation vectors of these $6 \times 4 = 24$ patches are used to initialize the concepts $\{\boldsymbol{c}_t\}_{t=1}^{24}$. The model is then trained by following Section III-B.

We test how the trained model performs when being fed images belonging to an unseen category. Fig. 13 shows the top 5 retrieved images for different example query images. We illustrate in detail how the trained model perceives an unseen category by using an example query image from the “cat” category in Fig 11. The activation strength of an input image over different concepts can be computed by a score function like

$$s_t = \ln\left(1 + \exp\left(\boldsymbol{\phi}^T \boldsymbol{c}_t\right)\right), \tag{18}$$

which serves as a part of Eq. (5) and its other enhanced versions. For each patch-based concept, Fig. 11(a) illustrates its

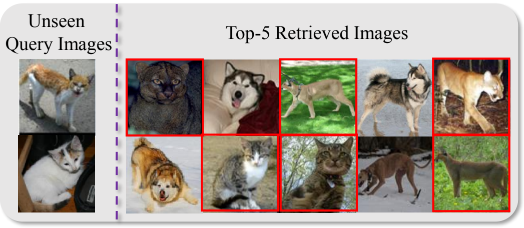


Fig. 13. Illustration of the top-5 retrieved images for different unseen sample query images. The accurately retrieved images are highlighted with red boxes.

corresponding activation score s_t and its reconstructed image by Eq. (12). Fig. 11(b) illustrates these patch-based concepts together with the original images that they are extracted from, where the top two most activated concepts are highlighted. We also illustrate the top retrieved image over each concept in Fig. 11(c). As seen in Figs. 11(a) and 11(c), a cat image maximally activates the third concept neuron from the set of dog patches with a score of 0.6980. The characteristics of the concept neurons and how much they are activated provide a way of interpreting how the proposed model decides how similar an image is to a query image.

In the second experiment, we allow the use of different CNNs to compute different image representation vectors over different concept neurons, and observe the characteristics of each concept through comparing the different feature maps computed over different neurons for the same image. The CNN architecture adopted here is a reduced version as that is used in the previous experiments, where $m = 2$ kernel maps are used in the second convolutional layer instead of $m = 16$. The same 6 images as used in the first experiment are used to initialize 6 concepts, where their representation vector computed by the original CNN is used. The model is trained by using the same training images as in the first experiment by following Section III-B. We show in Fig. 12 the $m = 2$ CNN feature maps computed over each of the 6 concepts, given the query image and the 6 selected cluster center images as the input. In the same figure, we also demonstrate the reconstructed image for each concept by Eq. (12). It can be seen that, the feature maps of the query image computed over the "dog" concepts are of higher quality (highlighted in red box), indicating the "dog" concepts contribute more for processing the unseen "cat" category.

V. Conclusion

We have proposed a novel interpretable deep architecture for modeling similarity distributions between objects. By enforcing the neuron connection to match the organization hierarchy of an a priory defined concept tree, which can either be extracted from existing semantic knowledge or computed by performing clustering analysis over image collections, the network structure becomes naturally interpretable. By designing effective message passing functions and weight initialization strategies, the characteristics of the hidden neurons can be summarized by their corresponding representation matrices or vectors. This facilitates the development of various strategies to observe the neuron characteristics and their associated visual patterns. Evaluated with various datasets and compared against various state-of-the-art algorithms, the proposed method offers the best retrieval performance. In addition, we have provided interesting examples for demonstrating and visualizing what is learned by the SNN neurons.

Acknowledgements

This work was funded by National Natural Science Foundation of China (61806066).

Appendix A
Multi-view Pretraining of CNN

The objective function for optimizing the CNN parameters (stored in $\boldsymbol{\eta}$) in the CNN pre-training is a sum of penalized distance errors between images, given by

$$O_{\text{pre-train}} = \sum_{ij} \text{sig}\left(\sigma_{ij} \left\|\boldsymbol{\phi}(I_i, \boldsymbol{\eta}) - \boldsymbol{\phi}(I_j, \boldsymbol{\eta})\right\|_2^2\right), \tag{19}$$

where $\boldsymbol{\phi}(I_i, \boldsymbol{\eta})$ and $\boldsymbol{\phi}(I_j, \boldsymbol{\eta})$ are the high-level representation of the ith and the jth image. The weight σ_{ij} reflects the similarity and neighbouring information between the ith and jth images, computed from the multi-view features obtained using different feature extraction methods. When there are more views agreeing on the neighbouring relation between two images, a higher weight is awarded to acknowledge that this is a reliable neighbouring image pair, given as

$$\sigma_{ij} = \sum_{\alpha=1}^{m} \frac{\alpha}{m} \sigma_{ij}^{(\alpha)}, \tag{20}$$

and

$$\sigma_{ij}^{(\alpha)} = \begin{cases} 0, & \text{if the } I_{ij}^{(\alpha)} = \emptyset, \\ \frac{1}{\alpha} \sum_{s \in I_{ij}^{\alpha}} P_{ij}^{(s)}, & \text{otherwise,} \end{cases} \tag{21}$$

where $P_{ij}^{(s)}$ is the Euclidean distance between the ith object and the jth object in the sth view, and α represents the number of the views that agree with each other. The set $I_{ij}^{(\alpha)}$ records the indices of the α views agreeing that the ith object and the jth object are neighbours through the distance comparison using the features of these corresponding views. More detailed information on this multi-view based pre-training can be found in [15].

References

[1] D. Chen, Z. Yuan, B. Chen, and N. Zheng, "Similarity learning with spatial constraints for person re-identification," in *IEEE Conference on Computer Vision and Pattern Recognition*, 2016, pp. 1268–1277.

[2] F. Gao, Y. Lou, Y. Bai, S. Wang, T. Huang, and L. Y. Duan, "Improving object detection with region similarity learning," *arXiv preprint arXiv:1703.00234*, 2017.

[3] H. Chen, X. He, L. Qing, and Q. Teng, "Single image super-resolution via adaptive transform-based nonlocal self-similarity modeling and learning-based gradient regularization," *IEEE Trans.on Multimedia*, 2017.

[4] X. Lu, X. Zheng, and X. Li, "Latent semantic minimal hashing for image retrieval," *IEEE Transactions on Image Processing*, vol. 26, no. 1, pp. 355–368, 2017.

[5] W. Ma and B. Manjunath, "Texture features and learning similarity," in *IEEE Conference on Computer Vision and Pattern Recognition*, 1996, pp. 425–430.

[6] E. Xing, M. Jordan, and S. Russell, “Distance metric learning with application to clustering with side-information,” in *Advances in Neural Information Processing Systems*, 2002, pp. 505–512.

[7] D. Grangier and S. Bengio, “A discriminative kernel-based approach to rank images from text queries,” *IEEE Trans.on Pattern Analysis and Machine Intelligence*, vol. 30, no. 8, pp. 1371–1384, 2008.

[8] Y. LeCun, Y. Bengio, and G. Hinton, “Deep learning,” *Nature*, vol. 521, no. 7553, pp. 436–444, 2015.

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

[10] J. Wan, D. Wang, S. Hoi, P. Wu, J. Zhu, Y. Zhan, and J. Li, “Deep learning for content-based image retrieval: A comprehensive study,” in *ACM International Conference on Multimedia*. ACM, 2014, pp. 157–166.

[11] R. G. Cinbis, J. Verbeek, and C. Schmid, “Weakly supervised object localization with multi-fold multiple instance learning,” *IEEE Trans on Pattern Analysis and Machine Intelligence*, vol. 39, no. 1, pp. 189–203, 2017.

[12] F. Radenović, G. Tolias, and O. Chum, “Cnn image retrieval learns from bow: Unsupervised fine-tuning with hard examples,” in *European Conference on Computer Vision*. Springer, 2016, pp. 3–20.

[13] Y. LeCun, L. Bottou, Y. Bengio, and P. Haffner, “Gradient-based learning applied to document recognition,” *Proceedings of the IEEE*, vol. 86, no. 11, pp. 2278–2324, 1998.

[14] P. Merkle, A. Smolic, K. Müller, and T. Wiegand, “Multi-view video plus depth representation and coding,” in *IEEE International Conference on Image Processing*, vol. 1, 2007, pp. I–201.

[15] X. Gao, T. Mu, and M. Wang, “Local voting based multi-view embedding,” *Neurocomputing*, vol. 171, pp. 901–909, 2016.

[16] W. Wang, R. Arora, K. Livescu, and J. Bilmes, “On deep multi-view representation learning,” in *International Conference on Machine Learning*, 2015, pp. 1083–1092.

[17] S. S. Rajagopalan, L. P. Morency, T. Baltrusaitis, and R. Goecke, “Extending long short-term memory for multi-view structured learning,” in *European Conference on Computer Vision*. Springer, 2016, pp. 338–353.

[18] X. Xu, W. Li, D. Xu, and I. W. Tsang, “Co-labeling for multi-view weakly labeled learning,” *IEEE Trans on Pattern Analysis and Machine Intelligence*, vol. 38, no. 6, pp. 1113–1125, 2016.

[19] H. Xia, S. Hoi, R. Jin, and P. Zhao, “Online multiple kernel similarity learning for visual search,” *IEEE Trans.on Pattern Analysis and Machine Intelligence*, vol. 36, no. 3, pp. 536–549, 2014.

[20] Y. Luo, T. Liu, D. Tao, and C. Xu, “Decomposition-based transfer distance metric learning for image classification,” *IEEE Trans.on Image Processing*, vol. 23, no. 9, pp. 3789–3801, 2014.

[21] V. E. Liong, J. Lu, G. Wang, P. Moulin, and J. Zhou, “Deep hashing for compact binary codes learning,” in *IEEE Conference on Computer Vision and Pattern Recognition*, 2015, pp. 2475–2483.

[22] X. Gao, T. Mu, J. Goulermas, and M. Wang, “Attention driven multi-modal similarity learning,” *Information Sciences*, vol. 432, pp. 530–542, 2018.

[23] Z. Lipton, “The mythos of model interpretability,” *International Conference on Machine Learning Workshop on Human Interpretability in Machine Learning*, 2016.

[24] M. Oquab, L. Bottou, I. Laptev, and J. Sivic, “Learning and transferring mid-level image representations using convolutional neural networks,” in *IEEE Conference on Computer Vision and Pattern Recognition*, 2014, pp. 1717–1724.

[25] B. Hu, Z. Lu, H. Li, and Q. Chen, “Convolutional neural network architectures for matching natural language sentences,” in *Advances in Neural Information Processing Systems*, 2014, pp. 2042–2050.

[26] X. Wang and A. Gupta, “Unsupervised learning of visual representations using videos,” in *IEEE International Conference on Computer Vision*, 2015, pp. 2794–2802.

[27] G. Li and Y. Yu, “Visual saliency based on multiscale deep features,” in *IEEE Conference on Computer Vision and Pattern Recognition*, 2015, pp. 5455–5463.

[28] J. Masci, U. Meier, D. Cireşan, and J. Schmidhuber, “Stacked convolutional auto-encoders for hierarchical feature extraction,” in *International Conference on Artificial Neural Networks*. Springer, 2011, pp. 52–59.

[29] X. Li, B. Zhao, and X. Lu, “Mam-rnn: multi-level attention model based rnn for video captioning,” in *International Joint Conference on Artificial Intelligence*. AAAI Press, 2017, pp. 2208–2214.

[30] Y. Li, M. Yang, and Z. Zhang, “Multi-view representation learning: A survey from shallow methods to deep methods,” *arXiv preprint arXiv:1610.01206*, 2016.

[31] J. Masci, M. M. Bronstein, A. M. Bronstein, and J. Schmidhuber, “Multimodal similarity-preserving hashing,” *IEEE Trans on Pattern Analysis and Machine Intelligence*, vol. 36, no. 4, pp. 824–830, 2014.

[32] B. Wang, J. Jiang, W. Wang, Z. Zhou, and Z. Tu, “Unsupervised metric fusion by cross diffusion,” in *IEEE Conference on Computer Vision and Pattern Recognition*. IEEE, 2012, pp. 2997–3004.

[33] J. Lu, G. Wang, W. Deng, P. Moulin, and J. Zhou, “Multi-manifold deep metric learning for image set classification,” in *IEEE Conference on Computer Vision and Pattern Recognition*, 2015, pp. 1137–1145.

[34] S. Rastegar, M. Soleymani, H. R. Rabiee, and S. M. Shojaee, “Mdl-cw: A multimodal deep learning framework with cross weights,” in *IEEE Conference on Computer Vision and Pattern Recognition*, 2016, pp. 2601–2609.

[35] H. Tong, J. He, M. Li, C. Zhang, and W. Ma, “Graph based multi-modality learning,” in *ACM International Conference on Multimedia*. ACM, 2005, pp. 862–871.

[36] M. Seleznova, A. Belyy, and A. Sholokhov, “Towards large-scale exploratory search over heterogeneous source,” *arXiv preprint arXiv:1811.07042*, 2018.

[37] L. Xie, J. Wang, B. Zhang, and Q. Tian, “Fine-grained image search,” *IEEE Transactions on Multimedia*, vol. 17, no. 5, pp. 636–647, 2015.

[38] K. Wang, Q. Yin, W. Wang, S. Wu, and L. Wang, “A comprehensive survey on cross-modal retrieval,” *arXiv preprint arXiv:1607.06215*, 2016.

[39] D. Bau, B. Zhou, A. Khosla, A. Oliva, and A. Torralba, “Network dissection: Quantifying interpretability of deep visual representations,” in *IEEE Conference on Computer Vision and Pattern Recognition*, 2017, pp. 6541–6549.

[40] A. Ghorbani, J. Wexler, and B. Kim, “Automating interpretability: Discovering and testing visual concepts learned by neural networks,” *arXiv preprint arXiv:1902.03129*, 2019.

[41] Y. Dong, H. Su, J. Zhu, and B. Zhang, “Improving interpretability of deep neural networks with semantic information,” in *IEEE Conference on Computer Vision and Pattern Recognition*, 2017, pp. 4306–4314.

[42] D. Mascharka, P. Tran, R. Soklaski, and A. Majumdar, “Transparency by design: Closing the gap between performance and interpretability in visual reasoning,” in *IEEE Conference on Computer Vision and Pattern Recognition*, 2018, pp. 4942–4950.

[43] A. Ross and F. Doshi-Velez, “Improving the adversarial robustness and interpretability of deep neural networks by regularizing their input gradients,” in *AAAI Conference on Artificial Intelligence*, 2018.

[44] D. Melis and T. Jaakkola, “Towards robust interpretability with self-explaining neural networks,” in *Advances in Neural Information Processing Systems*, 2018, pp. 7786–7795.

[45] M. Wu, M. Hughes, S. Parbhoo, M. Zazzi, V. Roth, and F. Doshi-Velez, “Beyond sparsity: Tree regularization of deep models for interpretability,” in *AAAI Conference on Artificial Intelligence*, 2018.

[46] Q. Zhang and S. Zhu, “Visual interpretability for deep learning: a survey,” *Frontiers of Information Technology & Electronic Engineering*, vol. 19, no. 1, pp. 27–39, 2018.

[47] S. Chakraborty, R. Tomsett *et al.*, “Interpretability of deep learning models: a survey of results,” in *IEEE SmartWorld/SCALCOM/UIC/ATC/CBDCom/IOP/SCI*. IEEE, 2017, pp. 1–6.

[48] Y.Lou, R.Caruana, and J.Gehrke, “Intelligible models for classification and regression,” in *ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. ACM, 2012, pp. 150–158.

[49] M. Zeiler and R. Fergus, “Visualizing and understanding convolutional networks,” in *European Conference on Computer Vision*. Springer, 2014, pp. 818–833.

[50] R. Girshick, J. Donahue, T. Darrell, and J. Malik, “Rich feature hierarchies for accurate object detection and semantic segmentation,” in *IEEE Conference on Computer Vision and Pattern Recognition*, 2014, pp. 580–587.

[51] A. Mahendran and A. Vedaldi, “Understanding deep image representations by inverting them,” in *IEEE Conference on Computer Vision and Pattern Recognition*. IEEE, 2015, pp. 5188–5196.

[52] L. van der Maaten and G. Hinton, “Visualizing data using t-sne,” *Journal of Machine Learning Research*, vol. 9, pp. 2579–2605, 2008.

[53] M. Marszalek and C. Schmid, “Semantic hierarchies for visual object recognition,” in *IEEE Conference on Computer Vision and Pattern Recognition*. IEEE, 2007, pp. 1–7.

[54] F. M. Suchanek, G. Kasneci, and G. Weikum, “Yago: A large ontology from wikipedia and wordnet,” *Web Semantics: Science, Services and Agents on the World Wide Web*, vol. 6, no. 3, pp. 203–217, 2008.

[55] R. Fu, J. Guo, B. Qin, W. Che, H. Wang, and T. Liu, “Learning semantic hierarchies via word embeddings,” in *Annual Meeting of the Association for Computational Linguistics*, vol. 1, 2014, pp. 1199–1209.

[56] N. Chen, J. Zhu, F. Xia, and B. Zhang, “Discriminative relational topic models,” *IEEE Trans on Pattern Analysis and Machine Intelligence*, vol. 37, no. 5, pp. 973–986, 2015.

[57] F. Zhou, F. Torre, and J. K. Hodgins, “Hierarchical aligned cluster analysis for temporal clustering of human motion,” *IEEE Trans on Pattern Analysis and Machine Intelligence*, vol. 35, no. 3, pp. 582–596, 2013.

[58] J. Pennington, R. Socher, and C. D. Manning, “Glove: Global vectors for word representation.” in *Conference on Empirical Methods on Natural Language Processing*, vol. 14, 2014, pp. 1532–1543.

[59] J. Weston, F. Ratle, H. Mobahi, and R. Collobert, “Deep learning via semi-supervised embedding,” in *Neural Networks: Tricks of the Trade*. Springer, 2012, pp. 639–655.

[60] X. Gao, T. Mu, J. Y. Goulermas, and M. Wang, “Topic driven multi-modal similarity learning with multi-view voted convolutional features,” *Pattern Recognition*, vol. 75, pp. 223–234, 2018.

[61] T. Chua, J. Tang, R. Hong, and H. Li, “Nus-wide: a real-world web image database from national university of singapore,” in *ACM International Conference on Image and Video Retrieval*. ACM, 2009, p. 48.

[62] A. Krizhevsky and G. Hinton, “Learning multiple layers of features from tiny images,” *M. S. Thesis*, 2009.

[63] J. Hull, “A database for handwritten text recognition research,” *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 16, no. 5, pp. 550–554, 1994.

[64] C. Wah, S. Branson, P. Welinder, P. Perona, and S. Belongie, “The caltech-ucsd birds-200-2011 dataset,” Tech. Rep., 2011.

[65] A. Khosla, N. Jayadevaprakash, B. Yao, and F. Li, “Novel dataset for fine-grained image categorization: Stanford dogs,” in *Proc. CVPR Workshop on Fine-Grained Visual Categorization (FGVC)*, vol. 2, no. 1, 2011.

[66] K. He, X. Zhang, S. Ren, and J. Sun, “Deep residual learning for image recognition,” in *IEEE Conference on Computer Vision and Pattern Recognition*, 2016, pp. 770–778.

[67] G. Huang, Z. Liu, L. D. Maaten, and K. Weinberger, “Densely connected convolutional networks,” in *IEEE Conference on Computer Vision and Pattern Recognition*, 2017, pp. 4700–4708.

[68] W. H. Day and H. Edelsbrunner, “Efficient algorithms for agglomerative hierarchical clustering methods,” *Journal of Classification*, vol. 1, no. 1, pp. 7–24, 1984.

[69] Y. Gong, S. Lazebnik, and A. Gordo, “Iterative quantization: A procrustean approach to learning binary codes for large-scale image retrieval,” *IEEE Trans.on Pattern Analysis and Machine Intelligence*, vol. 35, no. 12, pp. 2916–2929, 2013.

[70] L. Liu, M. Yu, and L. Shao, “Multiview alignment hashing for efficient image search,” *IEEE Trans.on Image Processing*, vol. 24, no. 3, pp. 956–966, 2015.

[71] R. Zhang, L. Lin, R. Zhang, W. Zuo, and L. Zhang, “Bit-scalable deep hashing with regularized similarity learning for image retrieval and person re-identification,” *IEEE Trans.on Image Processing*, vol. 24, no. 12, pp. 4766–4779, 2015.

[72] F. Zhao, Y. Huang, L. Wang, and T. Tan, “Deep semantic ranking based hashing for multi-label image retrieval,” in *IEEE Conference on Computer Vision and Pattern Recognition*, 2015, pp. 1556–1564.

[73] W. Liu, J. Wang, R. Ji, Y. G. Jiang, and S. F. Chang, “Supervised hashing with kernels,” in *IEEE Conference on Computer Vision and Pattern Recognition*. IEEE, 2012, pp. 2074–2081.

[74] J. Tang, Z. Li, M. Wang, and R. Zhao, “Neighborhood discriminant hashing for large-scale image retrieval,” *IEEE Trans.on Image Processing*, 2015.

[75] T. Zhang and J. Wang, “Collaborative quantization for cross-modal similarity search,” in *IEEE Conference on Computer Vision and Pattern Recognition*, 2016, pp. 2036–2045.

[76] F. Shen, X. Zhou, Y. Yang, J. Song, H. Shen, and D. Tao, “A fast optimization method for general binary code learning,” *IEEE Transactions on Image Processing*, vol. 25, no. 12, pp. 5610–5621, 2016.

[77] J. Demšar, “Statistical comparisons of classifiers over multiple data sets,” *Journal of Machine Learning Research*, vol. 7, no. Jan, pp. 1–30, 2006.

[78] M. Friedman, “The use of ranks to avoid the assumption of normality implicit in the analysis of variance,” *Journal of the American Statistical Association*, vol. 32, no. 200, pp. 675–701, 1937.

[79] “F critical values,” http://www.stat.purdue.edu/~jtroisi/STAT350Spring2015/tables/FTable.pdf.

Xinjian Gao received his Ph.D. in signal and information processing from the Hefei University of Technology, China, in 2017. He is currently a Lecturer in the School of Computer Science and Information Engineering, Hefei University of Technology. His research interests include machine learning, multimedia, and pattern recognition.

Tingting Mu received the B.Eng. degree in Electronic Engineering and Information Science from the Special Class for the Gifted Young, University of Science and Technology of China, Hefei, China, in 2004, and the Ph.D. degree in Electrical Engineering and Electronics from the University of Liverpool in 2008. She is currently a Lecturer in the School of Computer Science at the University of Manchester. Her research interests include machine learning and its applications to computer vision, natural language processing and text mining.

John Yannis Goulermas obtained the B.Sc. degree (1st class) in Computation from the University of Manchester (UMIST), in 1994, and the M.Sc. and Ph.D. degrees from the Control Systems Center, UMIST, in 1996 and 2000, respectively. He is currently a Reader in the Department of Computer Science at the University of Liverpool. His research interests include machine learning, combinatorial data analysis, data visualization as well as mathematical modeling. He has worked with various application areas including image/video analysis, biomedical engineering and biomechanics, industrial monitoring and control, and security.

J. Thiyagalingam received his Ph.D. degree in Computer Science from Imperial College, London, in 2005. Currently, he is a Lecturer at the University of Liverpool. Before joining the university, he held positions at Mathworks, U.K. and at the University of Oxford. His research interests include computationally efficient algorithms and models, specifically for learning systems, target tracking, estimation, filtering and data processing. He is a fellow of the British Computer Society and also a member of IET and IEEE.

Meng Wang is a professor at the Hefei University of Technology, China. He received his B.E. degree and Ph.D. degree in the Special Class for the Gifted Young and the Department of Electronic Engineering and Information Science from the University of Science and Technology of China (USTC), Hefei, China, in 2003 and 2008, respectively. His current research interests include multimedia content analysis, computer vision, and pattern recognition. He has authored more than 200 book chapters, journal and conference papers in these areas. He is the recipient of the ACM SIGMM Rising Star Award 2014. He is an associate editor of IEEE Transactions on Knowledge and Data Engineering (IEEE TKDE), IEEE Transactions on Circuits and Systems for Video Technology (IEEE TCSVT), and IEEE Transactions on Neural Networks and Learning Systems (IEEE TNNLS).