Heterogeneous Double-Head Ensemble for Deep Metric Learning

YOUNGMIN RO, (Member, IEEE), AND JIN YOUNG CHOI, (Member, IEEE)
Automation and Systems Research Institute, Seoul 08826, South Korea
Department of Electrical and Computer Engineering, Seoul National University, Seoul 08826, South Korea
Corresponding author: Jin Young Choi (jychoi@snu.ac.kr)

This work was supported by Institute for Information & communications Technology Promotion (IITP) grant funded by the Ministry of Science and ICT (No. 2017-0-00306, Outdoor Surveillance Robots) and Next-Generation Information Computing Development Program through National Research Foundation (NRF) of Korea funded by the Ministry of Science and ICT under Grant 2017M3C4A7077582.

ABSTRACT 

The structure of a multi-head ensemble has been employed by many algorithms in various applications including deep metric learning. However, their structures have been empirically designed in a simple way such as using the same head structure, which leads to a limited ensemble effect due to lack of head diversity. In this paper, for an elaborate design of the multi-head ensemble structure, we establish design concepts based on three structural factors: designing the feature layer for extracting the ensemble-favorable feature vector, designing the shared part for memory savings, and designing the diverse multi-heads for performance improvement. Through rigorous evaluation of variants on the basis of the design concepts, we propose a heterogeneous double-head ensemble structure that drastically increases ensemble gain along with memory savings. In verifying experiments on image retrieval datasets, the proposed ensemble structure outperforms the state-of-the-art algorithms by margins of over 5.3%, 6.1%, 5.9%, and 1.8% in CUB-200, Car-196, SOP, and Inshop, respectively.

INDEX TERMS

Ensemble learning, multi-head structure, deep metric learning, deep architecture design, image retrieval.

I. INTRODUCTION

Deep metric learning has been successfully used in various applications related to computer vision, such as image retrieval [1]–[10], person re-identification [11], [12], and face verification [13]–[15]. Deep metric learning refers to the design of feature extracting functions with deep neural networks so that the features of semantically similar images are close to others. Ensemble is a method of ensuring robust performance by training diverse models and aggregating their prediction results.

For deep metric learning, the ensemble requires two or more deep networks that extract diverse feature vectors [16]. A variety of feature vectors can be obtained by semantically diverse attention of the input image [11], [17] or by applying a loss to keep feature vectors away from others [2], [18]. The diversity can also be achieved by making networks converge to diverse local minima [19]. The goal of the ensemble is to generate a synergy between those diverse feature vectors [16]. However, such ensemble methods have a crucial disadvantage: the multiple models require considerable memory and heavy computation.

To alleviate the disadvantage, multi-head structures have been employed for many applications [20], [21], including deep metric learning [3], [11], [22], [23]. Multi-head structures can effectively reduce memory usage by sharing a part of low-level layers. However, most previous studies
simply use multiple heads having the same structure by copying the original block for each head, which leads to a limited ensemble effect due to lack of head diversity.

In this paper, to accommodate the trade-off between memory saving and performance, we present an elaborate design concept. By examining the trade-off based on design concepts, we search for an effective multi-head ensemble structure referred to as Heterogeneous Double-head Ensemble (HDhE), as shown in Figure 1. In our design, we consider three factors influential to memory saving and performance: designing a dimension-reduced feature vector, determining the shared body part, and designing diverse heads to generate diverse feature vectors favorable to the ensemble. Considering these, we prepare design concepts and we conduct rigorous evaluations on various variants chosen on the basis of design concepts. Based on such evaluation of the trade-off between memory saving and the ensemble performance, we determine the dimension-reduced feature vector, the low-level shared structure, and the heterogeneous structure of heads.

To show the advantages of our HDhE structure, we present the following experimental results. Our HDhE structure outperforms the baseline ensemble of two independent networks by over 4% in the CUB-200 dataset (Recall@1), using only one-quarter of baseline feature dimension and three-quarters of baseline memory usage. In addition, on four kinds of image retrieval benchmark datasets, our HDhE outperforms not only existing ensemble methods but also the state-of-the-art methods by large margins. Besides, we show that the HDhE is valid for other types of loss, such as triplet loss, even if it is designed only by using softmax loss.

The contributions of our work are summarized as follows:

- We find influential design factors to design an effective multi-head ensemble structure for memory saving and performance improvement.
- We establish the design concepts encompassing design factors to search the target ensemble structure.
- Following the design concepts, we propose a heterogeneous double-head ensemble structure that provides memory savings and renews the state-of-the-art performance.

II. RELATED WORKS

A. ENSEMBLE FOR THE DEEP METRIC LEARNING

For the deep metric learning, several studies \[1\], \[2\] employed an ensemble method to compensate for weaknesses of weak models. Yuan et al. \[1\] stated that even if it is the same sample, the difficulty evaluation of the sample is different depending on the complexity of the network. In their work, networks of different complexity successively evaluate samples, and the most complex network learns only hard samples. Opitz et al. \[2\] modified the network to extract different feature vectors by dividing the last layer into several sub-layers. Each sub-layer learns in a successive manner, in which the previous layer learns the samples and the next layer learns the re-weighed sample through the gradient generated from the previous layer. However, the algorithms \[4\], \[8\], which appeared later, can achieve higher performance without ensemble than the above algorithms, while using the same backbone network (inception-v1).

Meanwhile, Xuan et al. \[5\] proposed an ensemble algorithm that creates meta-classes through random grouping, and the independent models learn the different meta-class data. However, the strategy that renders the best performance of the algorithm uses 48 independent networks, which proves to be a disadvantage. This means that the computational power of 48 independent models is needed for one inference. On the other hand, our multi-head structural ensemble model requires fewer parameters compared to two independent models. ABE \[3\] is an ensemble algorithm that uses a multi-head structure for the deep metric learning. This algorithm introduces a new loss function in which the feature vectors of each head push each other. However, their 8-head ensemble performance designed with inception-v1 also lags behind the current single algorithm with the same backbone. In this paper, the proposed structure outperforms not only the existing ensemble algorithms but also the state-of-the-art algorithms by a large margin on four image retrieval benchmarks.

B. MULTI-HEAD ENSEMBLE STRUCTURE

Studies have used a multi-head structure by expanding high-level layers of the backbone network in various tasks such as person re-identification, image classification, and deep metric learning \[3\], \[11\], \[20\]–\[23\]. The shared layer structure of the multi-head ensemble is different for each study. For some examples, one study \[23\], using the ResNet-50 network, used the layers until Res3_2 as a shared layer and the other layers as multi-heads. In another study \[20\] using the ResNet-110 network, the layers until Res3_18 were used as a shared layer. There was also study \[3\] that used Inception-V1 network, where layers until inception_3b were used as shared layers. However, these studies did not clearly state how their multi-head structure was designed and employed. In this paper, we clarify the effective structure of the multi-head ensemble through insightful experiments and its trade-offs.

III. METHODOLOGY

The multi-head structure for deep metric learning, there are three structural design factors. First, the feature vector should be of a small dimension for making a concatenated ensemble feature. Second, the structure that shares low-level layers should save memory and not degrade performance over an ensemble of the independent models. Third, the feature vectors from each head should be diverse to make ensemble synergy. In this paper, we only consider the structural design of each head. In this section, we set up the design concepts according to the three design factors and propose a multi-head structure following the design concepts through
rigorous evaluation of variants on each factor. For the baseline network, we adopt the ResNet-50 [24] network, most popular backbone model. All experiments covered in this section are conducted on validating the CUB-200-2011 [25] dataset. The CUB-200-2011 training dataset is divided into training and validation sets equally. Values in the tables is the average value of the results obtained by 5 trials using different random seeds.

A. DIMENSION-REDUCED FEATURE VECTOR

**Design Concept:** A design concept of feature extraction block is presented to mitigate a redundancy by concatenation of ensemble vectors, thus improving the network efficiency. To this end, we set up multiple variants of the last feature block in the direction in order to reduce feature dimension and network complexity without considerable loss of performance. Then, we select one of the variants through a trade-off evaluation between complexity and performance for possible variants of the feature block.

**FIGURE 2.** Structure of variants for extracting a dimension-reduced feature vector. (a) is original structure of Res4_3, the last residual block of ResNet-50. (b),(c), and (d) are the modified versions to reduce feature dimensions. The apricot color blocks represent the ‘conv’ layers and the green one represents the ‘linear’ layer.

**TABLE 1.** Recall@K score on the validation set for finding an effective structure that extracting the dimension-reduced feature vector. The details of each variants are shown in Fig. 2.

| Methods        | Dim | Params | R@1 | R@2 | R@4 | R@8 |
|----------------|-----|--------|-----|-----|-----|-----|
| Original       | 2048| 23.51M | 78.94| 86.03| 91.67| 95.42|
| Adding Linear  | 512 | 24.56M | 77.24| 84.62| 90.34| 94.48|
| Penultimate    | 512 | 22.46M | 78.73| 86.14| 91.50| 95.47|
| Penultimate+   | 512 | 22.46M | 78.86| 86.05| 91.55| 95.48|

**Evaluation:** We present three representative variants that modified the last residual block of ResNet-50, as shown in Figure 2. Their performance is shown in Table 1. The trade-off evaluation results are as follows:

- The “original” gives 2048 dimensional feature vector and its time-complexity is $O(n^2 \times dim)$. Hence this one is not recommended in order to use feature vectors due to excessively large dimensions, even though it shows the best performance, as seen in Table 1.
- The variant “adding linear layer” variant is a method commonly used in other algorithms [3], [4], [8]. However, this method has a large performance drop compared to the original one, as shown in the second row of Table 1. Furthermore, adding a linear layer increases the network parameters, so it can be said that this method is not effective.
- The variants “Penultimate” and “Penultimate+” variants greatly reduce the complexity without considerable loss of performance. In addition, these variants reduced the parameters by removing the last convolutional layer. “Penultimate+” is chosen as our feature block because it performs slightly better performance than “Penultimate”.

**TABLE 2.** Train loss and Recall@K score on the validation set of the preliminary experiment for finding the effective structure of shared layers. To investigate the ensemble gain, the learning rate of the one head is set to two times of the others.

| Shared | Params | R@1 | R@2 | R@4 | R@8 |
|--------|--------|-----|-----|-----|-----|
| None   | 44.91M | 78.57| 86.19| 91.65| 95.50|
| Ensemble | | 79.43| 86.63| 91.96| 95.68|
| Until B1 | 44.69M | 78.61| 86.02| 91.64| 95.74|
| Ensemble | | 79.57| 86.69| 91.99| 95.58|
| Until B2 | 43.47M | 78.65| 86.00| 91.68| 95.71|
| Ensemble | | 79.68| 86.43| 91.84| 95.70|
| Until B3 | 36.37M | 78.23| 86.04| 91.62| 95.46|
| Ensemble | | 78.98| 86.27| 91.61| 95.46|

B. EFFECTIVE STRUCTURE OF SHARED BODY

**Design Concept:** The multi-head structure shares low-level layers, and the remaining high-level layers branch off into several heads. The design concept of the shared structure of low-level layers is provided to mitigate excessive memory usage, which is a side effect of the ensemble method. For this, we set up multiple variants with different shared layers. Since the goal of this design concept is to investigate the effective number of shared low-level layers, the number of heads is set to 2. Then, we investigate the trade-off between memory saving and performance variation. Finally, we choose a variant that shares the maximum number of layers without degrading performance for the two-head ensemble structure.

**Evaluation:** The layers of ResNet-50 network can be divided into five blocks as {conv1, Res1_x, Res2_x, Res3_x, Res4_x}. We have simply redefined it as {B1, B2, B3, B4} by combining conv1 into Res1_x. We have prepared four variants based on different shared blocks. To clearly observe the ensemble effect, the learning rate of one head was set to twice that of the other’s learning rate. The feature vector from each head is selected from Penultimate+. The performances of each head and ensemble are shown in Table 2. The trade-off evaluation results are as follows:
TABLE 3. The ensemble performance (Recall@1 on the validation set) for the combination of two heads. The increased value from the maximum among the two heads is shown in parentheses. The characteristics of each head are given in Table 4. The top performances of each cluster are marked by bold.

| Types      | A(η₁,s₂,t₃) | B(η₂,s₂,t₃) | C(η₂,s₂,t₃) | D(η₄,s₂,t₃) | E(η₁,s₁,t₂) | F(η₁,s₂,t₂) | G(η₁,s₂,t₂) | H(η₁,s₂,t₂) | I(η₁,s₁,t₂) |
|------------|-------------|-------------|-------------|-------------|-------------|-------------|-------------|-------------|-------------|
|            | 78.94 (+0.08) | 79.16 (+0.29) | 79.36 (+0.50) | 79.64 (+0.78) | 79.89 (-1.15) | 80.07 (+1.17) | 80.60 (+0.45) | 80.80 (+1.94) | 80.92 (+1.80) |
| B(η₀,s₂,t₃) | 78.92 (+0.33) | 79.41 (+0.82) | 80.09 (+1.50) |            | 80.28 (+0.25) | 80.03 (+1.13) | 80.69 (+0.54) | 80.63 (+2.04) | 80.70 (+1.59) |
| C(η₃,s₂,t₃) | -            | 79.67 (+1.67) | 79.25 (+1.24) |            | 79.88 (-1.15) | 80.49 (+1.59) | 80.78 (+0.63) | 80.59 (+2.33) | 80.76 (+1.65) |
| D(η₄,s₂,t₃) | -            | -            | 77.91 (+3.70) |            | 80.09 (+0.08) | 79.70 (+0.80) | 78.94 (-0.31) | 78.76 (+0.50) | 79.11 (-0.01) |
| E(η₁,s₁,t₂) | -            | -            | -            | -           | 80.63 (+0.60) | 80.80 (+0.77) | 81.41 (+1.26) | 81.82 (+1.78) | 81.87 (+1.84) |
| F(η₁,s₂,t₂) | -            | -            | -            | -           | -            | 79.94 (+0.83) | 80.17 (+0.02) | 80.84 (+1.34) | 80.51 (+1.60) |
| G(η₁,s₂,t₂) | -            | -            | -            | -           | 80.34 (+0.19) | 80.86 (+0.71) | 81.22 (+1.07) | -           | -           |
| H(η₁,s₂,t₂) | -            | -            | -            | -           | -            | -            | 78.46 (+0.19) | 79.04 (+0.07) | -           |
| I(η₁,s₁,t₂) | -            | -            | -            | -           | -            | -            | -            | -           | 79.29 (+0.18) |

- As the sharing portion becomes large, memory saving increases but the performance may decrease. However, interestingly, the ensemble performance does not always decrease with an increase in block sharing. In Until B1 and Until B2, the ensemble performances rather increase compared to without sharing (None).
- The structure of sharing Until B3 can reduce a large number of parameters, but the ensemble performance is degraded. This implies that the feature vector of each head could not be independent of the other due to much sharing.
- Although Until B1 and Until B2 have similar ensemble performance, the Until B2 is selected for our proposed structure design because Until B2 has the advantage of a large memory saving.

C. DIVERSE MULTI-HEADS

Design Concept: The third design concept of the proposed multi-head structure is to produce diverse feature vectors from multi-heads. We assume that the diverse designs of multi-heads can yield diverse feature vectors. Under this assumption, we investigate which structural combination is favorable for the ensemble via experimental evaluation. To this end, we prepare multiple head-variants based on the Until B2 shared structure and evaluate the ensemble performance for every combination of different head-features after training only the heads separately in fixing the remaining part. The head variants are designed by simply modifying the part of head layers in the hyper-parameter level (e.g., stride and learning rate changes). Based on the results, we employ a combination that provides the best ensemble performance for the proposed multi-head design.

Evaluation: We have prepared nine head-variants whose modification factors are summarized as follows:

- Learning rate change ($f_\eta$): The learning rate is considered as the first modification factor because it is a simple way to obtain an ensemble gain. The learning rate is a parameter that determines the step size of the update to compensate for the loss generated in the learning process. In the experiments, four learning rates $0.5 \times 1 \times 2 \times 4$, of the original learning rate ($5 \times 10^{-4}$) are considered.

- Stride change ($f_s$): The level of stride is the second modification factor. The stride is a design parameter of the convolution filter and determines the spatial filtering density of the input tensor. In the original ResNet-50, the two stride choices of the second convolution filter of Res4_1 are set to 1 and 2.

Higher layers pruning (Last layer): We leverage a technique that prunes higher layers. Ro and Choi [26] have shown that pruning a couple of higher layers shows better performance than the original one. Therefore, we consider the three variants whose Last layer are Res4_1 (Res4_2, Res4_3 pruned), Res4_2 (Res4_3 pruned), and Res4_3 (not pruned), respectively.

A total of nine head-variants are prepared by choosing the learning rate, the number of strides, and pruning variants, as shown in Table 4. The feature vectors are selected by passing over the last convolutional layer for each variant in the same manner as Penultimate+ in section III-A. The performance of single feature vector from each head variant, $f_\eta$ and $f_s$, mean learning rate factor and stride factor, respectively. 

The last layer represents the last remaining after higher layers pruning. The number of parameters for each head variant is counted without the shared layer.
In the top left of Table 3, among the combinations of learning rate variants, Type-B(\(\eta_5 \times 52 \times r_3\)) and Type-D(\(\eta_4 \times 52 \times r_3\)) ensemble has the highest performance. The performance of Type-D(\(\eta_4 \times 52 \times r_3\)) is the worst in Table 4, so its self-ensemble combination (Type-D,D) is also the worst, as shown in Table 3. However, Type-D is more helpful for an ensemble, and interestingly, it has a large variation from the default learning rate, compared to other learning rate variants. In the opposite case, Type-A(\(\eta_1 \times 52 \times r_3\)),B(\(\eta_5 \times 52 \times r_3\)), the combination with the smallest learning rates, has the lowest increase of ensemble performance. These results show that a successful ensemble of learning rate variants requires sufficient variations such as \(0.5 \times 4\times\).

Among the results in the top right of Table 3, Type-A(\(\eta_1 \times 52 \times r_3\)),I(\(\eta_1 \times r_1\)) shows the best ensemble performance with the different pruning (Res4_3 and Res4_1) and the different stride (2 and 1). From the comparison of Type-A(\(\eta_1 \times 52 \times r_3\)),E(\(\eta_1 \times 52 \times r_3\)), that it is shown that difference in stride does not contribute to making ensemble gains.

In the bottom right of Table 3, the combinations of Type-E(\(\eta_1 \times 52 \times r_3\)),H(\(\eta_1 \times 52 \times r_3\)) and Type-E(\(\eta_1 \times 52 \times r_3\)),I(\(\eta_1 \times 52 \times r_3\)) that have the different pruning (Res4_3 and Res4_1) provide the best and second-best performance. In the above cases and the case of Type-G(\(\eta_1 \times 52 \times r_3\)),I(\(\eta_1 \times 52 \times r_3\)), it is also confirmed that the different pruning from every other head is favorable to the ensemble.

Table 5 shows the representative results of the combination of triple heads. The triple-head structure using Type-E(\(\eta_1 \times 52 \times r_3\)), Type-D(\(\eta_4 \times 52 \times r_3\)), and Type-I(\(\eta_1 \times 52 \times r_3\)) shows the best ensemble performance, where each head adopts different structure from the others, i.e., Res4_3, Res4_2, and Res4_1 as Last layer, respectively. The best performance among the triple-head structures is slightly better than the best one among the double head structures. However, the triple-head structure requires larger complexity than the double head structure. Therefore, the structure of the double-head has been selected for our proposed structure in the consideration of trade-off between the memory savings and ensemble performance.

![Figure 3. Descriptions of the proposed HDhE structure and memory-saving version structure(Ms-HDhE).](image)

### Table 5. The ensemble performance (Recall@K on the validation set) for the combination of three heads. The characteristics of each head are given in Table 4.

| Combinations of triple-heads (Types) | R@1   | R@2   | R@4   | R@8   |
|-------------------------------------|-------|-------|-------|-------|
| I(\(\eta_1 \times 52 \times r_3\)) | 81.88 | 88.52 | 93.06 | 96.31 |
| I(\(\eta_1 \times 52 \times r_3\)) | 81.94 | 88.42 | 93.23 | 96.26 |
| I(\(\eta_1 \times 52 \times r_3\)) | 82.00 | 88.57 | 93.12 | 96.08 |

In summary, from our experimental investigation, we have found the following ensemble-favorable combinations for multi-head design. First, the large learning rate variation, such as \(0.5 \times 4\times\), is favorable to make the ensemble. Second, the Last layer difference between Res4_3 and Res4_1 is the most ensemble-favorable. Third, the difference in stride does not contribute to the ensemble, but the reduction of stride yields better performance in each head. Forth, the triple-head structure does not have large merit over the double-head structure in the performance comparison.

### D. PROPOSED MULTI-HEAD ENSEMBLE STRUCTURE

Based on the evaluation results, we propose a Heterogeneous Double-head Ensemble (HDhE) structure as shown in Figure 3-(a). Following the result in subsection A, the HDhE structure uses feature vectors that are reduced by one-quarter of the dimension of the original model. Following the result in subsection B, the shared body adopts Until B2, thus improving performance along with saving memory. Lastly, following the result in subsection C, the head structure is designed to make diverse feature vectors. The Last layer of the double-head is designed as Res4_3 and Res4_1, respectively, the learning rate is set to \(0.5 \times 4\times\), respectively, and the stride is set to 1 for both of two heads.

In addition, referring to the evaluation results, we can customize the multi-head structure in various views. For example in Figure 3-(b), a memory-saving version of HDhE (Ms-HDhE) can be suggested as follows. In Ms-HDhE, the shared body is enlarged Until B3 and the heads can be chosen by Res4_2 and Res4_1 with their learning rates of \(0.5 \times 4\times\), respectively, and the stride of 1 for both.

### IV. EXPERIMENTS

In this section, we provide experimental results for evaluating the proposed structure, especially in deep metric learning for image retrieval tasks. We first introduce the standard benchmark datasets of image retrieval tasks in section IV-A. Then we will give the implementation details of these experiments in section IV-B. The evaluation experiments contain four parts: the ablation studies on structural variants (section IV-C), the verification of suitability as a backbone.
TABLE 6. Recall@K score according to the variants for ablation study (effect of Penultimate+ and shared layers). Dim means the dimension of feature vector.

| Variants         | Dim  | Params | Shared layers | CUB-200 | Cars-196 |
|------------------|------|--------|---------------|---------|----------|
|                  |      |        |               | R@1    | R@2      | R@4    | R@8    | R@1    | R@2      | R@4    | R@8    |
| Baseline         | 4096 | 47.02M | None          | 68.75  | 80.00    | 87.47  | 92.39  | 87.39  | 93.25    | 96.34  | 97.95  |
| Baseline + Penultimate+ | 1024 | 44.92M | None          | 69.14  | 79.73    | 87.51  | 92.03  | 87.91  | 93.35    | 96.29  | 98.12  |
| HDhE-A           | 1024 | 25.34M | Until B3      | 71.74  | 81.70    | 88.37  | 93.15  | 89.53  | 94.40    | 96.90  | 98.41  |
| HDhE-B           | 1024 | 33.66M | Until B1      | 72.21  | 82.26    | 88.96  | 93.46  | 90.43  | 94.80    | 97.10  | 98.63  |
| Proposed HDhE    | 1024 | 32.44M | Until B2      | 72.87  | 82.29    | 89.01  | 93.59  | 90.62  | 94.99    | 97.17  | 98.56  |
| Single (ResNet-50)| 2048 | 22.46M | None          | 67.79  | 78.67    | 86.3   | 91.63  | 87.02  | 92.59    | 95.86  | 97.79  |
| Proposed Ms-HDhE | 1024 | 20.88M | Until B3      | 71.27  | 81.36    | 88.42  | 93.52  | 89.78  | 94.27    | 97.05  | 98.45  |

A. DATASETS

The proposed structure has been evaluated following four benchmark datasets for the image retrieval task.

**CUB-200-2011 (CUB-200) [25]** CUB-200 dataset has 11,788 images of 200 different species of birds. The first 100 species of bird images are used in training and the other 100 species used in testing. In particular, for the previous methodology section (section III), we divided the training set into two sets for validation.

**Stanford Cars (Cars-196) [27]** Cars196 dataset has 196 categories of various car images. The dataset, which consists of 16,185 images, is used for the first 98 classes in the training phase and the other 98 classes in the testing phase.

**Stanford Online Products (SOP) [28]** SOP dataset has 22,634 categories of online products with 120,053 images. 11,318 categories and 11,316 categories are used in training and testing, respectively.

**In-shop Clothes Retrieval (Inshop) [29]** Inshop dataset has 11,735 categories of clothing items with 54,642 images. 3,997 categories are used in training and testing. Remained 3,985 categories are used in query and gallery set.

B. IMPLEMENTATION DETAILS

In the paper, our experiments were implemented using the PyTorch [30] library. As the backbone network, ResNet-50 [24] pre-trained in ImageNet [31] was used. The input size was set to 224 × 224 and the batch size was set to 40. We utilized a data augmentation that flips input images horizontally in the probability of 0.5 in training. Basically, our HDhE was trained along with classifier by softmax (cross-entropy) loss. The learning rates were initially set to 0.001 for convolutional blocks (B1, B2, B3, and B4) and 0.01 for the classifier. For CUB-200 and Car-196, each HDhE has trained with the initial setting for the first 10 epochs and subsequently 10 epochs trained with learning rate 0.1 × of the initial one. For SOP and Inshop, each HDhE has trained for the first 30 epochs with the initial learning rate, and the learning rate was dropped to 0.1 × for the next 20 epochs. The optimizer used in this paper was stochastic gradient descent (SGD) with nesterov momentum [32]. The initial momentum rate and the weight decay were set to 0.9 and 5 × 10^{-4}, respectively. For the evaluation metric, the Recall@K metric [28] is employed.

C. ABLATION STUDY

In this section, we provide ablation studies to evaluate the extent to which design factors contribute to performance improvements. To this end, we have evaluated the ablation variant models from the proposed Heterogeneous Double-head Ensemble (HDhE) model. For the baseline model, we used the simple ensemble of two independent networks.

1) EFFECT OF PENULTIMATE+

To verify the effect of proposed Penultimate+, we evaluated the model replacing the Res4_3 block by Penultimate+ in the baseline model. In the first and second rows of Table 6, the proposed Penultimate+ not only reduces the dimension of feature vector by one-quarter over the baseline but also provides better ensemble performance in Recall@1.

2) EFFECT OF SHARED LAYERS

To evaluate the effect of our shared layers, we have compared two variants of shared layers. As shown in the third row of Table 6, HDhE-A (Until B3) saves parameters by a large margin, but the performance drop is relatively large. On the other hand, HDhE-B (Until B1) delivers performance comparable with the proposed HDhE (Until B2). However, it requires more parameters than the proposed. Therefore, the selection of Until B2 is valid in the fact that it saves parameters and improves performance as well. In addition, the proposed Memory-saving version of the Heterogeneous Double-head Ensemble (Ms-HDhE) can reduce the model complexity without considerable loss of performance.

3) EFFECT OF HEAD DESIGN

HDhE-C, HDhE-D, and HDhE-E use the same learning rate in two heads unlike HDhE. Additionally, HDhE-C and HDhE-D use the pruned head-2, where HDhE-C uses the same learning rate in two heads unlike HDhE.
TABLE 7. Recall@K score according to the variants for the ablation study (effect of head design). \(f_\eta\) and Last layer mean learning rate factor and the last layer of the head, respectively.

| Variants   | Params | \(f_\eta\) | Last layer | \(f_\eta\) | Last layer | CUB-200 | Cars-196 |
|------------|--------|------------|------------|------------|------------|---------|----------|
|            |        |            |            |            |            | R@1     | R@2      | R@4      | R@8      | R@1     | R@2      | R@4      | R@8      |
| HDhE-C     | 43.47M | 1x         | Res4_3     | 1x         | Res4_3     | 69.50    | 79.74    | 87.22    | 92.29    | 88.00    | 93.25    | 96.04    | 98.06    |
| HDhE-D     | 39.00M | 1x         | Res4_3     | 1x         | Res4_2     | 71.12    | 81.41    | 88.37    | 92.74    | 89.60    | 94.16    | 96.59    | 98.33    |
| HDhE-E     | 32.44M | 1x         | Res4_3     | 1x         | Res4_1     | 72.20    | 81.97    | 88.59    | 93.21    | 89.78    | 94.28    | 96.79    | 98.55    |
| Proposed HDhE | 32.44M | 0.5x       | Res4_3     | 4x         | Res4_1     | 72.87    | 82.29    | 89.01    | 93.59    | 90.62    | 94.99    | 97.17    | 98.56    |

Res4_3 and HDhE-D uses Res4_2 as Last layer, unlike HDhE using Res4_1. The stride of every head was set to 1. The evaluation results of the above variants are shown in Table 7. In the comparison among the above variants, HDhE-E delivers the best performance, whereas HDhE-D delivers better performance than HDhE-C, implying that the diversity in head structure by pruning only head-2 contributes to the increased performance. In particular, the larger the difference between two heads (Res4_3 and Res4_1), the larger is the performance improvement. In conclusion, the proposed HDhE adopting different learning rates in two heads gives further performance improvement compared to HDhE-E adopting the same learning rate in two heads. This implies that the diversity of learning rate also contributes to the increased performance. The above evaluation results show that the heterogeneous design of multi-heads can achieve a significant ensemble gain.

4) DISCUSSION ON USE OF TRIPLE OR MORE HEADS

Table 8 shows the evaluation results of the model using triple-heads with Until B2 shared structure. The triple-heads were designed as Res4_3, Res4_2, and Res4_1 of Last layer and 0.5x, 2x, and 4x of the default learning rate. The stride was set to 1 for every head. The triple-head structure provides better performance than the HDhE on the Car-196 but rather a slight decrease in the Recall@1 of the CUB-200. On the other hand, the parameters of the triple-head structure are 16M larger than those of HDhE. According to the trade-off between complexity and performance improvement, the triple-head or four-head structure does not have significant merit compared to the double-head (HDhE) structure.

D. GENERALITY FOR THE OTHER LOSS FUNCTIONS

In this section, to verify the generality of HDhE, we conducted an additional experiment by applying another type of loss function that uses triplet loss [33] along with softmax loss. The proposed HDhE structure trained by only softmax loss with a random sampling manner, whose setting is a simple and fast way to train the model. However, the triplet loss requires a different type of sampling to select samples of triplet pairs. Thus, we employed an identity sampling method and used the hard sample mining technique [34]. We then applied the triplet and softmax losses to the double heads simultaneously. We will call this method Tri-HDhE.

Interestingly, Table 9 shows that the performance of Tri-HDhE is enhanced significantly from that of the HDhE (softmax loss). Even though the performances of Cars-196 and Inshop datasets are already over 90% in training only by softmax loss, there are performance improvements of over 1% in Recall@1. From these results of this experiment, the generality of the proposed HDhE structure is verified in that it can be used as a backbone network for further performance improvement via other losses or for various applications.

E. COMPARISON TO STATE-OF-THE-ART METHODS

We compared the proposed HDhE family with existing state-of-the-art algorithms. Surprisingly, the Tri-HDhE model outperforms the state-of-the-art methods in the four image datasets.
TABLE 10. Comparison with state-of-the-art methods on CUB-200 and Cars-196 datasets (Recall@K score). † means using ensemble.

| Method       | Backbone       | CUB-200 |          |          |          |          |          |          |
|--------------|----------------|---------|----------|----------|----------|----------|----------|----------|
|              |                | R@1     | R@2      | R@4      | R@8      |          |          |          |
| HDC [1]      | Inception-v1†  | 53.6    | 65.7     | 77.0     | 85.6     | 73.7     | 83.2     | 89.5     | 93.8     |
| A-BIER [2]   | Inception-v1†  | 57.5    | 68.7     | 78.3     | 86.2     | 82.0     | 89.0     | 93.2     | 96.1     |
| ABE-8 [3]    | Inception-v1†  | 60.6    | 71.5     | 79.8     | 87.4     | 85.2     | 90.5     | 93.9     | 96.1     |
| MS [4]       | Inception-v1   | 65.7    | 77.0     | 86.4     | 91.2     | 84.1     | 90.4     | 94.0     | 96.5     |
| DREML [5]    | ResNet-18      | 63.9    | 75.0     | 83.1     | 89.7     | 86.0     | 91.7     | 95.0     | 97.2     |
| NormSoft [6] | ResNet-50      | 61.3    | 73.9     | 83.5     | 90.0     | 84.2     | 90.4     | 94.4     | 96.9     |
| Margin [7]   | ResNet-50      | 63.6    | 74.4     | 83.1     | 90.0     | 79.6     | 86.5     | 91.9     | 95.1     |
| SCHM [8]     | Inception-v1   | 66.2    | 76.3     | 84.1     | 90.1     | 83.6     | -        | -        | -        |
| SoftTriple [9]| BN Inception   | 65.4    | 76.4     | 84.5     | 90.4     | 84.5     | 90.7     | 94.5     | 96.9     |
| Proxy-Anchor [10]| BN Inception    | 68.4    | 79.2     | 86.8     | 91.6     | 86.1     | 91.7     | 95.0     | 97.3     |
| Sh-HDhE (Ours)| ResNet-50      | 68.2    | 79.2     | 86.9     | 92.7     | 86.9     | 92.8     | 96.1     | 98.0     |
| Ms-HDhE (Ours)| ResNet-50†    | 71.3    | 81.4     | 88.4     | 93.5     | 89.8     | 94.3     | 97.1     | 98.4     |
| HDhE (Ours)  | ResNet-50†    | 72.9    | 82.3     | 89.0     | 93.6     | 90.6     | 95.0     | 97.2     | 98.6     |
| Tri-HDhE (Ours)| ResNet-50† | 73.5    | 83.3     | 89.5     | 93.7     | 92.2     | 95.7     | 97.4     | 98.6     |

TABLE 11. Comparison with state-of-the-art methods on SOP and Inshop (Recall@K score). † means using ensemble.

| Method       | Backbone       | SOP      |          |          |          |          | Inshop   |          |          |          |
|--------------|----------------|----------|----------|----------|----------|----------|----------|----------|----------|----------|
|              |                | R@1     | R@10     | R@10²    | R@10³    | R@1      | R@10     | R@20     | R@30     |
| HDC [1]      | Inception-v1   | 69.5    | 84.4     | 82.8     | 97.7     | 62.1     | 84.9     | 89.0     | 91.2     |
| A-BIER [2]   | Inception-v1†  | 74.2    | 86.9     | 94.0     | 97.8     | 83.1     | 95.1     | 96.9     | 97.5     |
| ABE-8 [3]    | Inception-v1†  | 76.3    | 88.4     | 94.8     | 98.2     | 87.3     | 96.7     | 97.9     | 98.2     |
| MS [4]       | Inception-v1   | 78.2    | 90.5     | 96.0     | 98.7     | 89.7     | 97.9     | 98.5     | 98.8     |
| DREML [5]    | ResNet-18      | 72.7    | 86.2     | 93.8     | 98.0     | -        | -        | -        | -        |
| Margin [7]   | ResNet-50      | 79.5    | 91.5     | 96.7     | -        | 89.4     | 97.8     | 98.7     | 99.0     |
| NormSoft [6] | ResNet-50      | 77.6    | 89.1     | 94.7     | -        | 91.9     | 98.0     | 98.7     | 99.0     |
| SCHM [8]     | Inception-v1   | 78.3    | 90.3     | 95.9     | -        | -        | -        | -        | -        |
| SoftTriple [9]| BN Inception   | 77.1    | 90.8     | 96.2     | 98.7     | 91.5     | 98.1     | 98.8     | 99.0     |
| Proxy-Anchor [10]| BN Inception    | 84.0    | 93.5     | 97.1     | 98.9     | 91.2     | 98.0     | 98.6     | 98.8     |
| Ms-HDhE (Ours)| ResNet-50†    | 84.5    | 93.8     | 97.4     | 99.0     | 92.1     | 98.2     | 98.8     | 99.0     |
| HDhE (Ours)  | ResNet-50†    | 85.0    | 94.3     | 97.6     | 99.2     | 93.3     | 98.6     | 99.0     | 99.2     |

retrieval datasets, as shown in Table 10 and Table 11. In addition, not only HDhE but also Ms-HDhE outperforms the existing deep metric learning algorithms. Moreover, the results show that HDhE outperforms the existing ensemble algorithms (HDC [1], A-BIER [2], ABE-8 [3], and DREML [5]) with a large margin. Impressively, even the single-head version of HDhE (Sh-HDhE) having only the pruned head (Last layer is Res4_1) shows a competitive performance to the latest algorithms (SCHM [8], SoftTriple [9], and Proxy-Anchor [10]). Furthermore, the proposed HDhE gives a performance improvement of over 4% of Recall@1 (CUB-200) from the single-head version of HDhE (Sh-HDhE). This implies that our ensemble structure provides meaningful ensemble gain.

V. CONCLUSION
In this paper, we have provided the factors that influence an effective multi-head ensemble structure design, and we set up the design concepts through the design factors. Through an evaluation of various variants following the design concepts, we propose a heterogeneous double-head ensemble structure. The proposed structure has the following advantages:

- The dimension of feature vector from each head is reduced to one-quarter of the baseline model while improving performance.
- The shared body contributes to not only memory savings but also the performance enhancement.
- The heterogeneous design of heads yields a large amount of ensemble gain.
- The designed structure outperforms the existing ensemble structure and achieves state-of-the-art performance on the image retrieval benchmarks.
- The designed structure can be used as a backbone for various applications and other types of loss such as triplet loss.

REFERENCES
[1] Y. Yuan, K. Yang, and C. Zhang, “Hard-aware deeply cascaded embedding,” in Proc. IEEE Int. Conf. Comput. Vis. (ICCV), Oct. 2017, pp. 814–823.
[2] M. Opitz, G. Waltner, H. Possegger, and H. Bischof, “BIER—Boosting independent embeddings robustly,” in Proc. IEEE Int. Conf. Comput. Vis. (ICCV), Oct. 2017, pp. 5189–5198.
[3] W. Kim, B. Goyal, K. Chawla, J. Lee, and K. Kwon, “Attention-based ensemble for deep metric learning,” in Proc. Eur. Conf. Comput. Vis. (ECCV), Sep. 2018, pp. 736–751.
[4] X. Wang, X. Han, W. Huang, D. Dong, and M. R. Scott, “Multi-similarity loss with general pair weighting for deep metric learning,” in Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR), Jun. 2019, pp. 5022–5030.

[5] H. Xuan, R. Souvenir, and R. Pless, “Deep randomized ensembles for metric learning,” in Proc. Eur. Conf. Comput. Vis. (ECCV), 2018, pp. 723–734.

[6] A. Zhai and H.-Y. Wu, “Classification is a strong baseline for deep metric learning,” 2018, arXiv:1811.12649. [Online]. Available: http://arxiv.org/abs/1811.12649

[7] C.-Y. Wu, R. Mannathu, A. J. Smola, and P. Krahenbuhl, “Sampling matters in deep embedding learning,” in Proc. IEEE Int. Conf. Comput. Vis. (ICCV), Oct. 2017, pp. 2840–2848.

[8] Y. Suh, B. Han, W. Kim, and K. M. Lee, “Stochastic class-based hard example mining for deep metric learning,” in Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR), Jun. 2019, pp. 7251–7259.

[9] Q. Qian, L. Shang, B. Sun, J. Hu, T. Tacoma, H. Li, and R. Jin, “SoftTriple loss: Deep metric learning without triplet sampling,” in Proc. IEEE/CVF Int. Conf. Comput. Vis. (ICCV), Oct. 2019, pp. 6450–6458.

[10] Y. Sun, L. Zheng, Y. Yang, Q. Tian, and S. Wang, “Beyond part models: Person retrieval with refined part pooling (and a strong convolutional baseline),” in Proc. Eur. Conf. Comput. Vis. (ECCV), Sep. 2018, pp. 480–496.

[11] Y. Ro, J. Choi, D. U. Jo, B. Heo, J. Lim, and J. Y. Choi, “Backbone cannot be trained at once: Rolling back to pre-trained network for person re-identification,” in Proc. AAAI, 2019, pp. 8859–8867.

[12] W. Liu, Y.-M. Zhang, X. Li, Z. Yu, B. Dai, T. Zhao, and L. Song, “Deep hyperspherical learning,” in Proc. Adv. Neural Inf. Process. Syst., 2017, pp. 3950–3960.

[13] W. Liu, Y. Wen, Z. Yu, and M. Yang, “Large-margin softmax loss for convolutional neural networks,” in Proc. Int. Conf. Mach. Learn., 2016, pp. 507–516.

[14] H. Wang, Y. Wang, Z. Zhou, X. Ji, D. Gong, J. Zhou, Z. Li, and W. Liu, “CosFace: Large margin cosine loss for deep face recognition,” in Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit., Jun. 2018, pp. 5265–5274.

[15] L. Breiman, “Random forests,” Mach. Learn., vol. 45, no. 1, pp. 5–32, 2001.

[16] W. Bu, X. Zhu, and S. Gong, “Harmonious attention network for person re-identification,” in Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit., Jun. 2018, pp. 2285–2294.

[17] M. Opitz, H. Possenegg, and H. Bischof, “Efficient model averaging for deep neural networks,” in Proc. Asian Conf. Comput. Vis. Cham, Switzerland: Springer, 2016, pp. 205–220.

[18] G. Huang, Y. Li, G. Pleiss, Z. Liu, J. E. Hopcroft, and K. Q. Weinberger, “Snapshot ensembles: Train 1, get m for free,” 2017, arXiv:1704.00109. [Online]. Available: http://arxiv.org/abs/1704.00109

[19] X. Zhu and S. Gong, “Collaborative learning for deep neural networks,” in Proc. Adv. Neural Inf. Process. Syst., 2018, pp. 7517–7527.

[20] G. Song and W. Chai, “Collaborative learning for deep neural networks,” in Proc. Adv. Neural Inf. Process. Syst., 2018, pp. 1832–1841.

[21] B. Chen and W. Deng, “Hybrid-attention based decoupled metric learning for zero-shot image retrieval,” in Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR), Jun. 2019, pp. 2750–2759.

[22] W. Liu, Y. Yuan, X. Chen, J. Li, and X. Zhou, “Learning discriminative features with multiple granularities for person re-identification,” in Proc. ACM Multimedia Conf. (MM), 2018, pp. 274–282.

[23] K. He, X. Zhang, S. Ren, and J. Sun, “Deep residual learning for image recognition,” in Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR), Jun. 2016, pp. 770–778.

[24] C. Wah, S. Branson, P. Welinder, P. Perona, and S. Belongie, “The CaltechUCSD birds-200-2011 dataset,” California Inst. Technol., Pasadena, CA, USA, Tech. Rep. CNS-TR-2011-001, 2011.

[25] Y. Ro and J. Y. Choi, “Layer-wise pruning and auto-tuning of layer-wise learning rates in fine-tuning of deep networks,” 2020, arXiv:2002.06048. [Online]. Available: http://arxiv.org/abs/2002.06048

[26] Y. Ro, J. Y. Choi: HDhE for Deep Metric Learning