Metric Learning as a Service With Covariance Embedding

Imam Mustafa Kamal, Hyerim Bae, *Member, IEEE*, and Ling Liu, *Fellow, IEEE*

***bstract*—Metric learning as a service MLaaS) represents one of the main learning streams to handle complex datasets in service computing research communities and industries. A common approach for dealing with high-dimensional and complex datasets is employing a feature embedding algorithm to compress data through dimension reduction while optimizing intra-class distance. To create generalizable MLaaS for high-performance artificial intelligence applications with high-dimensional Big Data, a robust and meaningful embedding space representation by efficiently optimizing both intra-class and inter-class relationships is required. We developed a novel MLaaS methodology that incorporates covariance to signify the direction of the linear relationship between data points in an embedding space. Our covariance-based feature embedding architecture introduces three different yet complementary mapping functions: inner-class mapping, intra-class with semi-inter-class mapping, and intra- and inter-class mapping. Unlike conventional metric learning, our covariance-embedding-enhanced approach is more expressive and explainable for computing similar or dissimilar measures and can capture positive, negative, or neutral relationships. Our MLaaS framework ensures efficient, composable, and extensible metric learning by supporting the selection of dimension reduction and data compression methods. Experiments conducted using various benchmark datasets demonstrate that the proposed model can obtain higher-quality, more separable, and more expressive embedding representations than existing models.**

***Index Terms*—AI-as-a-service, metric learning, semantic similarity, siamese network, covariance metric.**

I. Introduction

ARTIFICIAL intelligence (AI) has penetrated several business, science, and engineering domains, including self-driving cars, smart manufacturing, smart cities, healthcare diagnostics, information retrieval, recommendation systems, and cloud resource management. This trend continues to grow, fueled by the services computing infrastructure for delivering AI as a service [1] for effective resource and value creation [2] and cognitive intelligence [3], [4]. With the increase in the amount of high-dimensional data produced by a myriad of connected objects and the large amount of data required to train AI models for complex learning tasks, data compression or dimensionality reduction has become a critical data-efficient feature engineering process that involves training a deep neural network algorithm that can accurately preserve the semantic information and the similarity or distance of high-dimensional data in a low-dimensional latent embedding space. The notion of a distance or similarity measure plays a crucial role in deep learning algorithms for feature engineering. Conventionally, standard distance metrics, such as euclidean, cosine, and Mahalanobis [5], are applied by incorporating a *priori* knowledge of the domain. However, it is often challenging to devise metrics pertinent to particular data and tasks of interest.

Metric learning aims to automatically construct task-specific distance or similarity metrics using weakly supervised data in a machine-learning fashion. The outcome, which is low-dimensional data representation, can be used to perform various tasks, such as k-NN, clustering, and information retrieval. Moreover, metric learning can offer a natural solution to various machine learning problems because it has numerous benefits, including robustness to noisy data, a high generalization to unseen categories, the capability of a dimensionality-reduction model, reliability as a feature extraction model (for fine-tuning classification), and the ability to operate with small sample datasets [6]. Accordingly, research into metric learning has received increasing attention over the past few years and has been applied to many different fields, such as information retrieval [7], recommender systems [8], social media mining [9], face recognition [10], and speech recognition [11], to name a few. Siamese [12], Triplet [13], and N-pair [14] networks are prominent metric learning frameworks. A Siamese network is trained using two tandem networks with two different input data to compute their similarity. A triplet network is trained using three networks to define the similarity between three images, two of which will be similar (anchor and positive samples), and the third will be unrelated (a negative example). The N-pair network is trained using the cosine similarity to calculate the pairwise distance of N samples, where $N > 3$. Each of these networks is able to project high-dimensional data into a low-dimensional embedding space while preserving data separability. The samples of different categories must be dissimilar (far from each

Manuscript received 10 February 2022; revised 7 March 2023; accepted 6 April 2023. Date of publication 12 April 2023; date of current version 8 October 2023. The work of Imam Mustafa Kamal and Hyerim Bae was supported in part by the National Research Foundation of Korea (NRF) grant funded by the Korea government(MSIT) under Grant RS-2023-00208999. The work of Ling Liu was supported in part by the US National Science Foundation CISE under Grants 2038029, 2026945, an IBM faculty award, and a Cisco grant on Edge Computing. Recommended for acceptance by E. Damiani. *Corresponding author: Hyerim Bae.)*

Imam Mustafa Kamal is with the Institute of Intelligent Logistics Big Data, Pusan National University, 609-753 Busan, South Korea (e-mail: imamkamal@pusan.ac.kr).

Hyerim Bae is with the Industrial Data Science & Engineering, Department of Industrial Engineering, Pusan National University, 609-753 Busan, South Korea (e-mail: hrbae@pusan.ac.kr).

Ling Liu is with the College of Computing, Georgia Institute of Technology, Atlanta, GA 30332-0765 USA (e-mail: lingliu@cc.gatech.edu).

Digital Object Identifier 10.1109/TSC.2023.3266445

other); whereas the samples from the same categories must have a similar latent representation. Defining the closeness and remoteness among data or samples is a non-trivial task for high dimensional complex data. We argue that the metric learning with similarity measure is critical for obtaining a meaningful embedding space representation.

When similarity measures are not given a *priori*, although a generic function, such as euclidean distance, can be adopted, doing so for high dimensional data may produce unsatisfactory results [5]: If two data vectors have no attribute values in common, they may have a smaller distance than the other pair of data vectors containing the same attribute values. In addition, the magnitude of the vectors is not considered, and only their direction is considered, which is one of the known drawbacks of the cosine similarity [15]. In practice, this means that euclidean distance fails to fully capture the differences in values. To the best of our knowledge, existing metric learning models tend to explicitly minimize intra-class similarity while implicitly neglect the importance of inter-class relationship. As a result, the latent space representation is possibly less meaningful because it cannot capture the inter-class connections. Moreover, existing metric learning models, such as Siamese, Triplet, and N-pair networks often suffer from a low performance under a particular condition. The Siamese network results in fewer semantic outcomes because it neglects the inter-class relationship. The triplet network can diverge when the positive and negative samples have similar features. In addition, the N-pair network requires a large number of batches to obtain an appropriate result that is computationally expensive.

In this paper, we propose a novel metric learning method called CovNet with three original contributions. First, CovNet employs covariance to determine the relationship between two pairs of inputs. The covariance signifies the direction of the linear relationship between the two vectors. By direction, we are referring to whether the variables are directly or inversely proportional to each other. Increasing the value of one variable might have a positive or negative impact on the value of the other variable. In addition, unlike the cosine similarity, the covariance subtracts the means before taking the dot product, making it invariant to shifts. Second, we provide three types of mapping functions to embody the covariance embedding: inner-class mapping (IM), inner-class with semi-inter-class mapping (ISIM), and inner- and inter-class mapping (IIM). Finally, we conducted extensive experiments on seven benchmark datasets and demonstrated that CovNet outperforms existing deep metric learning models, such as Siamese, Triplet, and N-pair networks. The covariance metric is more expressive than the euclidean and the cosine similarity metrics because it can capture three possible relationships between two variables: a positive, neutral, or negative correlation. Compared with existing metric learning models, we show that CovNet is semantically richer and more expressive as it can obtain accurate inner and inter-class relationships. CovNet can be used not only as a framework for delivering metric learning as a service but also as an efficient method for data compression or dimensionality reduction, which are critical for latency reduction and privacy protection in edge-cloud computing [16].

An important challenge in designing our metric learning as a service (MLaaS) architecture is to ensure modularity, composability, and extensibility in dealing with high-dimensional and complex data. Our proposed approach aims to provide metric learning as a service framework, which is efficient and highly composable, by supporting a suite of neural embedding-based dimension reduction and data compression methods using the services computing architecture. By using metric learning as a service, we can reduce the latency of learning and model delivery between the cloud and edge layers to incorporate large and evolving amounts of data. We conjecture that our findings will contribute to advancements in service computing in the context of metric learning.

The remainder of this paper is organized as follows. Section II reviews the state-of-the-art (SOTA) metric learning models, from traditional to the latest approaches, and summarizes recent studies related to AI as a service system. Section III presents the problem formulation, mapping function (IM, ISIM, and IIM), network model, and the learning mechanism of the covariance network (CovNet). The experimental setting, results, and comparison with SOTA, along with a discussion, are presented in Section IV. Finally, we outline some conclusions, implications, and future directions for this research in Section V.

II. Related Works

Metric learning is an approach based directly on a similarity measure that aims to establish a relationship between data points. The relation provides a semantic similarity such that close data points will be considered similar, whereas remote data points will be considered dissimilar. It can also be categorized as a dimensionality reduction because it maps high-dimensional data points to a low-dimensional space while preserving its separable features [17]. Hence, it can also be useful for large-scale data or multimedia applications, which are ubiquitous in the modern era. The popularity of metric learning emerged in 2002 with the pioneering work of Xing et al. [18], who formulated it as a convex optimization problem. Therefore, several metrics learning applications in various domains have been introduced by some scholars, such as medicine, security, social media mining, speech recognition, information retrieval, recommender systems, and computer vision. In terms of business, the usage of metric learning as a service platform, such as recommendation system and image search similarity, makes consumption and buying decisions more effective and the user experience more comfortable by recommending only relevant items. It allows service providers to predict the customer's usage behavior by reusing data mining services.

According to the type of supervision applied during training, the metric learning model mainly falls into two main categories: supervised and weakly supervised learning. In supervised learning, the model is trained using data, where each sample has label information as a standard discriminative model or classifier. In summary, the model learns to map data points based on the label. Thus, the data points in the same label are considered similar and are placed in a close space representation; otherwise, they are mapped in a remote space representation.

Popular supervised metric learning models, such as a linear discriminant analysis (LDA), margin maximizing discriminant analysis (MMDA), learning with side information (LSI), relevant component analysis (RCA), and neighborhood component analysis (NCA). Because supervised metric learning directly employs a label in defining the similarity among the data points, the embedding space becomes rigid to the label, is prone to an overfitting similarly to a standard (deep learning) classifier, has difficulty capturing the semantic relationship across data points lying within different categories, and has problems dealing with new (unlisted) categories. In weakly supervised learning, the model has access to a set of data points with supervision at the tuple level (particularly pairs, triplets, quadruplets, or N pairs of data points). Thus, its mechanism is called weakly supervised because the model does not directly map the similarity of data points based on the corresponding label, i.e., the prominent models of weakly supervised metric learning such as Siamese [12], Triplet [13], and N-pair [14].

The Siamese network algorithm was first introduced by Bromley et al. [19] to two handwritten signatures in 1994. The model was defined as a binary classifier to distinguish whether the pair of data points belonged to the same class or originated in a different class. Owing to its simplicity and applicability, the Siamese network has become the most widely used network by scholars. Zhang et al. [20] developed a content-based image retrieval framework using a Siamese network. In addition, Qiao et al. [21] employed a Siamese network to define user identity linkages through web browsing. Moreover, Yang et al. [22] presented terahertz image verification using a symmetric Siamese network. As represented by a binary classification problem, the Siamese network solely overlooks the inner-class relationship. Thus, the connection between inter-class data points can be less semantic. To address this issue, the triplet network was introduced by Schroff et al. [13] and was originally devised for face recognition. Herein, the data points are mapped in a triplet consisting of an anchor and positive and negative data points; accordingly, the model can learn the similarity of all possible pairs of data points. Some previous triplet network applications are as follows. Zhang et al. [23] adopted a triplet loss for remote-sensing image retrieval. Boutros et al. [24] implemented a triplet loss for mask-image recognition. Lue et al. [25] also devised a visible-thermal-person re-identification framework using triplet loss. Learning the similarity in a triplet fashion is a promising technique for enhancing the semantic similarity of inter-class structures. Nonetheless, this mechanism is challenging to implement. Many scholars show that a triplet loss often suffers from slow convergence, partially because they employ only one negative example while not interacting with the other negative classes in each update [14]. Moreover, if some positive and negative data points are similar, the performance can be degraded. Consequently, Sohn and Kihyuk [14] introduced N-pair networks to solve this problem. In an N-pair network, the model learns the pairwise distance or similarity between all possible pairs in N data points. Commonly, N is represented as the number of batches. Therefore, the network is updated by calculating the pairwise similarity within a batch. Among the latest N-pair network applications, Chen and Deng [26] utilized N-pair loss for image retrieval and clustering. In addition, Pal et al. [27] employed N-pair loss in biomedical image classification. Moreover, Espejo et al. [28] extended the N-pair loss with a $(C_{N,2}+1)$-pair loss function for keyword spotting. Indeed, an N-pair network is a promising metric learning model for obtaining reliable semantic representations. Nevertheless, the N-pair loss tends to have inferior performance when the number of batches (N) is small. Moreover, a large number of batches requires a significantly high computational cost.

In metric learning, the distance or similarity measure also plays a crucial role in defining the embedding space. The base network of a metric learning model commonly produces a vector. Its value must represent the original input both accurately and semantically. The standard distance measure used to define the distance or similarity between vectors is euclidean and cosine. Nevertheless, these often fail to capture the idiosyncrasies of the data of interest [29]. Thus, Norouzi et al. [5] introduced a framework for learning a broad class of binary hash functions based on a triplet ranking loss designed to preserve the relative similarity. Zhang et al. [15] proposed a spherical embedding constraint (SEC) to regularize the distribution of the norms. Moreover, some scholars have recently used distance measures to accomplish a better semantic embedding space representation. Xu et al. [30] used a bi-level distance metric to enhance the similarity accuracy, and Ye et al. [31] incorporated multi-metric learning to capture multi-perspective data relationships. Cheng et al. [32] incorporated metric learning with a CNN model to address the challenges of within-class diversity and between-class similarity in remote-sensing images. Based on the above approaches, we extended this research in another direction. Unlike the previous studies, we devised an expressive but straightforward method for determining the relationship between embedding vectors using covariance metrics.

In recent years, numerous studies have developed AI as a service and design framework. Cao et al. [33] developed a convolutional neural network for food recognition within an edge-computing service infrastructure. Liu et al. [34] introduced reliable service recommendations and demand predictions using a deep neural network. Sami et al. [35] proposed deep reinforcement learning for intelligent fog and service placement (IFSP). For AI-as-a-service, Zhang et al. [3] utilized a Q-learning algorithm to optimize the throughput of AI co-inference in a distributed setting. Wang et al. [36] introduced a services-oriented deep learning architecture using various accelerators such as graphics processing units and field-programmable gate arrays. Moreira et al. [37] developed an AI-as-a-service architecture, particularly a CNN, to deliver AI models with a COVID-19 case study. To the best of our knowledge, our study is the first to build metric learning as a service framework. In addition to system-level composability and extensibility, our covariance-embedding-based approach promotes generalization and explainability, two important functional properties for delivering metric learning as a service.

$$cov(z, z') = \frac{\sum_{i=1}^{s}(z_i - \bar{z})(z'_i - \bar{z}')}{s-1}. \tag{1}$$

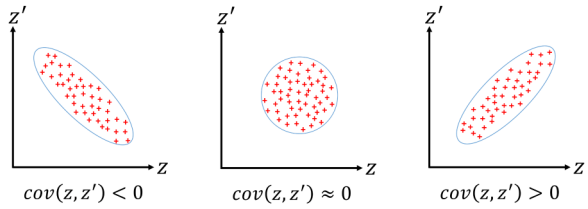

Fig. 1. Covariance between two embedding vectors capturing negative, neutral, and positive correlations.

III. Methodology

Rather than euclidean distance and cosine similarity, the covariance metric is more expressive because it can simultaneously capture three possible relationships between two variables, as depicted in Fig. 1. Given two random variables z and z', the relationship between them becomes negative when $cov(z, z') < 0$, neutral or uncorrelated when $cov(z, z') \approx 0$, and positive when $cov(z, z') > 0$. For instance, in the CIFAR-10 dataset, if z is a latent variable of the bulldog and z' is a puddle, they will have a positive relationship. Thus, if the z and z' are from the same category, they are likely to have a strong positive relationship ($cov(z, z') > 0$). If the z and z' are the latent variables of a ship and a dog, respectively, they can have a neutral or negative relationship. Notably, the neutral or negative relationship can be automatically learned based on the features of the data by the model employing covariance during training. This value can be extracted from the original covariance equation, as denoted in (1), where s represents the total number of variables, $\bar{z}$ is the mean value of z derived from $\bar{z} = \frac{\sum_{i=1}^{s} z_i}{s}$ and $\bar{z}'$ is given as $\bar{z}' = \frac{\sum_{i=1}^{s} z'_i}{s}$. Therefore, unlike euclidean and cosine similarity metrics, the covariance metric is invariant to shifts. Moreover, the covariance metric can capture more information because it can signify an inter-class relationship. For example, an eagle and an airplane can still have a positive relationship; however, their relationship is weaker than that between an eagle and a parrot, similar to that between a dog and cat.

$$\vec{z} = \begin{bmatrix} z_0 \\ z_1 \\ \vdots \\ z_{q-1} \end{bmatrix}, \ \vec{z}' = \begin{bmatrix} z'_0 \\ z'_1 \\ \vdots \\ z'_{q-1} \end{bmatrix} \tag{2}$$

$$\vec{C} = C(\vec{z}, \vec{z}') = \begin{bmatrix} c_0 & c_1 & c_2 & \dots & c_{q-1} \end{bmatrix}^T,$$

$$\text{where } \ c_i = (z_i - \bar{z}) \ \ (z'_i - \bar{z}'). \tag{3}$$

In this study, because we aimed to obtain both low-dimensional and expressive embedding representations, which are represented as vectors (as shown in (2)), the covariance must be calculated at the vector level. If $\vec{z} = (z_0, z_1, z_2, \dots, z_{q-1})$ and $\vec{z}' = (z'_0, z'_1, z'_2, \dots, z'_{q-1})$ is a random vector with $q-1$ dimension, the covariance between them can be formulated as $cov(\vec{z}, \vec{z}') = \frac{\sum_{i=}^{q-1} (z_i - \bar{z})\ (z'_i - \bar{z}')}{q-1}$. Where $\bar{z}$ and $\bar{z}'$ are the mean values of the vectors $\vec{z}$ and $\vec{z}'$, respectively. Notably, $cov(\vec{z}, \vec{z}')$ yields a scalar value because it is summarized and normalized by $q-1$. Herein, we introduce covariance vector $\vec{C}$, as expressed in (3), instead of a scalar because we cannot directly minimize $cov(z, z')$ using an optimization algorithm. Thus, element-wise multiplication between $(z_i - \bar{z})$ and $(z'_i - \bar{z}')$ yields a vector representing a low-dimensional embedding representation. Similar to covariance ((1)), the summation of all elements in $\vec{C}$ will be either positive, zero (uncorrelated), or negative.

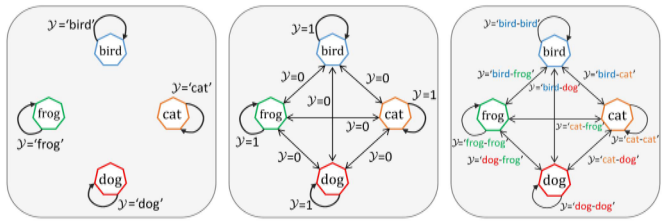

Fig. 2. Mapping functions: IM, ISIM, and IIM. Here, x and x' are samples from four categories {0: bird, 1: cat, 2: dog, 3: frog}. The directed arrows indicate the possible paired combination (labeled as $\mathcal{Y}$) that can be formed between two samples.

A. Problem Definition

Let us denote $X = \{x_0, x_1, \dots, x_{N-1}\}$ as real value N with a p-dimensional space, $X \in \ ^p$, and $Y = \{y_0, y_1, \dots, y_{N-1}\}$ as a label of X. The embedding network (F) will project X into q-dimensional space ($\vec{z} = F(x)$), where $\vec{z} \in \ ^q$ and $q \ll p$. We define a pair input $\{x, x'\}$, where $x = x_i$, $x' = x_j$, and $i \neq j$. $\vec{z}'$ can also be obtained after x' is processed using F. Subsequently, we can obtain the covariance vector, represented as a vector, between $\vec{z}$ and $\vec{z}'$ using (3). By using a mapping function (IM, ISIM, or IIM), we can obtain a label of $\mathcal{X} = \{x, x'\}$, which is represented as $\mathcal{Y}$. Note that Y is a raw label whereas $\mathcal{Y}$ represents the final label generated by the mapping function. The covariance vector $C(\vec{z}, \vec{z}')$ is directly projected to become $\mathcal{Y}$ through the function G ($G(C(\vec{z}, \vec{z}')) = \mathcal{Y}$), where G represents a dense layer with the number of units $|\mathcal{Y}|$. When the label is binary $|\mathcal{Y}| = 1$, and when the label is categorical $|\mathcal{Y}|$ is equal to the total number of categories. Unlike the euclidean and cosine similarity metrics, the covariance operation may result in a range value of $[-\infty, \infty]$; hence we cannot directly minimize it by using an optimization algorithm. Thus, we employ a classification approach to implicitly obtain the covariance value between $\vec{z}$ and $\vec{z}'$.

B. Data Processing and Network Model

Before the network model learns the data, we apply a data processing technique which is defined as a mapping function. The original dataset (X and Y) is mapped into pairwise samples ($\mathcal{X}$ and $\mathcal{Y}$) to estimate their covariance. In this study, we present three types of mapping functions to determine a paired of sample ($\mathcal{X} = \{x, x'\}$) and its label ($\mathcal{Y}$), namely, IM, ISIM, and IIM, as illustrated in Fig. 2. Note that $\mathcal{X}$ is an image and $\mathcal{Y}$ is a text. In the IM, the data are paired if they lie in the same class or category. Thus, this mapping function only learns the inner-class relationship. For instance,

in the CIFAR-10 dataset, if x is a bulldog, x' can be a puddle ($\{x = \text{bulldog}, x = \text{puddle}\}, \mathcal{Y} = '\text{dog}'$) and if x is Persian, x' can be a ragdoll ($\{x = \text{Persian}, x = \text{ragdoll}\}, \mathcal{Y} = '\text{cat}'$), and so on. Thus, suppose K is the total number of categories or classes in a dataset, and the total number of classes in IM (n_{class}) is equal to the total number of categories in a dataset ($n_{class} = 10$ in the CIFAR-10 dataset). The detailed IM mapping algorithm is presented in Algorithm 1. It indicates that the IM explicitly maximizes the inner-class similarity. Nevertheless, it can also implicitly minimize the inter-class similarity after the convergence (when all data points successfully fit with their corresponding categories). In the ISIM, the data are paired in the same as in the original Siamese network mapping. Therefore, there are only two possible labels (binary) for ISIM, namely, a pair containing the same class category (e.g., $\{x$=dog, x'=other dog$\}, \mathcal{Y} = 1$), and a pair consisting of a different class category (e.g., $\{x$=cat, x'=dog$\}, \mathcal{Y} = 0$). The ISIM is blind in defining the inter-class relationship. Because it is assumed that the similarity between a dog and a cat is equal to that between a dog and an airplane, thus, the capturing of the inter-class relationship may be explicitly neglected. The detailed ISIM mapping is presented in Algorithm 2. In the IIM, the data are paired based on all pair combinations among the categories. Hence, this mapping function progressively learns both the inner- and inter-class relationship since it considers all pair combinations of category. For example, ($\{x$=dog, x'=other dog$\}$, $\mathcal{Y} = '$dog-dog$'$), ($\{x$=dog, x'=cat$\}$, $\mathcal{Y} = '$dog-cat$'$), ($\{x$=dog, x'=airplane$\}$, $\mathcal{Y} = '$dog-airplane$'$), ($\{x$=dog, x'=automobile$\}, \mathcal{Y} = '$dog-automobile$'$), ($\{x$=dog, x'=truck$\}$, $\mathcal{Y} = '$dog-truck$'$), ($\{x$=dog, x'=ship$\}$, $\mathcal{Y} = '$dog-ship$'$), ($\{x$=dog, x'=deer$\}$, $\mathcal{Y} = '$dog-deer$'$), ($\{x$=dog, x'=bird$\}$, $\mathcal{Y} = '$dog-bird$'$), $\{x$=dog, x'=frog$\}$, $\mathcal{Y} = '$dog-frog$'$), ($\{x$=dog, x'=horse$\}$, $\mathcal{Y} = '$dog-horse$'$), and so forth for other categories. Note that because the covariance is symmetric, the order of label does not matter, thus the covariance of $\{x = cat, x' = dog\}$ is equal to $\{x = dog, x' = cat\}$. The total number of classes in IIM can be denoted as $n_{class} = K + \binom{K}{2}$. Because all possible combinations in the class categories are explicitly observed, IIM can learn to maximize and minimize inner- and inter-class similarities accurately. The detailed IIM mapping algorithm is presented in Algorithm 3. In addition, compared with IM and ISIM, IIM mapping can be robust in extracting the semantic similarity in both inner-class and inter-class relationships because it learns similarity of all combinations of categories. However, because the number of pair combinations can be significantly increased when the number of categories in a dataset is high, it will become a classification with a large number of categories. In which, this is still an active area of research in machine learning [38]. In addition, unlike IIM, the learning mechanism of ISIM is simpler since it can be represented as a binary classification problem.

The data format obtained from the processing technique is a pair input, $\{x, x'\}$ with a single label, $\mathcal{Y}$. They were trained in a supervised manner using a network model called CovNet. Thus, $\{x, x'\}$ is processed to estimate the label $\hat{\mathcal{Y}}$. The overall architecture of CovNet is illustrated in Fig. 3. We define F as an embedding network represented as a convolutional neural network (CNN). We visually present two embedding networks. However, because they share their weights, they are physically a single network (F). As a standard CNN, the model consists of convolutional, max-pooling, batch normalization, and dropout layers. In the last layer, we employ a global average-pooling layer to capture the extracted feature generated from the

Algorithm 1: IM.

Input: $X = \{x_0, x_1, ..., x_{N-1}\}, Y = \{y_0, y_1, ..., y_{N-1}\}$
Output: $\mathcal{X}$, $\mathcal{Y}$
$CG = \bigcup_k CG_k$, where $CG_k = \{i | y_i = k\}$; // In CG, x is grouped and indexed by y
$\mathcal{X} = []$; // initialize $\mathcal{X}$ as an empty list
$\mathcal{Y} = []$; // initialize $\mathcal{Y}$ as an empty list
for $i \leftarrow 0$ **to** $N - 1$ **do**
  do
    $j \leftarrow$ **random**(CG_{y_i}) ; // get random x_j in CG, where $y_j = y_i$
  while $x_i = x_j$;
  $\mathcal{X}$.**append**$(\{x_i, x_j\})$; // pair x and x'
  $\mathcal{Y}$.**append**(y_i) ; // label, note that $y_i = y_j$
end

Algorithm 2: ISIM.

Input: $X = \{x_0, x_1, ..., x_{N-1}\}, Y = \{y_0, y_1, ..., y_{N-1}\}$
Output: $\mathcal{X}$, $\mathcal{Y}$
$CG = \bigcup_k CG_k$, where $CG_k = \{i | y_i = k\}$; // In CG, x is grouped and indexed by y
$\mathcal{X} = []$; // initialize $\mathcal{X}$ as an empty list
$\mathcal{Y} = []$; // initialize $\mathcal{Y}$ as an empty list
for $i \leftarrow 0$ **to** $N - 1$ **do**
  /* add a matching pair */
  do
    $j \leftarrow$ **random**(CG_{y_i}) ; // get random x_j in CG, where $y_j = y_i$
  while $x_i = x_j$;
  $\mathcal{X}$.**append**$(\{x_i, x_j\})$; // pair x and x'
  $\mathcal{Y}$.**append**(1) ; // a matching label, $\mathcal{Y} = 1$
  /* add a non-matching pair */
  do
    $y_j \leftarrow$ **random_int**$(0, C - 1)$; // to make sure $y_i \neq y_j$
  while $y_j = y_i$;
  $j \leftarrow$ **random**(CG_{y_j}) ; // get random x_j in CG where $y_j \neq y_i$
  $\mathcal{X}$.**append**$(\{x_i, x_j\})$; // pair x and x'
  $\mathcal{Y}$.**append**(0) ; // a non-matching label, $\mathcal{Y} = 0$
end

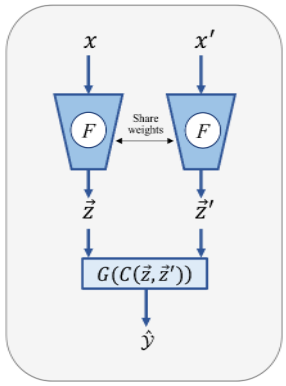

Fig. 3. An example covariance network (CovNet), where the input $\{x\ x'\}$ and output $\mathcal{Y}$ are obtained from the mapping function.

Algorithm 3: IIM.

Input: $X = \{x_0, x_1, ..., x_{N-1}\}$, $Y = \{y_0, y_1, ..., y_{N-1}\}$, K = total number of category
Output: $\mathcal{X}, \mathcal{Y}$

$CG = \bigcup_k CG_k$, where $CG_k = \{i | y_i = k\}$; // In CG, x is grouped and indexed by y
$\mathcal{X} = []$; // initialize $\mathcal{X}$ as an empty list
$\mathcal{Y} = []$; // initialize $\mathcal{Y}$ as an empty list
for $i \leftarrow 0$ **to** $N-1$ **do**
  for $k \leftarrow 0$ **to** $K-1$ **do**
    do
      $j \leftarrow$ **random**(CG_k) ; // get random x_j in CG having category y_k
    while $x_i = x_j$;
    $\mathcal{X}$.**append**$(\{x_i, x_j\})$; // pair x and x'
    $k_1 \leftarrow$ class index of y_i ; // $i \in CG_{k_1}$
    $k_2 \leftarrow$ class index of y_j ; // $j \in CG_{k_2}$
    /* Because the covariance is symmetric, the label of cat-dog is equal to dog-cat */
    if $k_1 \leq k_2$ **then**
      $\mathcal{Y}$.**append**(**str**(label(y_i))+"-"+**str**(label(y_j))) ; // label as a string (E.g. cat-dog, dog-truck, etc.)
    else
      $\mathcal{Y}$.**append**(**str**(label(y_j))+"-"+**str**(label(y_i))) ; // label as a string in ascending order only (E.g. dog-cat becomes cat-dog)
    end
  end
end

convolutional layer. Subsequently, a dense layer with L2 normalization (L2 norm.) is applied to obtain the embedding vector ($\vec{z}$). The L2 norm. has also been commonly used in previous metric learning models because it can result in stability during training. Thus, because $\vec{z}$ and $\vec{z}'$ are already normalized through the L2 norm, if the $\vec{z}$ and $\vec{z}'$ are centered (have zero means), it ((3)) will have the same result as the cosine similarity. In the tail layer, we incorporate it with a dense layer (G) using a softmax (S) (when the mapping function is IM or IIM) or sigmoid function (when the mapping function is ISIM) to determine the class probability.

C. Learning Mechanism

CovNet was trained as a standard neural network with a back-propagation algorithm. The learning mechanism of CovNet is described in Algorithm 4. The inputs are $\mathcal{X}$ and $\mathcal{Y}$, the values of which are obtained from the mapping function used (IM, ISIM, or IIM, as defined in Algorithms 1, 2, and 3, respectively). Note that each of the samples in $\mathcal{X}$ consists of tuples $\{x, x'\}$, and $\mathcal{Y}$ is represented as a one-hot vector (in IM and IIM mapping) and a binary vector (in ISIM mapping). The output of the algorithm is the optimum embedding network model (F^*). Initially, F produces z and z' given input x and x', respectively. Subsequently, we can obtain $C(\vec{z}, \vec{z}')$, which is formulated in (3). Then, G_ψ will classify whether $C(\vec{z}, \vec{z}')$ agrees on the same label. A predefined *loss_function* calculates the discrepancy loss between $\mathcal{Y}$ and $\hat{\mathcal{Y}}$. If we employ IM and IIM as our mapping function, the *loss_function* is the categorical cross-entropy ($\mathcal{L}_{CE}$) formulated in (4) and (5). However, if we utilize ISIM as our mapping function, the *loss_function* is a binary cross-entropy ($\mathcal{L}_{BE}$), denoted in (6). For simplicity, the model parameters

Algorithm 4: Training Procedure of CovNet.

Input: $\mathcal{X} = \{\{x_0, x'_0\}, \{x_1, x'_1\}, ..., \{x_{N-1}, x'_{N-1}\}\}$, $\mathcal{Y} = \{\mathcal{Y}_0, \mathcal{Y}_1, ..., \mathcal{Y}_{N-1}\}$
Output: F^*_ϕ

for $i \leftarrow 1$ **to** $NoEpoch$ **do**
  $\vec{z} = F_{\phi_i}(x)$; // embedding network extracts $\vec{z}$
  $\vec{z}' = F_{\phi_i}(x')$; // embedding network extracts $\vec{z}'$
  $C(\vec{z}, \vec{z}') =$ Eq. 3 ; // extract covariance vector
  $\hat{\mathcal{Y}} = G_{\psi_i}(C(\vec{z}, \vec{z}'))$; // estimate $\mathcal{Y}$
  $\mathcal{L}_{\Omega_i}(\phi_i, \psi_i) =$**loss_function**$(\mathcal{Y}, \hat{\mathcal{Y}})$; // calculate supervised loss betwen $\mathcal{Y}$ and $\hat{\mathcal{Y}}$
  $\Omega_i \leftarrow \Omega_i - \eta \nabla_{\Omega_i} \mathcal{L}_{\Omega_i}$; // update model parameter, $\Omega = \{\phi, \psi\}$
end

$F(\ )$ and $G(\psi)$ are wrapped as a single parameter , and will be updated using a gradient-descent based optimization algorithm. All aforementioned processes were repeated until reaching the predefined number of epochs ($NoEpoch$). Note that, in practice, the model is trained and updated using a predefined number of batches.

$$S(\hat{\mathcal{Y}}_i) = \frac{\exp^{\hat{\mathcal{Y}}_i}}{\sum_{i=1}^{n_{cl}\ ss} \exp^{\hat{\mathcal{Y}}_i}} \tag{4}$$

$$\mathcal{L}_{CE} = -\sum_{i=1}^{n_{cl}\ ss} \mathcal{Y}_i \log(S(\hat{\mathcal{Y}}_i)) \tag{5}$$

$$\mathcal{L}_{BE} = -[\mathcal{Y} \log(\hat{\mathcal{Y}}) + (1 - \mathcal{Y}) \log(1 - \hat{\mathcal{Y}})]. \tag{6}$$

IV. Experiments

This section presents the performance evaluation of our proposed model (CovNets), followed by a comparison and discussion between CovNets and existing metric learning models.

A. Experimental Setting

We evaluated our model on various well-known datasets including those with natural images (CIFAR-10 [39], Food-11 [40], Flower-102 [41], and CUB-200 [42]), biomedical images (Colorectal [43]), and facial images (Adience [44] and Yale [45]). Sample images from each dataset are illustrated in Fig. 4. The CIFAR-10 dataset consists of 50,000 32×32 pixels color training images and 10,000 test images, labeled over ten categories: airplanes, automobiles, birds, cats, deer, dogs, frogs, horses, ships, and trucks. We obtained 10 of the training set for the validation set. Food-11 is a dataset containing 16,643 food images grouped into 11 major food categories: bread, dairy products, desserts, eggs, fried foods, meat, noodles/pasta, rice, seafood, soup, and vegetable/fruit. The original images had various dimensions and were split into 9,866 training, 3,430 validation, and 3,347 testing sets. For simplicity, we uniformly resized the image dimensions to 128×128 pixels. We employ zero-contrast normalization and ZCA whitening as data preprocessing on Food-11 and CIFAR-10. Colorectal cancer is a biomedical dataset containing histological tiles from patients with colorectal cancer. It is made up of 150×150 pixels color images from eight classes, i.e., debris, mucosa, tumor, adipose,

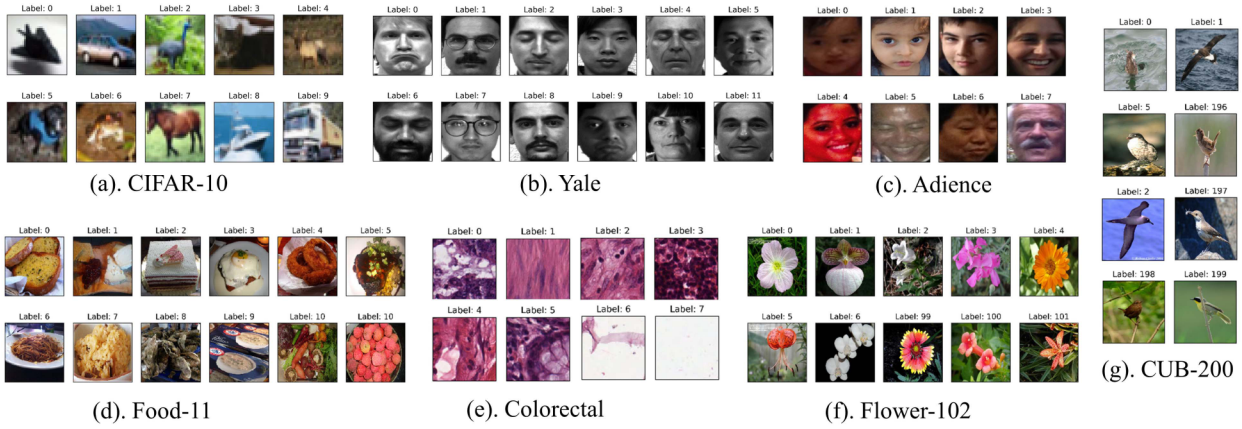

Fig. 4. Dataset overview.

TABLE I
EMBEDDING NETWORK (F) OF COVNET, SIAMESE, TRIPLET, AND N-PAIR NETWORKS IN ALL DATASETS

CIFAR-10: $Conv(64,3,Relu) - BN - Conv(64,3,Relu) - MP(3) - BN - DO(0.5) - Conv(128,3,Relu) - BN - Conv(128,3,Relu) - MP(3) - BN - DO(0.5) - Conv(256,3,Relu) - BN - Conv(256,3,Relu) - Conv(256,3,Relu) - MP(3) - BN - DO(0.5) - AP - Dense(256,Relu) - BN - DO(0.5) - Dense(100,Tanh) - L2Norm.$
Food-11 & Colorectal: $Conv(64,3,Relu) - BN - Conv(64,3,Relu) - MP(3) - BN - DO(0.5) - Conv(128,3,Relu) - BN - Conv(128,3,Relu) - MP(3) - BN - DO(0.5) - Conv(256,3,Relu) - BN - Conv(256,3,Relu) - Conv(512,3,Relu) - MP(3) - BN - DO(0.5) - AP - Dense(256,Relu) - BN - DO(0.5) - Dense(100,Tanh) - L2Norm.$
Adience: $ZP(5) - FaceNet - Dense(512,Relu) - BN - DO(0.25) - Dense(100,Tanh) - L2Norm.$
Yale: $FaceNet - Dense(512,Relu) - BN - DO(0.25) - Dense(100,Tanh) - L2Norm.$
Flower-102/CUB-200: $ResNet50 - AP - Dense(256/512) - L2Norm.$

stroma, lympho, complex, and empty. The original dataset consisted of 5,000 samples. We randomly split the datasets into training, validation, and testing sets at a ratio of 70 , 10 , and 20 , respectively. We reshaped the images into 128×128 pixels, standardized, and normalized to a value range of 0–1. Adience is composed of face images scraped from *Flickr.com* albums that were labeled for age and gender. The benchmark uses eight classes for age groups (0–2, 4–6, 8–13, 15–20, 25–32, 38–43, 48–53, 60+). A total of 38,740 images were split into five groups of 4,484, 3,730, 3,894, 3,446, 3,816, and 19,370. The Yale dataset contains 165 grayscale images of 15 individuals in GIF format. There were 11 images per subject, 1 for each different facial expression or configuration: center-light, w/glasses, happy, left-light, without glasses, normal, right-light, sad, sleepy, surprised, and wink. We preprocessed the Yale and Adience datasets by cropping and centering on the faces and resized them into 150×150 pixels and 160×160 pixels color images for the Adience and Yale datasets, respectively. Flower-102 is a natural image that contains 102 flower categories. The data are initially divided into 1,020, 1,020, and 6,149 samples for training, validation, and testing, respectively. CUB-200, which stands for Caltech-UCSD Birds-200-2011, contains photos of 200 bird species. It has 11,788 images, which have been split into 5,994 and 5,794 images for training and testing, respectively. In the CUB-200, we randomly obtained a 10 validation set from the training set.

We compared our experimental results with the state-of-art metric learning models, such as Siamese, Triplet, and N-pair networks, to assess the effectiveness of our proposed model. We employed the same embedding network (F) in all models for a fair comparison, as shown in Table I. Here, $Conv(i,j,k)$ denotes a convolutional layer with i number of filters, a kernel size of $j \times j$, and k activation functions. In addition, $MP(i)$ is a 2D max-pooling layer with a size of $i \times i$, BN is a batch normalization layer, and $DO(i)$ corresponds to the dropout layer with probability i. In addition, AP is a 2D average-pooling layer, $Dense(i,j)$ represents a dense layer with i neurons and j activation functions. Moreover, $ZP(i)$ indicates zero-padding with size i. For the Adience and Yale datasets, we employed a pre-trained network (FaceNet) [13], which is an inception model trained on the MS-Celeb-1 M dataset, as a backbone network. In the Flower-102 and CUB-200 datasets, we employed ResNet50 (pre-trained network with ImageNet weights) as a base model on top of the embedding network (F). We then added a 2D

TABLE II
COMPARISON OF NETWORK ARCHITECTURES

Model	Input mapping	Embeding net.	Merging layer	Tail layer	Loss function
CovNet v1	IM	$F_{dataset}$	Covariance	Dense(n_{class}, Softmax)	Categorical crossentropy
CovNet v2	ISIM	$F_{dataset}$	Covariance	BN + Dense(1, Sigmoid)	Binary crossentropy
CovNet v3	IIM	$F_{dataset}$	Covariance	Dense(n_{class}, Softmax)	Categorical crossentropy
Siamese	ISIM	$F_{dataset}$	Euclidean dist.	BN + Dense(1, Sigmoid)	Contrastive loss
Triplet	TM*	$F_{dataset}$	Triplet dist.	–	Triplet loss
N-pair	IM	$F_{dataset}$	–	–	N-pair cosine similarity

* Triplet mapping.

global-average-pooling (AP) layer and a dense layer. The dense layer has 256 and 512 units of neurons, which represent the length of the vector z for Flower-102 and CUB-200, respectively, as shown in Table I.

The overall network architecture of all models is summarized in Table II. The value of F can vary based on the dataset, which is referred to in Table I. Here, CovNet v1 and CovNet v3 have different properties in both the input mapping and tail layer. In this case, CovNet v1 utilizes IM, whereas CovNet v3 employs IIM, which is notably more complex than IM. Because n_{class} is defined by all pair combinations of the class label in CovNet v3, the n_{class} of the CovNet v3 value is significantly higher than that of CovNet v1. The n_{class} of CovNet v1 is equal to the total number of class categories. In addition, CovNet v2 has the same architecture as the Siamese network except in the merging layer, which is represented as a binary classification. Both triplet and N-pair networks are more straightforward than the others because they do not have a tail layer. Nevertheless, their learning mechanism can be trickier because triplet and N-pair losses are directly optimized during the training phase. We employed batch sizes of 128, 128, 32, 64, 16, 32, and 32 for the CIFAR-10, Food-11, Colorectal, Adience, Yale, Flower-102, and CUB-200 datasets, respectively. For CovNet v1 in Flower-102, we employed 16 as the batch size. Each network, except the network for Flower-102 and CUB-200, which was trained with 100 epochs, was trained for 200 epochs using Adam optimization. An early stopping mechanism was executed when the best model was obtained based on the validation sets.

$$accuracy = \frac{\sum_{i=1}^{N}\sum_{j=1}^{k} c_i^j}{N \times k} \tag{7}$$

$$c_i^j \begin{cases} = 1, & \hat{\mathcal{Y}}_i^j = \mathcal{Y}_i \\ 0, & \text{otherwise} \end{cases}. \tag{8}$$

B. Near Neighbor Analysis

The outcome of an embedding network is commonly represented as a vector having a significantly lower dimension than the original input. The main objective of the embedding method is to preserve the data separability in the embedding space to be the same as possible as in the original space. Accordingly, any machine learning task with complex and large dimensional datasets, such as images, can be simply solved using the near neighbor algorithm. Eq. (7) measures the accuracy of the near neighbors of each point in the embedding space based on the corresponding label, where k represents the number of neighbors, and N is the total number of data. The c_i^j value is 1 when the data point and the predicted neighbor have the same label; otherwise, it is zero, as shown in (8). Note that the denominator of (7) is $N \times k$, which represents the total number of neighbors of each sample in the dataset. In this study, we employed a simple k-nearest neighbor algorithm with cosine similarity to measure the closeness among the points (represented as vectors) in the embedding space.

The k nearest neighbor performance of the embedding network for all models is shown in Fig. 5. In general, the accuracy decreases slightly with increasing values of k (except for Adience, in which the accuracy is significantly decreased). This indicates that the higher the value of k is, the more diverse the neighbors are. Nevertheless, in real applications, such as search engines and recommendation systems, users are commonly interested only in the top-10 neighbors. CovNet v3 outperforms other methods in all datasets with a small number of categories because it explicitly minimizes the inner-class separability and maximizes the inter-class diversity using IIM. In a small number of categories, a Siamese network is the most competitive existing method for the proposed models. However, it has a slightly lower performance compared to the other approaches in the Adience and Yale datasets, which used a pre-trained network. In comparison, both Siamese and CovNet v2 utilized a binary representation, which implicitly neglected the inter-class relationship. However, CovNet v2 is generally better than a Siamese network. Thus, we can reveal that employing covariance in the merging layer is more robust than the euclidean distance. The triplet network for both Food-11 and Colorectal has worse performance than the other models. This is because some samples in different classes can be nearly similar, as shown in Fig. 4; for example, images labeled 3 and 6 in Food-11 are on the same plate, and

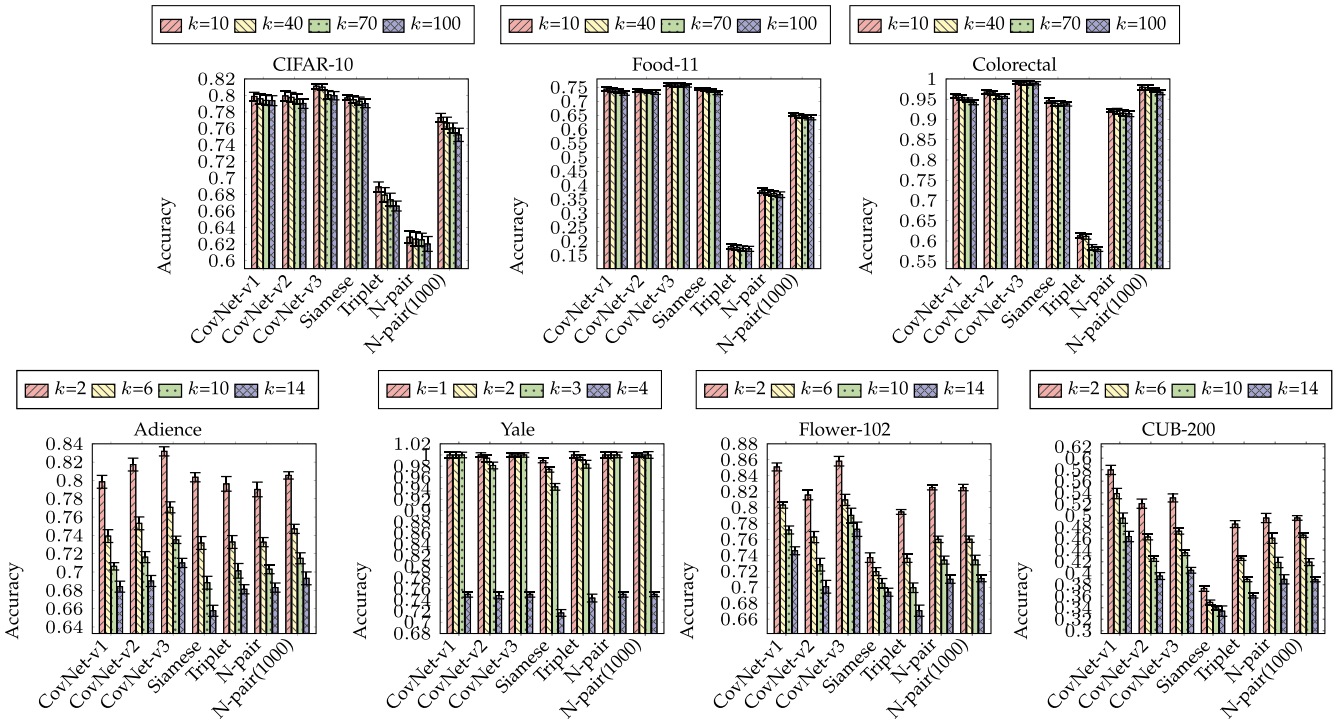

Fig. 5. Effect of varying k (nearest neighbor) on image search accuracy.

those labeled 6 and 7 in colorectal have nearly similar characteristics. Accordingly, the distance between anchor-positive and anchor-negative can be inconsistent. In addition, the N-pair with a small batch achieves a low performance. By contrast, if we increase the number of batches to 1000 (N-pair(1000)), its performance significantly increases. However, the computational cost is considerably increased. In addition, the performance of CovNet v3 was degraded for datasets with a large number of categories (Flower-10 and CUB-200) because the number of paired class comparisons can be significantly increased, making it more challenging for the model to maximize and minimize inter- and intra-class similarity, respectively.

The Triplet and N-pair models show significantly different performances compared to others in the CIFAR-10 and Food-11 datasets. This is mainly because (1) they did not use a pre-trained model, unlike in other datasets (Adience, Yale, Flower-102, and CUB-200); (2) unlike the Colorectal dataset, CIFAR-10 and Food-11 are natural and complex images with various backgrounds. Moreover, converging of the Triplet network with triplet loss, as reported in [46], is challenging when it encounters natural and complex images because it can result in inconsistent distance between "anchor-positive" and "anchor-negative" samples during training. The triplet network uses triplet loss to optimize the metric space embedding representation, which (i) minimizes the distance of the anchor training example to the positive one (the positive sample is randomly selected in a batch from the same class), and (ii) maximizes the distance of the anchor training example to the negative one (the negative sample is randomly selected in a batch from different classes). For example, in CIFAR-10, when the anchor sample is a blue car (class = automobiles), the positive sample is a red truck (class = automobiles) and the negative sample is a blue ship (class = ships). Based on their feature (shape and color) similarity, the triplet loss can be intractable because the distance from the anchor to the positive sample is greater than the distance from the anchor to the negative sample. This particular condition can also occur in other categories in CIFAR-10 (such as "dog-cat" and "horse-deer") and in Food-11 (such as "bread-desserts" and "friedfood-seafood") where the distance from the anchor to the negative sample can be similar with or even shorter than the distance from the anchor to the positive sample. Thus, it can be difficult to minimize and maximize inter- and intra-class similarities, respectively. Meanwhile, as a default setting in CIFAR-10 and Food-11, the number N in the N pair equals the number of batch sizes (128). As is known from the N-pair loss, the more the N, the more sample comparisons are counted to discriminate the inter-class relationship. Unfortunately, in complex and natural images, such as CIFAR-10 and Food-11, N=128 is not optimum enough to minimize and maximize inter- and intra-class similarity, respectively. However, if N is increased to 1,000, as in the N-pair (1,000), it significantly improves the performance, which is not substantially different from that of other models.

In addition, the original Siamese network utilized a binary representation to define the similarity for each sample, where a pair of samples originating from the same class is labeled one; otherwise, it is labeled zero. Hence, it disables the capture of the semantic connections of inter-categories. For example, the distance between a cat and a dog is the same as that between a cat and a truck. Moreover, the original Siamese network employs euclidean distance, which only considers the magnitude to define the closeness between two embedding vectors; thus, it is prone

Fig. 6. Image search application on CIFAR-10: The first column (yellow box) represents the query image, the remaining columns are the 10 nearest images of the query image, and the irrelevant image result is marked by the red box.

to yield unsatisfactory results [5]. In contrast, in the cosine similarity used in the N-pair, the magnitude of the vectors is not considered, and only their direction is considered, which is a known drawback of cosine similarity [15]. Our covariance embedding considers both magnitude and direction in defining the closeness between two embedding vectors in a low-dimensional space. It considers magnitude because each element of the vector is subtracted by its mean value, whereas it considers the direction because it is the same as cosine similarity, which can be represented as a dot product between two normalized vectors. Furthermore, the intra-class similarity is enhanced, or inter-class dissimilarity is improved, in our CovNets because, unlike the existing models, we provide a more comprehensive paired comparison for counting the closeness among embedding vectors, namely, IM, ISIM, and IIM.

As a service, the goal of computing technology is to perform business services more efficiently and effectively. Nowadays, many real-world applications must deal with large-scale datasets, such as image similarity searches and content-based recommendation systems. Thus, metric learning can be efficient because it can project high-dimensional images into a small vector. A vector representation must also reflect the original data effectively. One of the interesting applications of near-neighbor (or similarity) problems is image search similarity. In this case, given a query image, the model retrieves k similar images as the query image. The visualization of the image similarity is illustrated in Figs. 6 and 7. The first column (yellow box) represents the query image, whereas the remaining columns correspond to the top-10 similar images. The red box represents irrelevant images based on the label of the query image. In Fig. 6, the irrelevant images mainly occur on dogs-cats and deer-horses, which share similar features. In Fig. 7, because the label (face age) naturally has an ordinal relationship, the error commonly occurs in the nearest age; for instance, 15–20 a-olds are often predicted as 25–32 a-olds. In general, we can infer that CovNet v3 outperforms the other models. Here, CovNet v1 and v2 slightly outperformed the Siamese network. Compared with the triplet and N-pair networks, the Siamese network is more competitive with our proposed models.

C. Semantic Analysis

In this section, the semantic relationships between both inter-class separability and inter-class relationships are observed. We present Fig. 8 as a visualization of the embedding space to assess inter-class separability. The output of the embedding network was a vector with 100 dimensions. Accordingly, we utilized t-stochastic embedding (t-SNE) to project them into a two-dimensional space. As illustrated in Fig. 8, our proposed methods have a better embedding space separability compared with Siamese, Triplet, and N-pair networks. Therefore, we can reveal that covariance embedding generates separable outcomes more effectively than the euclidean and cosine similarity metrics. Note that, unlike the euclidean and cosine similarity metrics, covariance embedding is more expressive in capturing the relationship between two inputs because it can reflect the positive, negative, or even neutral correlation simultaneously. In addition, for the merging layer (see Table II), CovNet obtains a vector that is directly assessed using categorical or binary cross-entropy. By contrast, other models result in a scalar in the merging layer. Thus, this mechanism can be more effective in preserving the data separability in the embedding space because assessing a

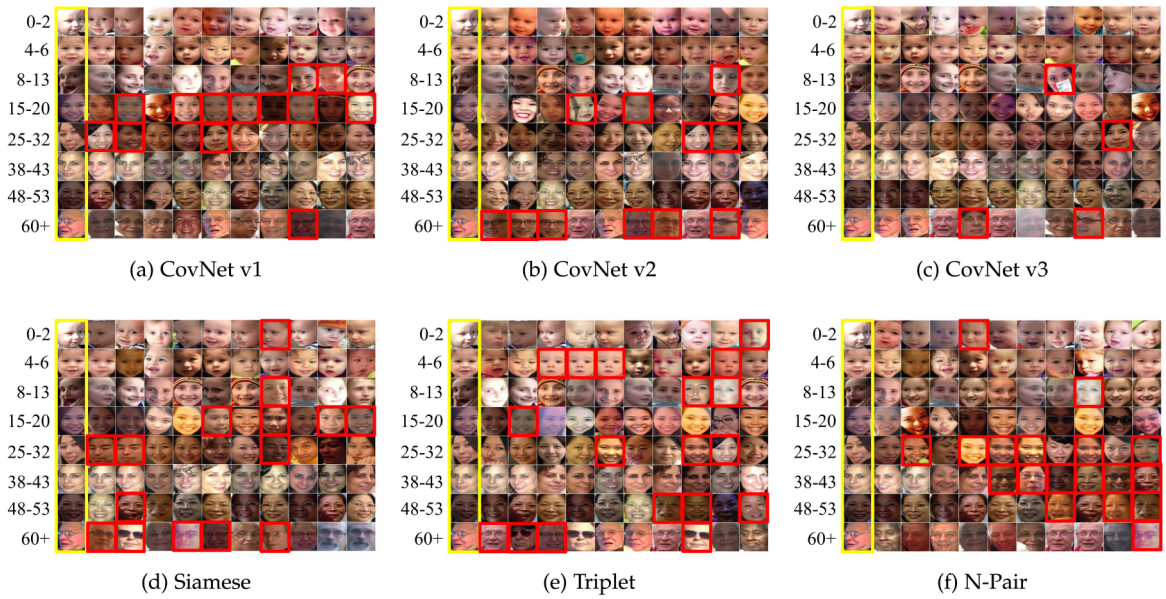

Fig. 7. Image search application on Adience: The first column (yellow box) represents the query image, the remaining columns are the 10 nearest images of the query image, and the irrelevant image result is marked by the red box.

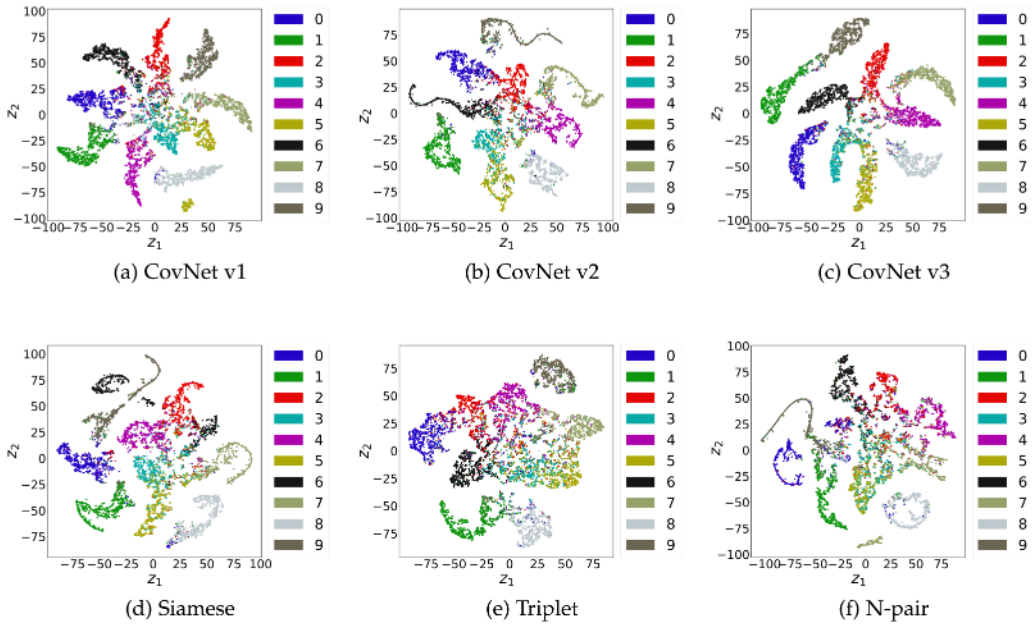

Fig. 8. Visualization of latent separability on CIFAR-10 by projecting an embedding space to a two-dimensional space using t-SNE.

vector (multivariate) as a feature is more effective for guaranteeing separability than representing it as a scalar (univariate).

In addition, we argue that a reliable embedding-space representation can be assessed not only through inter-class separability but also by inter-class relationships. For instance, semantically, we agree that dog-cat and truck-automobile share similar shapes and properties. Therefore, a good embedding-space representation must also preserve this special proximity property. This property is useful for content-based recommendation systems and search engine applications. For better services, in addition to recommending similar products, it is better if the system also offers different products that have similar characteristics as the user query. Consequently, we employed the Pearson correlation method to assess the relationships among the inter-class images. The results are summarized in a correlation matrix, as shown in Fig. 9. Compared with other models, our CovNet is more expressive in describing an inter-class relationship. Unlike the existing models, the inter-class correlation value is still high, whereas the value is still lower than the inner-class correlation value. Inherently, a well-known similar object in CIFAR-10 is in a different class, such as cat-dog, airplane-bird, deer-horse, and automobile-truck (red box in Fig. 9). Nevertheless, unlike

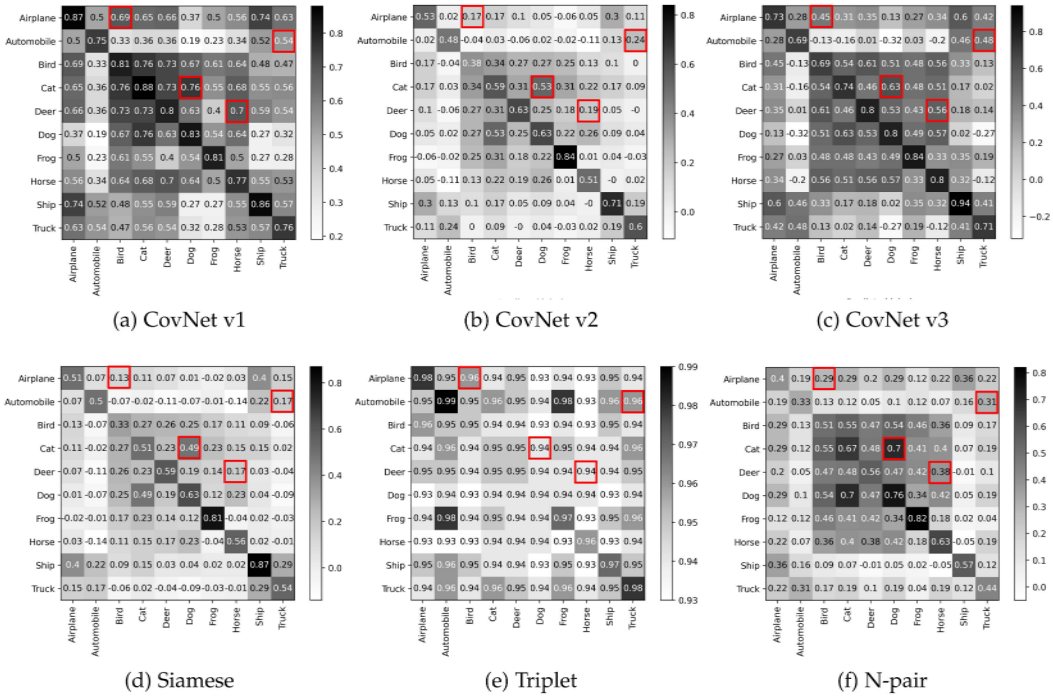

Fig. 9. Pearson correlation matrix to capture the semantic relationship among categories on CIFAR-10.

TABLE III
INSPECTION OF EMBEDDING SPACE SEPARABILITY USING A STANDARD CLASSIFIER (SUPPORT VECTOR MACHINE)

Model	Accuracy % (mean ± standard deviation)						
	CIFAR-10	Food-11	Colorectal	Adience	Yale	Flower-102	CUB-200
CovNet v1	85.4 ± 0.48	81.2 ± 0.72	84.7 ± 0.53	65.3 ± 0.49	100.0 ± 0	82.79 ± 0.52	68.20 ± 0.54
CovNet v2	85.2 ± 0.41	79.9 ± 0.70	83.6 ± 0.67	64.4 ± 0.43	100.0 ± 0	78.41 ± 0.38	65.53 ± 0.37
CovNet v3	86.1 ± 0.39	82.3 ± 0.65	86.5 ± 0.46	66.0 ± 0.37	100.0 ± 0	84.06 ± 0.62	65.61 ± 0.70
Siamese	85.2 ± 0.42	79.0 ± 0.69	82.1 ± 0.50	64.4 ± 0.47	100.0 ± 0	74.57 ± 0.66	51.67 ± 0.43
Triplet	74.1 ± 0.61	36.6 ± 0.94	77.0 ± 0.73	65.0 ± 0.68	100.0 ± 0	75.04 ± 0.46	62.16 ± 0.40
N-pair	71.7 ± 0.58	48.9 ± 0.74	48.5 ± 0.65	62.1 ± 0.65	100.0 ± 0	80.07 ± 0.59	65.12 ± 0.51
N-pair(1000)	83.5 ± 0.35	75.1 ± 0.67	67.2 ± 0.59	64.6 ± 0.37	100.0 ± 0	80.07 ± 0.51	65.14 ± 0.72

our CovNet (particularly v1 and v3), they have low correlation values in Siamese, Triplet, and N-pair networks. The correlation matrix developed using Siamese is slightly similar to that of CovNet v2 because they have a similar architecture. However, semantically, CovNet v2 has a slightly higher correlation in terms of inter-class relationships than Siamese. In the triplet network, the inter-class correlation value is too aggressive, and it has a higher magnitude than its inner-class correlation. Moreover, the N-pair network is too fierce in defining inter-class relationships, and thus some inner-class correlation values also become low.

D. Classification and Parameter

Because of their effectiveness, several metric learning models, such as Siamese, Triplet, and N-pair networks, are commonly used for fine-tuning classification. In this study, a metric learning model acts as a feature extraction network (base model). Here, we train only a few layers on top of it. Meanwhile, the weights of the pre-trained network were not updated during the training. Commonly, we use a deep neural network (DNN) as the top model to classify the data. However, because of its stochastic nature, and for the sake of a fair comparison, we employ a standard classifier, i.e., an SVM with a radial basis function (RBF) kernel, to classify the embedding outcome. The classification performances are presented in Table III. In line with the previous analysis results in the previous sections, CovNet v3 has the best performance compared with other metric learning models. Siamese and CovNet v2 have a competitive performance because their architectures are quite similar. The triplet network is slightly better than N-Pair, except in Food-11. In addition, increasing the number of batches in the N-pair network can increase the classification performance, except in the Yale dataset, which is a relatively small dataset (165 instances). Thus, it was unaffected by the number of batches. Moreover, all models become competitive with each other if we utilize a pre-trained network in the dataset with a small number of categories, as shown in the Adience and Yale datasets.

TABLE IV
COMPARISON OF THE NUMBER OF TRAINED PARAMETERS IN EACH MODEL

Model	Number of trainable parameters						
	CIFAR-10	Food-11	Colorectal	Adience*	Yale*	Flower-102†	CUB-200†
CovNet v1	1,830,294	2,486,523	2,486,523	119,180	119,887	550,758	1,151,688
CovNet v2	1,829,585	2,485,713	2,485,713	118,673	118,673	525,313	1,050,625
CovNet v3	1,834,839	2,492,078	2,489,078	122,008	130,492	1,874,565	2,451,688
Siamese	1,829,585	2,485,713	2,485,713	118,673	118,673	524,548	1,049,092
Triplet	1,829,284	2,485,412	2,485,412	970,340	128,472	524,544	1,049,088
N-pair	1,829,284	2,485,412	2,485,412	970,340	128,472	524,544	1,049,088

* Employing FaceNet [13] (pre-trained network) as a base model.
† Employing ResNet50 (pre-trained network) as a base model.

TABLE V
COMPUTATIONAL TIME (SECONDS) COMPARISON OF EACH MODEL IN THE CIFAR-10 DATASET

Phase	Model						
	CovNet v1	CovNet v2	CovNet v3	Siamese	Triplet	N-pair	N-pair(1000)
Mapping functing	0.388 ± 0.010	0.939 ± 0.007	5.470 ± 0.040	0.934 ± 0.007	1.422 ± 0.015	0.408 ± 0.012	0.411 ± 0.011
Network training (per-epoch)	16.535 ± 0.193	31.811 ± 0.423	102.989 ± 1.784	31.697 ± 0.423	17.248 ± 0.134	18.797 ± 0.182	127.514 ± 1.966
Average of training time	16.923	32.750	108.459	32.632	18.669	19.205	127.925
Inference time	2.203 ± 0.002	2.199 ± 0.002	2.204 ± 0.002	2.201 ± 0.002	2.207 ± 0.002	2.204 ± 0.002	2.206 ± 0.002

From Fig. 5 and Table III, we can see that even under a large number of categories in Flower-102 and CUB-200, all three versions of our CovNet outperformed the existing metric learning models, such as Siamese, Triplet, and N-pair networks. One of the main reasons for this is the intelligent use of covariance embedding and comprehensive mapping functions (IM, ISIM, and IIM) in CovNets for more expressiveness and higher efficiency in classifying the latent variable (z) in low-dimensional space.

In comparison, unlike the datasets with a small number of categories, such as CIFAR-10, Food-11, Colorectal, Adience, and Yale, CovNet v1 tends to perform better than CovNet v2 and CovNet v3. One reason for this is that the inter-class relationship in CovNet v2 is only captured using a binary setting. Hence, a larger number of categories will make the optimization of maximizing inter-class similarity more challenging. Meanwhile, the number of pairing combinations among all classes increased significantly for the datasets with a large number of categories. Hence, CovNet v3 exhibited a lower accuracy performance. The technical challenge for metric learning is, to some extent, similar to deep learning for classification tasks, which tends to degrade performance when a large number of categories are involved in classification learning. The Siamese network uses the euclidean distance, which only considers the magnitude to calculate the distance between the latent variables (z), where the latent variable size (z length) is 256 and 512; thus, it has an inferior performance than those of others. Moreover, there is a positive relationship between the number of pair comparisons (N) and the accuracy when we increase N, the number of categories, i.e., when we perform triplet (three-pair comparison), N-pair (32 pair comparison), and N-pair (1,000 comparisons). However, the difference in accuracy between the N-pair and N-pair (1,000) is relatively small, indicating that it can be saturated if we add more N because it has reached its optimum value.

In addition, we present a comparison of the number of trainable parameters in each model, as described in Table IV. In the dataset with a small number of categories, the number of parameters in each model is not significantly different. CovNet v3 has a higher trained parameter because the tail layer has more neuron units compared with the others, as described in Section IV-A. CovNet v2 and Siamese have the same number of parameters because they have the same architecture except for the merging layer. The merging layer in Table II has no parameters. The triplet and N-pair networks have the least number of parameters because the triplet loss and cosine similarity loss can be directly optimized such that a tail layer is not required. Nevertheless, their performance is significantly lower than that of the other models. The numbers of parameters in the Adience, Yale, Flower-102, and CUB-200 datasets are less than the others because they employ a pre-trained network as a base model.

Finally, we compared the computational times for each model, as listed in Table V. The training phase consists of the processing time of the mapping function and network training. N-pair (1,000) has the longest total average training time because it counts the cosine similarity for all possible pair combinations of 1,000 samples in each batch. CovNet v2 has a similar total average training time as that of Siamese because it has the same mapping function and a similar number of network parameters. Note that Triplet considers counting the similarity among three samples, while N-pair considers N (128 samples as default in our study) samples for each pair. Therefore, our CovNet v1 has a faster total average training time than the others because it only compares two samples in a pair. The inference time refers to the processing time required to generate an embedding vector (z) by the embedding network (F) in the testing set. As shown in the last row of Table V, all the models have relatively the same inference time.

V. Conclusion

Metric learning as a service has emerged as one of the main streams in the services computing research community and industry. We present CovNet, a novel metric learning method, as a service framework. By incorporating covariance to signify the direction of the linear relationship between data points in an embedding space, our covariance-based feature embedding architecture leverages three expressive and explainable mapping functions (inner-class mapping, intra-class with semi-inter-class mapping, and intra- and inter-class mapping) to learn and estimate the covariance of data. Our covariance-embedding-enhanced approach enables metric learning to capture positive, negative, or neutral relationships and to compute similar or dissimilar measures with greater expressiveness and better interpretability. Through the development of CovNets, we show two important properties: (1) A desirable metric learning model should not only separate the dataset based on its categories but also maintain the inter-class semantic relationship. (2) Covariance-enhanced embedding can make complex machine-learning tasks more expressive, more explainable, and more efficient. For example, by producing covariance embedding, face verification simply involves thresholding the distance between two embeddings. Similarly, object recognition, recommender systems, and image search similarity when producing covariance embedding with CovNets can be reduced to a k-NN classification problem. High-dimensional data clustering with CovNets feature-engineering through covariance embedding can be reduced to a linear-space problem solvable using simple bottom-up techniques, including agglomerative clustering. Extensive empirical experiments on seven benchmark datasets demonstrate the effectiveness of CovNets over representative SOTA metric learning models, such as Siamese, Triplet, and N-pair networks.

References

[1] S. N. Yoon and D. Lee, "Artificial intelligence and robots in healthcare: What are the success factors for technology-based service encounters?," *Int. J. Healthcare Manage.*, vol. 12, no. 3, pp. 218–225, 2019.

[2] S. Makridakis, "The forthcoming artificial intelligence (AI) revolution: Its impact on society and firms," *Futures*, vol. 90, pp. 46–60, 2017.

[3] C. Zhang, M. Dong, and K. Ota, "Employ AI to improve AI services: Q-learning based holistic traffic control for distributed co-inference in deep learning," *IEEE Trans. Serv. Comput.*, vol. 15, no. 2, pp. 627–639, Mar./Apr. 2022.

[4] M.-H. Huang and R. T. Rust, "Engaged to a robot? The role of AI in service," *J. Service Res.*, vol. 24, no. 1, pp. 30–41, 2021.

[5] M. Norouzi, D. J. Fleet, and R. R. Salakhutdinov, "Hamming distance metric learning," in *Proc. Adv. Neural Inf. Process. Syst.*, F. C. J. Pereira, C. Burges, L. Bottou, and K. Q. Weinberger, Eds., Curran Associates, Inc., 2012, pp. 1061–1069.

[6] P. Moutafis, M. Leng, and I. A. Kakadiaris, "An overview and empirical comparison of distance metric learning methods," *IEEE Trans. Cybern.*, vol. 47, no. 3, pp. 612–625, Mar. 2017.

[7] Q. Liu, W. Li, Z. Chen, and B. Hua, "Deep metric learning for image retrieval in smart city development," *Sustain. Cities Soc.*, vol. 73, 2021, Art. no. 103067.

[8] H. Wu, Q. Zhou, R. Nie, and J. Cao, "Effective metric learning with co-occurrence embedding for collaborative recommendations," *Neural Netw.*, vol. 124, pp. 308–318, 2020.

[9] Y. Liu, D. Pi, and L. Cui, "Metric learning combining with boosting for user distance measure in multiple social networks," *IEEE Access*, vol. 5, pp. 19342–19351, 2017.

[10] J. Lu, G. Wang, W. Deng, and K. Jia, "Reconstruction-based metric learning for unconstrained face verification," *IEEE Trans. Inf. Forensics Secur.*, vol. 10, no. 1, pp. 79–89, Jan. 2015.

[11] P.-J. Last, H. A. Engelbrecht, and H. Kamper, "Unsupervised feature learning for speech using correspondence and siamese networks," *IEEE Signal Process. Lett.*, vol. 27, pp. 421–425, Feb. 2020.

[12] D. Chicco, *Siamese Neural Networks: An Overview*. New York, NY, USA: Springer, 2021, pp. 73–94.

[13] F. Schroff, D. Kalenichenko, and J. Philbin, "FaceNet: A unified embedding for face recognition and clustering," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2015, pp. 815–823.

[14] K. Sohn, "Improved deep metric learning with multi-class n-pair loss objective," in *Proc. Adv. Neural Inf. Process. Syst.*, D. Lee, M. Sugiyama, U. Luxburg, I. Guyon, and R. Garnett, Eds., Curran Associates, Inc., 2016, pp. 1857–1865.

[15] D. Zhang, Y. Li, and Z. Zhang, "Deep metric learning with spherical embedding," in *Proc. Adv. Neural Inf. Process. Syst.*, H. Larochelle, M. Ranzato, R. Hadsell, M. F. Balcan, and H. Lin, Eds., Curran Associates, Inc., 2020, pp. 18772–18783.

[16] X. Luo, J. Liu, D. Zhang, and X. Chang, "A large-scale web QoS prediction scheme for the industrial Internet of Things based on a kernel machine learning algorithm," *Comput. Netw.*, vol. 101, pp. 81–89, 2016.

[17] I. M. Kamal and H. Bae, "Super-encoder with cooperative autoencoder networks," *Pattern Recognit.*, vol. 126, 2022, Art. no. 108562.

[18] E. Xing, M. Jordan, S. J. Russell, and A. Ng, "Distance metric learning with application to clustering with side-information," in *Proc. Adv. Neural Inf. Process. Syst.*, S. Becker, S. Thrun, and K. Obermayer, Eds., MIT Press, 2003, pp. 521–528.

[19] J. Bromley, I. Guyon, Y. LeCun, E. Säckinger, and R. Shah, "Signature verification using a "Siamese" time delay neural network," in *Proc. Adv. Neural Inf. Process. Syst.*, J. Cowan, G. Tesauro, and J. Alspector, Eds., Morgan-Kaufmann, 1993, pp. 737–744.

[20] K. Zhang et al., "Content-based image retrieval with a convolutional siamese neural network: Distinguishing lung cancer and tuberculosis in CT images," *Comput. Biol. Med.*, vol. 140, 2022, Art. no. 105096.

[21] Y. Qiao, Y. Wu, F. Duo, W. Lin, and J. Yang, "Siamese neural networks for user identity linkage through web browsing," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 31, no. 8, pp. 2741–2751, Aug. 2020.

[22] X. Yang, H. Guo, N. Wang, B. Song, and X. Gao, "A novel symmetry driven siamese network for THz concealed object verification," *IEEE Trans. Image Process.*, vol. 29, pp. 5447–5456, Apr. 2020.

[23] M. Zhang, Q. Cheng, F. Luo, and L. Ye, "A triplet nonlocal neural network with dual-anchor triplet loss for high-resolution remote sensing image retrieval," *IEEE J. Sel. Topics Appl. Earth Observ. Remote Sens.*, vol. 14, pp. 2711–2723, Feb. 2021.

[24] F. Boutros, N. Damer, F. Kirchbuchner, and A. Kuijper, "Self-restrained triplet loss for accurate masked face recognition," *Pattern Recognit.*, vol. 124, 2022, Art. no. 108473.

[25] H. Liu, Y. Chai, X. Tan, D. Li, and X. Zhou, "Strong but simple baseline with dual-granularity triplet loss for visible-thermal person re-identification," *IEEE Signal Process. Lett.*, vol. 28, pp. 653–657, Mar. 2021.

[26] B. Chen and W. Deng, "Deep embedding learning with adaptive large margin n-pair loss for image retrieval and clustering," *Pattern Recognit.*, vol. 93, pp. 353–364, 2019.

[27] A. Pal et al., "Deep metric learning for cervical image classification," *IEEE Access*, vol. 9, pp. 53266–53275, 2021.

[28] I. López-Espejo, Z.-H. Tan, and J. Jensen, "A novel loss function and training strategy for noise-robust keyword spotting," *IEEE/ACM Trans. Audio, Speech, Lang. Process.*, vol. 29, pp. 2254–2266, Jun. 2021.

[29] A. Bellet, A. Habrard, and M. Sebban, "A survey on metric learning for feature vectors and structured data," 2013, *arXiv:1306.6709*.

[30] J. Xu, L. Luo, C. Deng, and H. Huang, "Bilevel distance metric learning for robust image recognition," in *Proc. Adv. Neural Inf. Process. Syst.*, S. H. Bengio, H. Wallach Larochelle, K. Grauman, N. Cesa-Bianchi, and R. Garnett, Eds., Curran Associates, Inc., 2018, pp. 4202–4211.

[31] H.-J. Ye, D.-C. Zhan, X.-M. Si, Y. Jiang, and Z.-H. Zhou, "What makes objects similar: A unified multi-metric learning approach," in *Proc. Adv. Neural Inf. Process. Syst.*, D. Lee, M. Sugiyama, U. Luxburg, I. Guyon, and R. Garnett, Eds., Curran Associates, Inc., 2016, pp. 1243–1251.

[32] G. Cheng, C. Yang, X. Yao, L. Guo, and J. Han, "When deep learning meets metric learning: Remote sensing image scene classification via learning discriminative CNNs," *IEEE Trans. Geosci. Remote Sens.*, vol. 56, no. 5, pp. 2811–2821, May 2018.

[33] C. Liu et al., "A new deep learning-based food recognition system for dietary assessment on an edge computing service infrastructure," *IEEE Trans. Serv. Comput.*, vol. 11, no. 2, pp. 249–261, Mar./Apr. 2018.
[34] Z.-Z. Liu et al., "Accurate and reliable service recommendation based on bilateral perception in multi-access edge computing," *IEEE Trans. Serv. Comput.*, vol. 16, no. 2, pp. 886–899, Mar./Apr. 2023.
[35] H. Sami, A. Mourad, H. Otrok, and J. Bentahar, "Demand-driven deep reinforcement learning for scalable fog and service placement," *IEEE Trans. Serv. Comput.*, vol. 15, no. 5, pp. 2671–2684, Sep./Oct. 2022.
[36] C. Wang et al., "SOLAR: Services-oriented deep learning architectures-deep learning as a service," *IEEE Trans. Serv. Comput.*, vol. 14, no. 1, pp. 262–273, Jan./Feb. 2021.
[37] L. F. Rodrigues Moreira, R. Moreira, B. A. N. Travençolo, and A. R. Backes, "An artificial intelligence-as-a-service architecture for deep learning model embodiment on low-cost devices: A case study of COVID-19 diagnosis," *Appl. Soft Comput.*, vol. 134, 2023, Art. no. 110014.
[38] F. Abramovich and M. Pensky, "Classification with many classes: Challenges and pluses," *J. Multivariate Anal.*, vol. 174, 2019, Art. no. 104536.
[39] A. Krizhevsky, "Learning multiple layers of features from tiny images," M.S. thesis, Dept. Comput. Sci., Toronto Univ., 2009.
[40] A. Singla, L. Yuan, and T. Ebrahimi, "Food/non-food image classification and food categorization using pre-trained GoogLeNet model," in *Proc. 2nd Int. Workshop Multimedia Assist. Dietary Manage.*, New York, NY, USA, 2016, pp. 3–11.
[41] M.-E. Nilsback and A. Zisserman, "Automated flower classification over a large number of classes," in *Proc. Indian Conf. Comput. Vis., Graph. Image Process.*, 2008, pp. 722–729.
[42] P. Welinder et al., "Caltech-UCSD birds 200," California Institute of Technology, Pasadena, CA, Tech. Rep. CNS-TR-2010–001, 2010.
[43] J. N. Kather et al., "Multi-class texture analysis in colorectal cancer histology," *Sci. Rep.*, vol. 6, 2016, Art. no. 27988.
[44] E. Eidinger, R. Enbar, and T. Hassner, "Age and gender estimation of unfiltered faces," *IEEE Trans. Inf. Forensics Secur.*, vol. 9, no. 12, pp. 2170–2179, Dec. 2014.
[45] Y. Ling, X. Yin, and S. Bhandarkar, "Sirface vs. fisherface: Recognition using class specific linear projection," in *Proc. Int. Conf. Image Process.*, 2003, pp. III–885.
[46] K. Li, Z. Ding, K. Li, Y. Zhang, and Y. Fu, "Vehicle and person re-identification with support neighbor loss," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 33, no. 2, pp. 826–838, Feb. 2022.

Imam Mustafa Kamal received the BS degree in informatics & computer engineering from the Electronic Engineering Polytechnic Institute of Surabaya - Sepuluh Nopember Institute of Technology (ITS), in 2015, and the MS and PhD degrees in big data from Pusan National University, in 2021. He is a postdoctoral researcher with the Institute of Intelligent Logistics Big Data, Pusan National University. His research interests include machine learning, deep learning, and data science.

Hyerim Bae (Member, IEEE) received the PhD degree in industrial engineering from Seoul National University, South Korea, in 2002. He is currently a full professor of Business Process Management with the Department of Industrial Engineering, Pusan National University, Busan, South Korea. His research interests include business process management, business intelligence, software development, information technology, cloud computing, and logistics. He is currently an associate editor in ICIC-EL, *International Journal of Innovation in Enterprise System*, and *International Journal of Innovative Computing*, *Information and Control*.

Ling Liu (Fellow, IEEE) is a full professor with the College of Computing, Georgia Institute of Technology, Atlanta, Georgia. She directs the research programs in Distributed Data Intensive Systems Lab (DiSL), examining various aspects of largescale data-intensive systems. Her current research interests include performance, availability, privacy, security and trust of Big Data systems, distributed computing, as well as Internetscale systems and applications. She has published more than 300 international journal and conference articles and is a recipient of the Best Paper Award from numerous top venues, including IEEE ICDCS, The World Wide Web Conference, IEEE Cloud, IEEE ICWS, and ACM/IEEE CCGrid. She has served on the editorial boards of over a dozen international journals. Currently, she is the editor-in-chief of *ACM Transactions on Internet Computing TOIT)*. Her current research is sponsored primarily by NSF and IBM.