# Learning Compatible Embeddings

Qiang Meng     Chixiang Zhang     Xiaoqiang Xu     Feng Zhou
Algorithm Research, AiBee Inc.

{qmeng, cxzhang, xiaoqiangxu, fzhou}@aibee.com

## Abstract

*Achieving backward compatibility when rolling out new models can highly reduce costs or even bypass feature re-encoding of existing gallery images for in-production visual retrieval systems. Previous related works usually leverage losses used in knowledge distillation which can cause performance degradations or not guarantee compatibility. To address these issues, we propose a general framework called Learning Compatible Embeddings (LCE) which is applicable for both cross model compatibility and compatible training in direct/forward/backward manners. Our compatibility is achieved by aligning class centers between models directly or via a transformation, and restricting more compact intra-class distributions for the new model. Experiments are conducted in extensive scenarios such as changes of training dataset, loss functions, network architectures as well as feature dimensions, and demonstrate that LCE efficiently enables model compatibility with marginal sacrifices of accuracies. The code will be available at* `https://github.com/IrvingMeng/LCE`.

## 1. Introduction

Visual search or retrieval systems [36, 37] are widely used in many real-world applications such as face recognition [42, 39, 33, 26, 27], person re-identification [40, 16, 34], car re-identification [19] and image retrieval [2, 12]. To obtain steady improvement, models would be occasionally upgraded by training on larger or cleaner datasets, adopting more powerful network structures and training losses, or applying techniques like network architecture search [56, 50], knowledge distillation [17] and network pruning [23, 11]. However, to harvest the benefits of new models, a process known as "backfilling" or "re-indexing" [38] is indispensable to re-encode all images in the gallery set to recreate clusters. This process could be impractical to execute when there are limited computational resources for re-encoding, or original images are legally forbidden to be preserved without user authorization. Model compatibility techniques, which can heavily reduce costs of or even by-
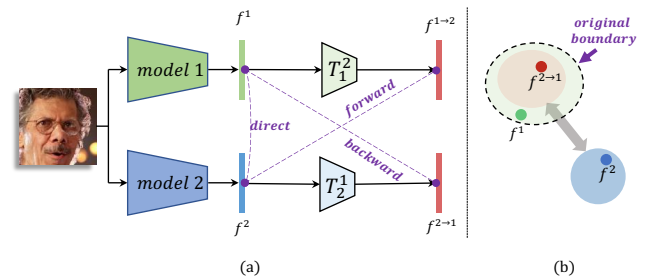


Figure 1: An overview of model compatibility problems. (a) Model compatibility matches model 1 and model 2 through three optional directions. Features $f^1$, $f^2$ from two models can make a direct comparison, noted as direct compatible method. With transformations $T_1^2$ and $T_2^1$ involved, we achieve either backward compatible by comparing $f^{2 \to 1}$ with $f^1$, or forward compatible if compare $f^{1 \to 2}$ and $f^2$. (b) Our compatibility is achieved by aligning the class centers across models through a direct comparison (by setting $T$s to identity mappings) or via transformations, and restricting more compact intra-class distributions for the new model. **Best viewed in color.**

pass the process, therefore are of great practical values.

Because of the multiplicity of application scenarios in visual search/retrieval fields, as well as compatibility directions to implement, model compatibility has never been employed via a general, unified set of patterns. Viewing in the perspective of application scenarios, there are two types of model compatibility methods. The first type is called cross model compatibility (CMC) whose goal is to find compatible mappings between the previous and upgraded models, where the upgraded models already exist. As model performance may degrade if compatibility is considered, CMC serves well for the scenarios that model performances are more valuable than compatibility. The second type is compatible training (CT) which aims at upgrading models with compatibility constraints. When bringing up against the scenarios where frequent iterations for online models are required, CT possesses an inherent feasibility to take charge.

The perspective of compatible directions categorizes model compatibility into three types: backward, forward and direct compatible methods (Fig. 1a). Direct compatible methods compare new features with old gallery sets di-

rectly, which is the most efficient as it completely prevents "backfilling" processes. Backward compatible methods endow a backward transformation that maps new features into old feature spaces, and they are capable of waiving re-encoding large gallery sets. Lightweight improvement for cumbersome model architectures is another potential scenario for backward compatible methods since old models spawn more separable feature spaces than new models in the circumstances. Forward compatible methods utilize a forward transformation that maps old features into new feature spaces, which is aimed at upgrading small gallery sets, especially when new models have much better performances than old ones.

Despite its great practical values, model compatibility is a relatively unexplored research area. Few efforts, R$^3$AN [4], RBT [44] (for CMC) and BCT [38] (for CT), are proposed and reach comparable results. However, these three works are only appropriate for limited application scenarios or compatible directions. Moreover, their model performances and compatibility are still arduous to be guaranteed, not only because of the further intensified nature of incompatibility for two different feature spaces when upgraded models dramatically changing training factors (*e.g.*, backbones, losses, training datasets, or even settings of optimizers), but also as a result of the point-wise losses they utilize, which align each feature pair and therefore prevent learning of more discriminative features.

To address the above issues, we propose a general framework called LCE for model compatibility as shown in Fig. 1. Model 1 represents the target compatible model. Model 2 is pre-existed for CMC while trained for CT. Transformations $T^1, T^2$ map features to another spaces and enable direct/backward/forward compatibility. Our compatibility is achieved by aligning feature classes across models and restricting the mapped features to be distributed within the original boundaries (*i.e.*, more compact intra-class distributions). By this means, the mapped feature $f^{2\rightarrow1}$ is distributed inside the correct class in the original feature space and that makes features comparable.

Besides the compatibility, our framework also benefits the learning of more discriminative features in the following aspects: (a) Our method works in a point-to-set manner instead of employing point-wise constraints. As shown in Fig. 1b, features from same instance are not restricted to be distributed closely. (b) By the restriction of the original boundary, intra-class distributions are more compact compared to those from the old model. (c) The introduced transformation $T$ relaxes the requirement for consistent inter-class distributions, as inter-class distributions may vary a lot across models (*e.g.*, ResNet100 [15] can separate features more easily than MobileFace [5]). Overall, we make the following contributions:

- Compared to previous methods which work in a point-

wise manner, we reformulate the model compatibility from the aspect of classes and highly decouple models. Specifically, we align feature classes across models and restrict the new-to-old mapped features to be distributed within the original boundaries. With the proposed point-to-set constraints and transformations, we achieve model compatibility with marginal sacrifices of accuracies.

- We unify the two model compatibility problems (*e.g.*, CMC and CT) into a unified training framework called LCE, which can work in direct/backward/forward manners. Extensive experiments and ablation study are conducted on both problems with various factors such as network structures and losses. The proposed method achieves noteworthy results compared to current state-of-the-arts.

## 2. Related Works

Model compatibility is essentially transferring knowledge across models. This section reviews related works in fields of transfer learning, knowledge distillation and model compatibility, from the perspective of feature embeddings.

### 2.1. Feature Representation Transfer Learning

Feature representation transfer learning aims at transforming each original feature into a new feature representation for knowledge transfer [55]. Pan *et al.* [29] propose to learn a low-dimensional latent feature space where the distributions between the source and target domain data are the same or close to each other. JDA [24] jointly adapts both the marginal distribution and conditional distribution and constructs new feature representation for substantial distribution difference. JDA is further extended by utilizing the label and structure information [18], clustering information [41], various statistical and geometrical information [53] and balanced distributions [47], *etc*. Feature augmentation based methods [7, 21, 10, 22] transfer the original features by feature replication to augment datasets for target domain. Even though is related to model compatibility, feature representation transfer learning focuses on transferring the knowledge contained in source domains to target domains instead of achieving comparable features.

### 2.2. Knowledge Distillation

Knowledge distillation (KD), which aims at transferring knowledge acquired in a teacher model to a student model, was first proposed by Hinton *et al.* [17]. There exists a great amount of knowledge sources and knowledge types in this field (see [13, 48] for recent reviews). Various knowledge sources are adopted in literature such as classification logits [17, 1, 28], hint layers [35, 54, 49, 3] and multi-layer groups [52, 51, 20]. Knowledge types can be categorized

into feature embeddings [35, 49], feature maps [3] and instance relationships [30, 32], *etc*. Despite the diverse categories, most works adopt KL-divergence or $l_2$ distance as the loss to transfer knowledge. For example, both Dark-Rank [6] and Mirzadeh *et al*. [28] learn to match softened logits between teacher model and student model using KL-divergence. PKT [31] proposes to align probability distributions in the feature spaces for representation/metric learning tasks. RKD [30] introduces distance-wise and angle-wise distillation losses that mimic structural differences in relations. CCKD [32] combines the two losses to ensure congruence between teacher and student models, in both instance-level and group-level. Their experiments are conducted on classification tasks and metric learning tasks and achieve promising results.

### 2.3. Model Compatibility

Model compatibility is to make features comparable across models and is of great practical values for visual retrieval systems. This technique can heavily reduce computational costs of re-encoding gallery sets which is large-volume in modern applications but less studied in literatures. For CMC whose goal is to find mappings between models, R$^3$AN [4] learns feature transformation by a process of reconstruction, representation and regression for face recognition. RBT [44] designs a light-weight residual bottleneck transformation module following the split-transform-merge strategy and achieves remarkable improvements compared to R$^3$AN in their experiments. Their module is trained by a classification loss, an $l_2$ similarity loss between feature embeddings and a KL-divergence loss between logits in classifiers. For CT, BCT [38] introduces an influence loss and trains new features on both new and old classifiers, which is essentially aligning the logits. As sharing the same high-level idea of transferring knowledge in KD, RBT [44] and BCT [38] both utilize similar ideas and losses to achieve their compatibility.

The main differences between KD and model compatibility mainly lie in two aspects. The first difference is that the old model in KD is a teacher model and normally performs better than the student model. For model compatibility, performances of upgraded models may decrease in some special cases such as light-weight structures or small feature dimensions introduced. However, in most scenarios, the upgrade models have potentially better performances with more data involved and technological advances. Directly distilling the exact logits or locations between each pair of old and new features, usually adopted by previous works, is too strict and could inherit bad inter-class distributions from old models to new ones. For example, class C and D in the old feature space are overlapped in Fig. 2a. Directly aligning features prevents the new model from learning more discriminative features.
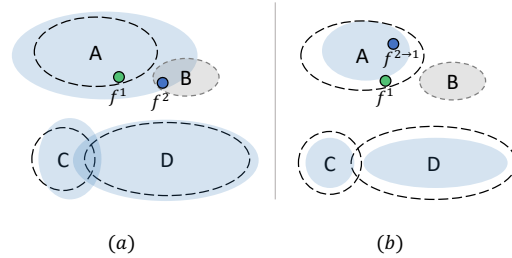


Figure 2: Model compatibility in an open-set system. $f^1, f^2$ are features from model 1, 2. Regions inside dash lines represent feature clusters from model 1 and the colored region are from model 2. In this example, class A, C, D are from the training dataset. Class B is an absent class in training dataset but can appear during test. (a) Traditional methods can lead to incompatible feature spaces. (b) Feature distributions in our method. Here $f^{2 \rightarrow 1}$ is the mapped feature, whose distribution is restricted inside the boundary of model 1.

Another difference is that traditional losses used in KD cannot fully guarantee model compatibility, as intra-class boundaries are difficult to be constrained by traditional KD losses. As shown in Fig. 2a, $f^1, f^2$ have similar logits and a small $l_2$ distance during training because of the absence of class B. Therefore, the conventional losses in KD perform weak penalties. While during the testing phase, $f^2$ belongs to cluster B and thus is not compatible with the original feature $f^1$.

Our proposed LCE framework is capable of alleviating the vulnerabilities of traditional KD losses. As Fig. 2b depicts, our method not only forces more compact intra-class distributions (class A, C, D), but also renders more discriminative inter-class distributions (class C, D are separable with new features) for the new model.

## 3. Methodology

Our LCE framework is developed following three rules: (1) Compatibility should be ensured and the issues presented in Fig. 2a are expected to be prevented whenever possible. (2) The new model should be decoupled from the previous one as much as possible to reduce performance drops. (3) The method should fit both CMC and CT problems.

### 3.1. Criterions for Compatibility

Suppose there are $N$ samples $\{x_i, y_i\}_{i=1}^N$ of $n$ classes and their embeddings from two models are $\{f_i^1, f_i^2\}_{i=1}^N$, respectively. For $\forall i_1, \forall i_2 \in \{1, 2, \cdots N\}$ and $i_1 \neq i_2$, the point-wise compatibility criterion (*e.g*., BCT [38]) is defined as

$$\text{(Point-wise)} \quad \begin{aligned} d(f_{i_1}^2, f_{i_2}^1) &\geq d(f_{i_1}^1, f_{i_2}^1), \text{ if } y_{i_1} \neq y_{i_2}, \\ d(f_{i_1}^2, f_{i_2}^1) &\leq d(f_{i_1}^1, f_{i_2}^1), \text{ if } y_{i_1} = y_{i_2}. \end{aligned} \quad (1)$$

Here $d$ measures the distances. This criterion is too strict as it adds constraints to all pairs of samples (*i.e*., $N^2 - N$ con-
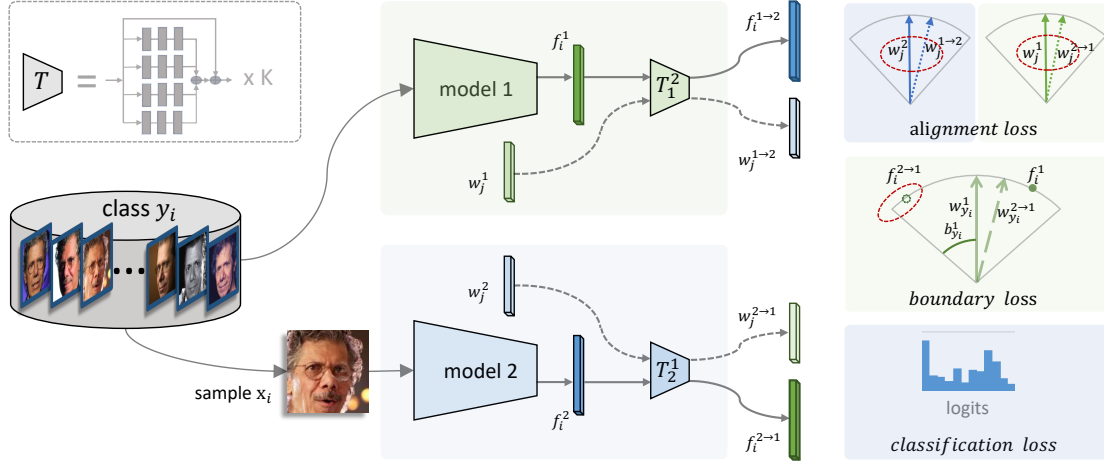
Figure 3: Overview of the proposed LCE method and the associated losses. We assume model 1 is the compatible target model and therefore its weights are fixed during the training. In the feature space of model 1, denote class $j$ has a maximum angle $b_j^1$ to its class center $w_j^1$. The model 2 can be pre-existed or to be trained based on the real situations, and transformations $T_1^2, T_2^1$ map embeddings $\{f_i\}_{i=1}^N$ and class centers $\{w_j\}_{j=1}^n$ from one feature space to another. Our compatibility is achieved by (1) an alignment loss which aligns class centers between two models, (2) a boundary loss which enforces the mapped $f_i^{2\to1}$ to have more compact intra-class distributions (formulated as the angle between $f_i^{2\to1}$ and $w_{y_i}^1$ is smaller than $b_{y_i}^1$ in our work) and (3) a classification loss.

straints) and leads to excluded solutions especially if models are trained with different networks/losses.

Our compatibility is defined differently. To start with, we first map features to another feature space, which relaxes the requirement for direct comparisons. We call the mapped features as $f_i^{1\to2}, f_i^{2\to1}$. Denote features of class $j$ from model 1 are distributed in a feature region $C_j^1$. Our criterion for compatibility is simply defined as $f_i^{2\to1} \in C_{y_i}^1$ for all $i$, which is from the perspective of point-to-set comparison. Specifically, we restrict the $f_i^{2\to1}$ to be distributed inside the corresponding original boundary as indicated in Fig. 2b. By this means, the new feature space regulates more constrictive intra-class boundaries and more diacritical inter-class distributions.

As the real feature region is unavailable (*i.e.*, a feature region contains an infinity number of features while the estimated features are of a limited number), we estimate the set $C_{y_i}^1$ by the class center $w_{y_i}^1$ and its boundary $b_{y_i}^1$ (as shown in Fig. 3, boundary loss). Then the criterion is re-formulated by

$$\text{(Point-to-set)} \quad \theta(f_i^{2\to1}, w_{y_i}^1) \leq b_{y_i}^1 \quad \forall i \in \{1, 2, \cdots, N\}. \tag{2}$$

Here $\theta(w, f) = \arccos(\frac{w^T f}{\|w\|\|f\|})$ which measures the angle between vectors $w$ and $f$.

### 3.2. Transformation Module

Besides the point-to-set compatibility, we further decouple models by transformations $T_1^2, T_2^1$ which relieve gaps between feature spaces preferred by various models. The transformations $T_1^2, T_2^1$ map features/class centers from one feature space to another and mapped embeddings are de-

noted by superscripts $1 \to 2, 2 \to 1$. Specifically, we have

$$f_i^{2\to1} = T_2^1(f_i^2), \ f_i^{1\to2} = T_1^2(f_i^1), \quad \forall j \in \{1, 2, \cdots, N\}$$
$$w_j^{2\to1} = T_2^1(w_j^2), \ w_j^{1\to2} = T_1^2(w_j^1), \quad \forall j \in \{1, 2, \cdots, n\} \tag{3}$$

With the transformations, we can map new features to the old feature space, map old features to the new feature space or direct compare features. These correspond to backward, forward and direct compatibility respectively. The structure of our used transformation module is presented in top left of Fig. 3. The module is modified from the residual bottleneck transformation module introduced in RBT [44]. Note that original module requires same feature dimension for input and output. Thus, we append one additional fully connected layer if dimension changes. Besides, the number of sub-modules $K = 4$ for backward and forward compatibility. For direct compatibility, $K = 0$ and therefore $T_1^2, T_2^1$ degrades to identity mappings.

To train the transformations, we define four types of class regions $C_j^1, C_j^2, C_j^{1\to2}, C_j^{2\to1}$. Our transformations are estimated by aligning locations of class regions $\{C_j^1, C_j^{2\to1}\}$ and $\{C_j^2, C_j^{1\to2}\}$. In the end, the module is learned by enforcing the consistency between class centers (*i.e.*, $w_j^1 = w_j^{2\to1}$ and $w_j^2 = w_j^{1\to2}$).

### 3.3. The General Framework

Fig. 3 illustrates the details of our method. As a general framework, our method can fit both CMC and CT problems. Specifically, model 1 is the compatible target model and therefore its weights are always fixed. For CMC, model 2 is also fixed and the purpose is to find mappings between

feature spaces of two models. For CT, we aim at training the model 2 while keeping compatible with model 1.

**Pre-processing.** Before training, we first pre-compute normalized feature $f_i^1$ of training sample $\{x_i, y_i\}$ from model 1. For each class $j \in \{1, 2, \cdots, n\}$ with $N_j$ samples, we calculate the class center $w_j^1$ by taking the average of the features. After that, we calculate $N_j$ angles between the class center and corresponding samples and denote them as $\{\theta_i^1\}_{i=1}^{N_j}$. Then the original boundary for class $j$ in the old feature space $b_j^1$ is estimated by

$$b_j^1 = \max\{\mathcal{I}(\theta_1^1) \cdot \theta_1^1, \mathcal{I}(\theta_2^1) \cdot \theta_2^1, \cdots, \mathcal{I}(\theta_{N_j}^1) \cdot \theta_{N_j}^1\}. \quad (4)$$

Here $\mathcal{I}(\theta)$ is 0 if $\theta$ is estimated as an outlier and 1 otherwise. The outliers are identified by the 1.5IQR rule which is a common method in statistical analysis [43].

$\{w_j^2\}_{j=1}^n$ is also pre-computed for the CMC problem. For CT, $\{w_j^2\}_{j=1}^n$ are weights in the classifier and are learned from data. As mentioned before, our compatibility is achieved by aligning class centers between new and old models via the transformation module, and force the new-learned features to be more compact than the old ones. That corresponds to the alignment loss $L_a$ and boundary loss $L_b$ in the framework. A classification loss $L_{cls}$ is further introduced to guarantee model performances.

**Alignment loss.** Define function $d(\cdot)$ as a measurement of cosine distances. Our alignment loss is used to align class locations between two models and defined as

$$L_a = \sum_{j=1}^{n} \left( d(w_j^{1 \to 2}, w_j^2) + d(w_j^{2 \to 1}, w_j^1) \right). \quad (5)$$

**Boundary loss.** The boundary loss constrains the new features to be more centralized to class centers than the old models and is defined as

$$L_b = \sum_{i=1}^{N} \max \left( 0, \theta(f_i^{2 \to 1}, w_{y_i}^1) - b_{y_i}^1 \right). \quad (6)$$

It not only contributes to the compatibility, but also leads to smaller intra-class variations and helps learns more discriminative embeddings.

**Classification loss.** The general format of a classification loss for sample $i$ is

$$L_{cls}(w, f_i) = -\log \frac{e^{s \cdot \mu(w_{y_i}, f_i)}}{e^{s \cdot \mu(w_{y_i}, f_i)} + \sum\limits_{j=1, j \neq y_i}^{n} e^{s \cdot \nu(w_j, f_i)}}. \quad (7)$$

$\mu, \nu$ generate similarity scores and are in various formats for different losses. For example, $\mu(w, f) = \nu(w, f) = \frac{w^T f}{\|w^T f\|}$ for NormFace [45] while $\mu(w, f) = \frac{w^T f}{\|w^T f\|} - m, \nu(w, f) = \frac{w^T f}{\|w^T f\|}$ for CosFace [46].

For CMC, as $f_i^1, f_i^2$ are already well-trained for classification, we compute the classification loss on $f_i^{1 \to 2}$

and precalculated class centers for model 2 (*i.e.*, $L_c = L_{cls}(w^2, f_i^{1 \to 2})$). For CT, the loss is $L_c = L_{cls}(w, f_i^2)$ where $w$ are learned during training.

**The whole loss.** In the end, the loss for the framework is

$$L_{LCE} = \lambda_a L_a + \lambda_b L_b + L_c \quad (8)$$

Here $\lambda_a, \lambda_b$ are two hyperparameters.

# 4. Experiments

In our experiments, we begin describing implementation details in Sec. 4.1 and evaluation metrics in Sec. 4.2. Sec. 4.3 is the ablation study which examines effects of the components in LCE. Then we evaluate our proposed method on CMC in Sec. 4.4. For the CT problem, only direct compatibility is studied for fair comparisons with baselines. Sec. 4.5 presents the results on CT with different training dataset, loss functions and network architectures, while experiments on different feature dimensions and sequential compatibility are in Sec. A (in the supplementary).

## 4.1. Implementation Details

**Datasets.** The original MS-Celeb-1M dataset [14] contains about 10 million images of 100k identities. However, it consists of a great many noisy face images. Instead, MS1Mv2 [9] (5.8M images, 85k identities) is adpoted as our training dataset. For change of training datasets, we also collect faces from the first 50% of identities from MS1Mv2 and call the dataset as MS1Mv2(1/2). For evaluation, we adopt a top challenging benchmark called IJB-C [25], which covers about 3,500 identities with a total of 31,334 images and 117,542 unconstrained video frames. We calculate the TARs at FAR=1e-4 in the 1:1 verification on IJB-C benchmark, where 19k positive and 15M negative matches are involved. All the images are aligned to $112 \times 112$ following the setting in ArcFace [9].

**Training.** All the models are trained on 8 1080Tis by stochastic gradient descent. For CMC, we train the transformations for 20 epochs, with learning rate initialized at 0.1 and divided by 10 at epoch 5, 10, 15. For CT, we initialize the learning rate by 0.1 and divide it by 10 at 10, 18, 22 epochs. The training stops at the 25th epoch. The weight decay is set to 5e-4 and the momentum is 0.9. The recommended hyper-parameters are used for classification losses from the original papers (*e.g.*, $m = 0.5, s = 64$ for ArcFace [9]). We only augment training samples by random horizontal flipping and empirically set $\lambda_a = 100, \lambda_b = 0.1$ for LCE.

**Baselines.** In our experiments, the main baselines are RBT [44] for CMC and BCT [38] for CT, which are current state-of-the-arts in the corresponding fields. For RBT, we use the recommended hyper-parameters in their paper. For BCT, the recommended weight for their influence loss

are not provided. We tried several values and found 0.5 is the best value in our implementation. Therefore, 0.5 is used for all BCT experiments.

## 4.2. Evaluation Metrics

**Model performance and compatibility performance.** We evaluate model performance by 1v1 verification accuracy (abbreviated as Ver. Acc. in all tables) on the IJB-C benchmark. To measure the compatibility performance between the old model $\phi^1$ and the new model $\phi^2$, we use $\phi^1$ to extract the feature embedding for the first template in the pair and $\phi^2$ for the second, and report the cross verification accuracy (abbreviated as Cross Veri. Acc. in all tables).

**Performance gain.** For the scenario of CT, the performance of the upgraded model is also vital to the whole system besides the compatibility. Therefore, we report the relative performance improvement from $\phi^1$ to $\phi^2$ by

$$\mathcal{G}_{per.} = \frac{M(\phi^2) - M(\phi^1)}{|M(\phi^2_{upper}) - M(\phi^1)|}. \quad (9)$$

Here $M(\phi)$ measures the recognition performance of model $\phi$. Compared to $\phi^2$, $\phi^2_{upper}$ is trained without compatibility purpose and therefore serves as the upper bound for the new model. We add the absolute symbol to the denominator as it can be negative (*i.e.*, $M(\phi^2_{upper}) < M(\phi^1)$). For example, the previous deployed model has a high latency and the new model is of a light-weight structure. In this case, the model performance will degrade.

**Upgrade gain.** The upgrade gain is similarly defined in BCT [38]. We revise their definition by taking the absolute value of the denominator to handle the model degradation and define our upgrade gain as

$$\mathcal{G}_{upgrade} = \frac{M(\phi^2, \phi^1) - M(\phi^1)}{|M(\phi^2_{upper}) - M(\phi^1)|}. \quad (10)$$

Here $M(\phi^2, \phi^1)$ measures the cross-model recognition performances between $\phi^2$ and $\phi^1$ (*i.e.*, compatibility performances). If using transformations, upgrade gain is the better result of backward and forward compatibility.

## 4.3. Ablation Study

In this part, we conduct several ablation studies to investigate the effects of different components in our method. ResNet50 [15] is adopted as the backbone network, and we use the full version of MS1Mv2 as the training data. The classification loss used in the old model is NormFace [45] and that used in new models is ArcFace [9].

**Effects of components in LCE.** We train the new models with different combinations of alignment loss, boundary loss as well as the transformation module as shown in Tab. 1. Note that $\phi^2_{upper}$ is trained directly without additional losses and therefore serves as the upper bound of

the recognition performances for new models. The performance gain is 84.28% and the upgrade gain is 60.82% with only alignment loss used. If introducing boundary loss, the performance gain is 89.07% and upgrade gain increases evidently with the number of 7.52%. The transformation module can significantly increase the model performance as it loose the constraints of directly aligning class locations. Performance gains of $\phi^2_{a\_t}$ and $\phi^2_{a\_b\_t}$ are all over 96%, which is close to the upper bound. Also, the transformation module offers the choices of whether to transform features forward and backward. As the full version of LCE, $\phi^2_{a\_b\_t}$ achieves a verification accuracy of 91.92% when transforming new features to the original feature space, and 93.11% in the opposite direction. In addition, $\phi^2_{a\_b\_t}$ achieves the best performance gain of 99.32% and upgrade gain of 72.44%, which demonstrates the efficacy of our proposed method.
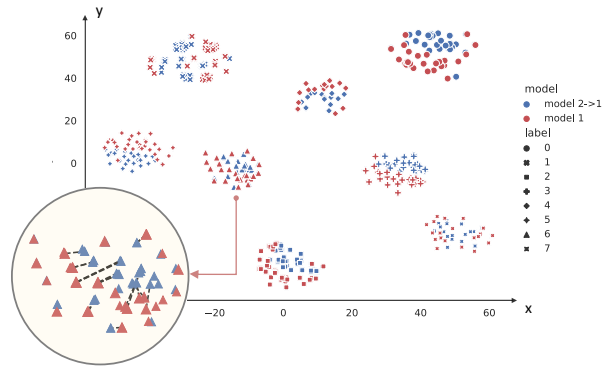


Figure 4: A visualization of deep features from two models. The area of class 6 is magnified for better revealing pair information. Each dash line connects features of a same instance from two models. **Best viewed in color**.

**Visualization.** To reveal the effects of the proposed method, we sample 8 classes from MS1Mv2 and visualize deep features in Fig. 4. Here model 1 is $\phi^1$ and model 2 is $\phi^2_{a\_b\_t}$ from Tab. 1. Model 2→1 produces features by mapping those from model 2 to the feature space of model 1 by the learned transformations. T-SNE is chosen to map high-dimensional features into top 2 dimensions. The figure demonstrates that the mapped features are well-aligned with the original features and have more compact intra-class distributions than original ones.

We further pick class 6 and visualize feature pairs of its first 10 samples to show their pair-wise relationships. Features of the same image from two models can be either close to or far away from each other, as shown in the figure, which reveals our compatibility works in a point-to-set instead of strict point-wise manner.

Tab. 2 presents the intra-class and inter-class distances of three types of features. The average intra-class distance is calculated by $\frac{1}{N} \sum_{i=1}^{N} (\frac{f_i}{\|f_i\|} - \frac{w_{y_i}}{\|w_{y_i}\|})^2$ and the average inter-class distance is calculated by

| New Model | Old Model | Losses | | | Transformation | Veri. | Cross Veri. Acc. | | | Perf. | Upgrade |
|---|---|---|---|---|---|---|---|---|---|---|---|
| | | Alignment | Boundary | Classification | | Acc. | Direct | Backward | Forward | Gain (%) | Gain (%) |
| $\phi^1$ | - | - | - | NormFace | - | 89.93 | - | - | - | - | - |
| $\phi_a^2$ | $\phi^1$ | ✓ | | ArcFace | | 93.63 | 92.60 | - | - | 84.28 | 60.82 |
| $\phi_{a\_b}^2$ | $\phi^1$ | ✓ | ✓ | ArcFace | | 93.84 | 92.93 | - | - | 89.07 | 68.34 |
| $\phi_{a\_t}^2$ | $\phi^1$ | ✓ | | ArcFace | ✓ | 94.16 | - | 89.25 | 92.89 | 96.36 | 67.43 |
| $\phi_{a\_b\_t}^2$ | $\phi^1$ | ✓ | ✓ | ArcFace | ✓ | 94.29 | - | 91.92 | 93.11 | **99.32** | **72.44** |
| $\phi_{upper}^2$ | - | - | - | ArcFace | - | 94.32 | 0.01 | - | - | +100.00 | - |

Table 1: 1:1 verification TAR (%@FAR=1e-4) on the IJB-C dataset with different combinations of components.

| | model 1 | model 2→1 | model 2 |
|---|---|---|---|
| intra-class | 0.347 | 0.283 | 0.277 |
| inter-class | 1.767 | 1.792 | 1.827 |

Table 2: Average intra-class and inter-class distances of features from different models.

| Model | Backbone | Loss | Training Dataset | Ver. Acc. |
|---|---|---|---|---|
| $\phi_a^*$ | ResNet100 | ArcFace [9] | MS1Mv2 | 95.72 |
| $\phi_b^*$ | ResNet50 | ArcFace [9] | MS1M-REFINE-v1 | 92.52 |
| $\phi_c^*$ | ResNet50 | Subcenter [8] | MS1Mv0 | 95.44 |

Table 3: Details of selected pretrained models. Here MS1Mv0 is the original MS-Celeb-1M dataset [14] while the other two datasets are cleaned versions by authors of ArcFace [9]. Verification accuracies (%) are TARs@FAR=1e-4 on IJB-C.

$\frac{1}{n(n-1)} \sum_i^n \sum_{j=i+1}^n \left( \frac{w_i}{\|w_i\|} - \frac{w_j}{\|w_j\|} \right)^2$. Because of the proposed alignment and boundary loss, model 2→1 shrinks the distribution of each class and therefore its features have much smaller intra-class distances than the original model 1 (0.283 *vs.* 0.347). The transformations further add flexibility of class locations and helps model to learn more separated inter-class distributions. The average inter-class distance increases from 1.767 to 1.827 in this case.

## 4.4. Cross Model Compatibility

For CMC, we adopt three pretrained models provided by the InsightFace project[1], which are of different network architectures and trained on different version of MS1M datasets as well as different losses. Details of the models and their abbreviations can be found in Tab. 3. We select NormFace [45] and ArcFace [9] as the classification loss to train the transformations and present the experimental results in Tab. 4.

RBT maps features from two models to a third feature space. In contrast, our LCE maps feature forward or backward and therefore has two cross verification accuracies in each experiment. In all results, $(\phi_a^*, \phi_c^*)$ always have the best cross verification accuracies than other two combinations. That indicates compatibility performances are also influenced by performances of source models. When using ArcFace, both methods achieve similar compatibility per-

---

[1]https://github.com/deepinsight/insightface/

| Method | Classification Loss | $(\phi_a^*, \phi_b^*)$ | $(\phi_a^*, \phi_c^*)$ | $(\phi_b^*, \phi_c^*)$ |
|---|---|---|---|---|
| direct | - | 0.01 | 0.01 | 0.03 |
| RBT [44] | ArcFace | 94.04 | **95.29** | 93.68 |
| LCE | ArcFace | **94.08**/93.90 | 95.19/95.27 | **93.72**/93.51 |
| RBT [44] | NormFace | 92.78 | 94.01 | 92.33 |
| LCE | NormFace | **94.08**/93.86 | **95.20**/95.02 | **93.72**/93.40 |

Table 4: 1:1 verification TAR (%@FAR=1e-4) on the IJB-C for cross model compatibility.

formances. While using NormFace, the performances of RBT drop dramatically with more than 1% for all cases. That is because RBT compare features in the third feature space and therefore the performance is heavily relied on the goodness of that feature space. In contrast, our method is robust to changes of classification losses and achieves consistent high cross verification accuracies. Besides the robustness, the computational costs of LCE are only half of RBT. Our method is also more flexible in real-world applications as one can choose to either re-encode the feature in gallery sets or map new feature to the old feature space.

## 4.5. Compatible Training

In this section and Sec. A in the supplementary, we explore the scenario of CT. The new models are upgraded from a compatible target model with the change of different training datasets, classification losses, network architectures as well as the feature dimensions. We adopt two baselines: (1) a naive KD baseline which uses $l^2$-distance and (2) BCT [38]. For fair comparison with BCT [38], we set $K = 0$ in the transformation module and only consider direct compatibility. We call the old model as $\phi^1$, the new model without compatibility losses as $\phi_{upper}^2$ and with $l_2$ baseline, BCT, LCE as $\phi_{l2}^2, \phi_{bct}^2, \phi_{lce}^2$, respectively. In all the experiments below, cross verification accuracies between $\phi^1$ and $\phi_{upper}^2$ are all near 0, which demonstrates that model compatibility cannot be directly achieved without usage of compatibility methods.

Unless stated otherwise, the default setting is to employ MS1Mv2 as the training dataset, ResNet50 [15] as the backbone network, ArcFace as the classification loss and feature dimension as 512. Below we present results with one factor changed each time while others are set to default. Note that

$\phi^1$ in different tables may refer to different models.

**Changes of training datasets.** The old model $\phi^1$ is trained on MS1Mv2(1/2) and all the new models are trained with the full version of MS1Mv2. Tab. 5 presents the results. $\phi^2_{l_2}$ directly matches features and therefore has a performance similar to the previous one. $\phi^2_{bct}$ has a performance gain of 47.06% but its upgrade gain is close to $\phi^2_{l_2}$ (both around 23%). A possible reason is that the old model is trained on a reduced dataset and therefore cannot clearly separate classes in the full data. The point-wise compatibility constraints used in baselines prevent simultaneous improving recognition performances and achieving model compatibility. In contrast, we use a point-to-set constraint and that makes our $\phi^2_{lce}$ achieve much better results. The gains are 84.31% and 81.37% respectively in our implementation.

| Model | Veri. Acc. | Cross Veri. Acc. | Perf. Gain (%) | Upgrade Gain (%) |
|---|---|---|---|---|
| $\phi^1$ | 93.16 | - | - | - |
| $\phi^2_{l2}$ | 92.92 | 93.40 | -23.53 | +23.53 |
| $\phi^2_{bct}$ | 93.64 | 93.39 | +47.06 | +22.54 |
| $\phi^2_{lce}$ | 94.02 | 93.99 | **+84.31** | **+81.37** |
| $\phi^2_{upper}$ | 94.18 | 0.02 | +100.00 | - |

Table 5: 1:1 verification TAR (%@FAR=1e-4) on the IJB-C dataset with changes of training datasets.

**Changes of classification losses.** With old model trained on ArcFace, we train new models on different classification losses. We pick NormFace [45], CosFace [46] and report the results in Tab. 6. The old model is trained on the best classification loss and therefore can separate classes well. $\phi^2_{l2}$ trained by the naive approach performs poorly in this scenario. Our $\phi^2_{lce}$ still achieve better results especially for the model compatibility. In particular, the upgrade gains are 27.69% and 12.76% higher than those from $\phi^2_{bct}$ when training with CosFace.

| Model | Classification Loss | Veri. Acc. | Cross Veri. Acc. | Perf. Gain (%) | Upgrade Gain (%) |
|---|---|---|---|---|---|
| $\phi^1$ | ArcFace | 94.18 | - | - | - |
| $\phi^2_{l2}$ | CosFace | 92.79 | 93.09 | -106.92 | -83.84 |
| $\phi^2_{bct}$ | CosFace | 93.56 | 93.20 | -47.69 | -75.38 |
| $\phi^2_{lce}$ | CosFace | 93.53 | 93.56 | **-50.00** | **-47.69** |
| $\phi^2_{upper}$ | CosFace | 92.88 | 0.01 | -100.00 | - |
| $\phi^2_{l2}$ | NormFace | 89.28 | 92.32 | -192.91 | -73.22 |
| $\phi^2_{bct}$ | NormFace | 90.42 | 93.25 | -97.92 | -24.22 |
| $\phi^2_{lce}$ | NormFace | 90.58 | 93.74 | **-93.75** | **-11.46** |
| $\phi^2_{upper}$ | NormFace | 90.34 | 0.00 | -100.00 | - |

Table 6: 1:1 verification TAR (@FAR=1e-4) on the IJB-C dataset with changes of classification losses.

Another interesting phenomenon is that the verification performances of $\phi^2_{lce}, \phi^2_{bct}$ are higher than the $\phi^2_{upper}$. This

| Model | Backbone | Veri. Acc. | Cross Veri. Acc. | Perf. Gain (%) | Upgrade Gain (%) |
|---|---|---|---|---|---|
| $\phi^1$ | ResNet50 | 94.18 | - | - | - |
| $\phi^2_{l2}$ | ResNet18 | 89.52 | 90.10 | -119.79 | -104.88 |
| $\phi^2_{bct}$ | ResNet18 | 88.78 | 92.06 | -138.82 | -54.49 |
| $\phi^2_{lce}$ | ResNet18 | 89.71 | 92.81 | **-114.91** | **-35.22** |
| $\phi^2_{upper}$ | ResNet18 | 90.29 | 0.01 | -100.00 | - |
| $\phi^2_{l2}$ | MobileFace | 87.60 | 83.93 | -109.67 | -170.83 |
| $\phi^2_{bct}$ | MobileFace | 87.10 | 91.46 | -118.00 | **-45.33** |
| $\phi^2_{lce}$ | MobileFace | 87.84 | 91.31 | **-105.66** | -47.50 |
| $\phi^2_{upper}$ | MobileFace | 88.18 | 0.00 | -100.00 | - |
| $\phi^2_{l2}$ | ResNet100 | 94.40 | 93.43 | +22.45 | -76.53 |
| $\phi^2_{bct}$ | ResNet100 | 94.01 | 94.69 | -17.35 | +52.04 |
| $\phi^2_{lce}$ | ResNet100 | 94.64 | 95.07 | **+46.94** | **+90.82** |
| $\phi^2_{upper}$ | ResNet100 | 95.16 | 0.03 | +100.00 | - |

Table 7: 1:1 verification TAR (%@FAR=1e-4) on the IJB-C dataset with changes of network architectures.

is probably because the old model performs better than $\phi^2_{upper}$, the compatibility losses in $\phi^2_{lce}, \phi^2_{bct}$ help transferring good knowledge to new models. In Tab. 7, the old model is also better than $\phi^2_{upper}$ with ResNet50 when new models have structures of ResNet18 or MobileFace. However, we do not observe similar phenomenon. We guess the reason is that compared to losses, knowledge is hard to transfer across models with different network architectures. **Changes of network architectures.** In this part, we study the model compatibility across different network architectures. The old model is trained with ResNet50 while the new models are trained with ResNet100, ResNet18 [15] or MobileFaceNet [5]. Results are presented in Tab. 7. In all cases, $\phi^2_{l2}$ has good performance gains while poor upgrade gains. The gains of $\phi^2_{bct}$ are in completely opposite patterns. In contrast, our $\phi^2_{lce}$ achieves remarkable results for both recognition and compatibility performance, which demonstrates the superiority of the proposed method.

## 5. Conclusion

In this paper, we design a general framework called LCE for model compatibility. Our compatibility works in a point-to-set manner which is realized by aligning the class centers between models and restricting more compact intra-class distributions for the new model. The adequate experimental results demonstrate that our method can achieve model compatibility in various scenarios while with marginal sacrifices of accuracies. As a general framework, our method can be applied to different combinations of scenarios (CMC/CT) and compatible directions (direct/forward/backward). Moreover, it can be potentially extended to benefit other tasks such as person/car re-identification and image retrieval.

# References

[1] Lei Jimmy Ba and Rich Caruana. Do deep nets really need to be deep? *arXiv preprint arXiv:1312.6184*, 2013.

[2] Abhijit Bendale and Terrance E Boult. Towards open set deep networks. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 1563–1572, 2016.

[3] Hanting Chen, Yunhe Wang, Chang Xu, Chao Xu, and Dacheng Tao. Learning student networks via feature embedding. *IEEE Transactions on Neural Networks and Learning Systems*, 2020.

[4] Ken Chen, Yichao Wu, Haoyu Qin, Ding Liang, Xuebo Liu, and Junjie Yan. R3 adversarial network for cross model face recognition. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 9868–9876, 2019.

[5] Sheng Chen, Yang Liu, Xiang Gao, and Zhen Han. Mobile-facenets: Efficient cnns for accurate real-time face verification on mobile devices. In *Chinese Conference on Biometric Recognition*, pages 428–438. Springer, 2018.

[6] Yuntao Chen, Naiyan Wang, and Zhaoxiang Zhang. Darkrank: Accelerating deep metric learning via cross sample similarities transfer. In *AAAI Conference on Artificial Intelligence*, 2018.

[7] Hal Daumé III. Frustratingly easy domain adaptation. *arXiv preprint arXiv:0907.1815*, 2009.

[8] Jiankang Deng, Jia Guo, Tongliang Liu, Mingming Gong, and Stefanos Zafeiriou. Sub-center arcface: Boosting face recognition by large-scale noisy web faces. In *European Conference on Computer Vision*, 2020.

[9] Jiankang Deng, Jia Guo, Niannan Xue, and Stefanos Zafeiriou. ArcFace: Additive angular margin loss for deep face recognition. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 4690–4699, 2019.

[10] Lixin Duan, Dong Xu, and Ivor Tsang. Learning with augmented features for heterogeneous domain adaptation. *arXiv preprint arXiv:1206.4660*, 2012.

[11] Jonathan Frankle and Michael Carbin. The lottery ticket hypothesis: Finding sparse, trainable neural networks. *arXiv preprint arXiv:1803.03635*, 2018.

[12] Albert Gordo, Jon Almazán, Jerome Revaud, and Diane Larlus. Deep image retrieval: Learning global representations for image search. In *European Conference on Computer Vision*, pages 241–257. Springer, 2016.

[13] Jianping Gou, Baosheng Yu, Stephen John Maybank, and Dacheng Tao. Knowledge distillation: A survey. *arXiv preprint arXiv:2006.05525*, 2020.

[14] Yandong Guo, Lei Zhang, Yuxiao Hu, Xiaodong He, and Jianfeng Gao. MS-Celeb-1M: A dataset and benchmark for large-scale face recognition. In *European Conference on Computer Vision*, pages 87–102. Springer, 2016.

[15] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 770–778, 2016.

[16] Alexander Hermans, Lucas Beyer, and Bastian Leibe. In defense of the triplet loss for person re-identification. *arXiv preprint arXiv:1703.07737*, 2017.

[17] Geoffrey Hinton, Oriol Vinyals, and Jeff Dean. Distilling the knowledge in a neural network. *arXiv preprint arXiv:1503.02531*, 2015.

[18] Cheng-An Hou, Yao-Hung Hubert Tsai, Yi-Ren Yeh, and Yu-Chiang Frank Wang. Unsupervised domain adaptation with label and structural consistency. *IEEE Transactions on Image Processing*, 25(12):5552–5562, 2016.

[19] Sultan Daud Khan and Habib Ullah. A survey of advances in vision-based vehicle re-identification. *Computer Vision and Image Understanding*, 182:50–63, 2019.

[20] Jangho Kim, SeoungUK Park, and Nojun Kwak. Paraphrasing complex network: Network compression via factor transfer. In *Annual Conference on Neural Information Processing Systems*, 2018.

[21] Abhishek Kumar, Avishek Saha, and Hal Daume. Co-regularization based semi-supervised domain adaptation. *Annual Conference on Neural Information Processing Systems*, 23:478–486, 2010.

[22] Wen Li, Lixin Duan, Dong Xu, and Ivor W Tsang. Learning with augmented features for supervised and semi-supervised heterogeneous domain adaptation. *IEEE Trans on Pattern Analysis and Machine Intelligence*, 36(6):1134–1148, 2013.

[23] Zhuang Liu, Mingjie Sun, Tinghui Zhou, Gao Huang, and Trevor Darrell. Rethinking the value of network pruning. *arXiv preprint arXiv:1810.05270*, 2018.

[24] Mingsheng Long, Jianmin Wang, Guiguang Ding, Jiaguang Sun, and Philip S Yu. Transfer feature learning with joint distribution adaptation. In *International Conference on Computer Vision*, pages 2200–2207, 2013.

[25] Brianna Maze, Jocelyn Adams, James A Duncan, Nathan Kalka, Tim Miller, Charles Otto, Anil K Jain, W Tyler Niggel, Janet Anderson, Jordan Cheney, et al. IARPA Janus benchmark-C: Face dataset and protocol. In *International Conference on Biometrics*, pages 158–165. IEEE, 2018.

[26] Qiang Meng, Xiaqing Xu, Xiaobo Wang, Yang Qian, Yunxiao Qin, Zezheng Wang, Chenxu Zhao, Feng Zhou, and Zhen Lei. PoseFace: Pose-invariant features and pose-adaptive loss for face recognition. *arXiv preprint arXiv:2107.11721*, 2021.

[27] Qiang Meng, Shichao Zhao, Zhida Huang, and Feng Zhou. MagFace: A universal representation for face recognition and quality assessment. In *IEEE Conference on Computer Vision and Pattern Recognition*, 2021.

[28] Seyed Iman Mirzadeh, Mehrdad Farajtabar, Ang Li, Nir Levine, Akihiro Matsukawa, and Hassan Ghasemzadeh. Improved knowledge distillation via teacher assistant. In *AAAI Conference on Artificial Intelligence*, 2020.

[29] Sinno Jialin Pan, James T Kwok, Qiang Yang, et al. Transfer learning via dimensionality reduction. In *AAAI Conference on Artificial Intelligence*, volume 8, pages 677–682, 2008.

[30] Wonpyo Park, Dongju Kim, Yan Lu, and Minsu Cho. Relational knowledge distillation. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 3967–3976, 2019.

[31] Nikolaos Passalis, Maria Tzelepi, and Anastasios Tefas. Probabilistic knowledge transfer for lightweight deep representation learning. *IEEE Transactions on Neural Networks and Learning Systems*, 2020.

[32] Baoyun Peng, Xiao Jin, Jiaheng Liu, Dongsheng Li, Yichao Wu, Yu Liu, Shunfeng Zhou, and Zhaoning Zhang. Correlation congruence for knowledge distillation. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 5007–5016, 2019.

[33] Rajeev Ranjan, Carlos D Castillo, and Rama Chellappa. L2-constrained softmax loss for discriminative face verification. *arXiv preprint arXiv:1703.09507*, 2017.

[34] Ergys Ristani and Carlo Tomasi. Features for multi-target multi-camera tracking and re-identification. In *European Conference on Computer Vision*, pages 6036–6046, 2018.

[35] Adriana Romero, Nicolas Ballas, Samira Ebrahimi Kahou, Antoine Chassang, Carlo Gatta, and Yoshua Bengio. Fitnets: Hints for thin deep nets. *arXiv preprint arXiv:1412.6550*, 2014.

[36] Walter J Scheirer, Anderson de Rezende Rocha, Archana Sapkota, and Terrance E Boult. Toward open set recognition. *IEEE Trans on Pattern Analysis and Machine Intelligence*, 35(7):1757–1772, 2012.

[37] Walter J Scheirer, Lalit P Jain, and Terrance E Boult. Probability models for open set recognition. *IEEE Trans on Pattern Analysis and Machine Intelligence*, 36(11):2317–2324, 2014.

[38] Yantao Shen, Yuanjun Xiong, Wei Xia, and Stefano Soatto. Towards backward-compatible representation learning. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 6368–6377, 2020.

[39] Yi Sun, Ding Liang, Xiaogang Wang, and Xiaoou Tang. DeepID3: Face recognition with very deep neural networks. *arXiv preprint arXiv:1502.00873*, 2015.

[40] Yifan Sun, Liang Zheng, Yi Yang, Qi Tian, and Shengjin Wang. Beyond part models: Person retrieval with refined part pooling (and a strong convolutional baseline). In *European Conference on Computer Vision*, 2018.

[41] Jafar Tahmoresnezhad and Sattar Hashemi. Visual domain adaptation via transfer feature learning. *Knowledge and Information Systems*, 50(2):585–605, 2017.

[42] Yaniv Taigman, Ming Yang, Marc'Aurelio Ranzato, and Lior Wolf. DeepFace: Closing the gap to human-level performance in face verification. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 1701–1708, 2014.

[43] John W Tukey et al. *Exploratory data analysis*, volume 2. Reading, Mass., 1977.

[44] Chien-Yi Wang, Ya-Liang Chang, Shang-Ta Yang, Dong Chen, and Shang-Hong Lai. Unified representation learning for cross model compatibility. *arXiv preprint arXiv:2008.04821*, 2020.

[45] Feng Wang, Xiang Xiang, Jian Cheng, and Alan Loddon Yuille. Normface: L2 hypersphere embedding for face verification. In *ACM International Conference on Multimedia*, pages 1041–1049, 2017.

[46] Hao Wang, Yitong Wang, Zheng Zhou, Xing Ji, Dihong Gong, Jingchao Zhou, Zhifeng Li, and Wei Liu. CosFace: Large margin cosine loss for deep face recognition. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 5265–5274, 2018.

[47] Jindong Wang, Yiqiang Chen, Shuji Hao, Wenjie Feng, and Zhiqi Shen. Balanced distribution adaptation for transfer learning. In *IEEE international conference on data mining*, pages 1129–1134. IEEE, 2017.

[48] Lin Wang and Kuk-Jin Yoon. Knowledge distillation and student-teacher learning for visual intelligence: A review and new outlooks. *IEEE Trans on Pattern Analysis and Machine Intelligence*, 2021.

[49] Xiaobo Wang, Tianyu Fu, Shengcai Liao, Shuo Wang, Zhen Lei, and Tao Mei. Exclusivity-consistency regularized knowledge distillation for face recognition. In *European Conference on Computer Vision*, 2020.

[50] Xiaqing Xu, Qiang Meng, Yunxiao Qin, Jianzhu Guo, Chenxu Zhao, Feng Zhou, and Zhen Lei. Searching for alignment in face recognition. In *AAAI Conference on Artificial Intelligence*, 2020.

[51] Kaiyu Yue, Jiangfan Deng, and Feng Zhou. Matching guided distillation. In *European Conference on Computer Vision*, 2020.

[52] Sergey Zagoruyko and Nikos Komodakis. Paying more attention to attention: Improving the performance of convolutional neural networks via attention transfer. *arXiv preprint arXiv:1612.03928*, 2016.

[53] Jing Zhang, Wanqing Li, and Philip Ogunbona. Joint geometrical and statistical alignment for visual domain adaptation. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 1859–1867, 2017.

[54] Guorui Zhou, Ying Fan, Runpeng Cui, Weijie Bian, Xiaoqiang Zhu, and Kun Gai. Rocket launching: A universal and efficient framework for training well-performing light net. In *AAAI Conference on Artificial Intelligence*, 2018.

[55] Fuzhen Zhuang, Zhiyuan Qi, Keyu Duan, Dongbo Xi, Yongchun Zhu, Hengshu Zhu, Hui Xiong, and Qing He. A comprehensive survey on transfer learning. *Proceedings of the IEEE*, 109(1):43–76, 2020.

[56] Barret Zoph and Quoc V Le. Neural architecture search with reinforcement learning. *arXiv preprint arXiv:1611.01578*, 2016.