Compressed Self-Attention for Deep Metric Learning

Ziye Chen,[1] Mingming Gong,[2] Yanwu Xu,[3] Chaohui Wang,[4] Kun Zhang,[5] Bo Du[1]*

[1]School of Computer Science, Wuhan University
[2]School of Mathematics and Statistics, University of Melbourne
[3]Department of Biomedical Informatics, University of Pittsburgh
[4]Université Paris-Est, LIGM (UMR 8049), CNRS, ENPC, ESIEE Paris, UPEM, F-77455 Marne-la-Vallée, France
[5]Department of Philosophy, Carnegie Mellon University
{ziyechen, remoteking}@whu.edu.cn, mingming.gong@unimelb.edu.au,
yanwuxu@pitt.edu, chaohui.wang@u-pem.fr, kunz1@cmu.edu

Abstract

In this paper, we aim to enhance self-attention (SA) mechanism for deep metric learning in visual perception, by capturing richer contextual dependencies in visual data. To this end, we propose a novel module, named *compressed self-attention (CSA)*, which significantly reduces the computation and memory cost with a neglectable decrease in accuracy with respect to the original SA mechanism, thanks to the following two characteristics: i) it only needs to compute a small number of base attention maps for a small number of base feature vectors; and ii) the output at each spatial location can be simply obtained by an adaptive weighted average of the outputs calculated from the base attention maps. The high computational efficiency of CSA enables the application to high-resolution shallow layers in convolutional neural networks with little additional cost. In addition, CSA makes it practical to further partition the feature maps into groups along the channel dimension and compute attention maps for features in each group separately, thus increasing the diversity of long-range dependencies and accordingly boosting the accuracy. We evaluate the performance of CSA via extensive experiments on two metric learning tasks: person re-identification and local descriptor learning. Qualitative and quantitative comparisons with latest methods demonstrate the significance of CSA in this topic.

Introduction

Metric learning aims at finding appropriate similarity measures between pairs of data samples that preserve desired distance structures, which is of great importance for visual recognition. With the remarkable success of convolutional neural networks (Krizhevsky, Sutskever, and Hinton 2012), recent works have demonstrated promising results on learning semantic feature embeddings where similar examples are close to each other and dissimilar examples are far apart. In visual recognition, deep metric learning have been successfully applied to various tasks, such as person re-identification (Sun et al. 2018; Wang et al. 2018a), face recognition (Deng et al. 2019; Liu et al. 2018), and local descriptor learning (Mishchuk et al. 2017; Xu et al. 2019).

However, because of the local nature of convolution operators, the features in the shallow layers of CNNs are not able to capture long-range dependencies. While long-range dependencies can be captured in deep layers, optimization algorithms may have trouble in discovering appropriate values for the parameters that carefully coordinate multiple layers to capture those dependencies (Zhang et al. 2018). Increasing the convolutional kernel size can partially address this issue, but at the same time it introduces more parameters as well and necessitates more training data.

In this paper, we aim to enhance the *self-attention (SA)* mechanism for modeling rich long-range contextual dependencies in deep metric learning. Originally proposed in the field of natural language processing (NLP) (Vaswani et al. 2017), SA has shown promising results in vision tasks such as video analysis (Wang et al. 2018b), image segmentation (Fu et al. 2019), and image generation (Zhang et al. 2018; Gong et al. 2019). Because SA calculates response at a position as a weighted sum of the features at all positions, where the calculation of the weights only requires a small number of parameters, it can capture long-range contextual interactions with high statistical efficiency. However, SA requires computation of pairwise similarities among all the spatial positions, which leads to high computation and memory costs, and thus limits its usage to small inputs.

To address this problem, we propose a new SA module, called *compressed self-attention (CSA)*, which is able to boost metric learning accuracy with little additional computation or memory cost. Instead of computing an attention map for each spatial location, our CSA only requires computation of a small number of base attention maps for a small number of base feature vectors. The original input is then processed by these base attention maps to produce elementary outputs. Finally, the output at each location can be simply obtained by an adaptive weighted average of the elementary outputs. Due to the inherent redundancy of at-

*Corresponding author. This work was supported in part by the National Natural Science Foundation of China under Grants 61822113, the National Key R & D Program of China under Grant 2018YFA060550, the Natural Science Foundation of Hubei Province under Grants 2018CFA050 and the Science and Technology Major Project of Hubei Province (Next-Generation AI Technologies) under Grant 2019AEA170.

tention maps of all locations, our CSA method only incurs a slight decrease in accuracy compared to the original SA mechanism.

The computation and memory efficiency of CSA enables more flexible use of the SA mechanism. First, we can partition the feature maps into groups along the channel dimension and compute attention maps for each group independently. By doing so, CSA is able to select the features that optimally describe the images' specific meaning in any given context, which increases the diversity of long-range interactions. Second, it is more affordable to apply CSA to high-resolution shallow CNN layers, which can benefit more from modeling long-range dependencies. To demonstrate the effectiveness of our CSA module, we conduct extensive experiments on two well-known metric learning tasks in computer vision, namely person re-identification and local descriptor learning. Experimental results on the two tasks demonstrate the effectiveness and efficiency of the proposed CSA module.

Related Work

Deep Metric Learning

With the powerful feature representation capability of CNNs, deep metric learning has achieved great success. Currently, there are two ways to improve the performance of deep metric learning: one is to design a more effective loss function, and the other is to design a more reasonable network structure. For the former, Deng et al. (2019) proposed an additive angular margin loss to obtain highly discriminative features for face recognition. Xu et al. (2019) proposed a robust angular loss to tackle the incorrect correspondences for local descriptor learning. For the latter, Sun et al. (2018) proposed a part-based convolutional baseline (PCB) and a refined part pooling method (RPP) to obtain refined part-level features for person re-identification. In this paper, we focus on improving the network architecture by efficiently incorporating self-attention mechanism.

Self-Attention Mechanism

To the best of our knowledge, the work (Vaswani et al. 2017) was the first to propose the self-attention mechanism and applied it in the task of machine translation. Then self-attention has been increasingly applied in the computer vision field. Zhang et al. (2018) and Gong et al. (2019) applied self-attention to efficiently find global dependencies within internal representations for better image generation. Fu et al. (2019) proposed a dual self-attention mechanism for semantic segmentation, which includes a position attention module and a channel attention module to adaptively integrate local features with their global dependencies. Ramachandran et al. (2019) proposed a local self-attention mechanism to replace the convolution operation without a sharp increase in computation and memory cost. However, the locality makes it difficult to make full use of the global information. To the best of our knowledge, our work is the first to compress self-attention modules without loss of global context modeling.

Compression of CNNs

There are also plenty of works on compressing CNNs with low-rank approximation or sparse decomposition, which are most related to our compression of self-attention mechanism. Yu et al. (2017) proposed a unified framework integrating the low-rank and sparse decomposition of weight matrices with the feature map reconstructions to significantly reduce the parameters. Lin et al. (2018) proposed a holistic CNN compression framework (LRDKT) which includes a low-rank decomposition (LRD) scheme and a knowledge transfer (KT) based training scheme, to pursue a joint compression of convolutional layers and fully-connected layers. Different from these methods, we take advantage of the redundancy of the feature vectors in a feature map to compress attention maps, and the combination coefficients in our CSA are functions of the feature inputs in the corresponding location rather than fixed parameters.

Compressed Self-Attention (CSA) Module

In this section, we first briefly review the self-attention mechanism for the purpose of completeness. Then, we introduce the proposed CSA module in detail, including its mathematical formulation and implementation. Finally, we present the way to perform CSA for individual groups of feature maps together with the application in deep metric learning.

Review of Self-Attention

As shown in Figure 1, let us consider the input as a feature map $\mathbf{I} \in \mathbb{R}^{C \times H \times W}$ in a 2D CNN layer, where C, H and W represent the channels, height, and width of the input feature map, respectively. Self-attention generates an output $\mathbf{O} \in \mathbb{R}^{C \times H \times W}$ that has the same size as the input. The feature vector at the i-th output position $O_i \in \mathbb{R}^C$ is calculated as

$$O_i = \sum_{j=1}^{HW} f(I_i, I_j) g(I_j), \tag{1}$$

where $I_i \in \mathbb{R}^C$ denotes the feature vector at the i-th input position (same for I_j), $g(\cdot)$ is a feature extraction function, and $f(\cdot, \cdot)$ computes a scalar representing the pairwise relationship between the feature vectors at i-th and j-th input locations, and generates attention maps $\mathbf{A} \in \mathbb{R}^{HW \times H \times W}$. A commonly used pairwise function is

$$f(I_i, I_j) = \frac{e^{I_i^\intercal W_\theta^\intercal W_\phi I_j}}{\sum_{j'=1}^{HW} e^{I_i^\intercal W_\theta^\intercal W_\phi I_{j'}}}, \tag{2}$$

where $W_\theta \in \mathbb{R}^{C' \times C}$ and $W_\phi \in \mathbb{R}^{C' \times C}$ map the original input $\mathbf{I}$ to the feature embeddings $\mathbf{Q} \in \mathbb{R}^{C' \times H \times W}$ and $\mathbf{K} \in \mathbb{R}^{C' \times H \times W}$. For simplicity, we only consider g in the form of a linear transformation: $g(I_i) = W_g I_i$, where $W_g \in \mathbb{R}^{C \times C}$ is a learnable weight matrix which maps the original input $\mathbf{I}$ to a new feature map $\mathbf{V} \in \mathbb{R}^{C \times H \times W}$. The feature embeddings $\mathbf{Q}$ and $\mathbf{K}$ and the feature map $\mathbf{V}$ can be implemented as 1×1 convolutions in space, as shown in Figure 1.

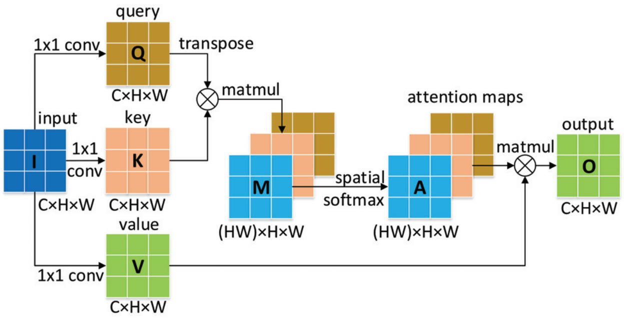


Figure 1: The illustration of self-attention.

As opposed to convolutions that only operate in local regions, self-attention can model long-range dependencies by calculating response at a position as a weighted sum of the features at all positions based on pairwise relationship. Its connections to the non-local filtering operations in image processing have been analyzed in (Wang et al. 2018b). Compared to fully-connected layers, the parameterization is remarkably efficient: the number of parameters does not grow with spatial resolution of the input. However, an apparent disadvantage of self-attention is that the computation and memory cost grows quickly with the input size. As shown in Figure 1, SA requires $H \times W$ attention maps, each of which is of size $H \times W$, limiting SA to small inputs. To apply SA to shallow CNN layers, which lack long-range dependencies, we propose the compressed self-attention module.

Compressed Self-Attention (CSA)

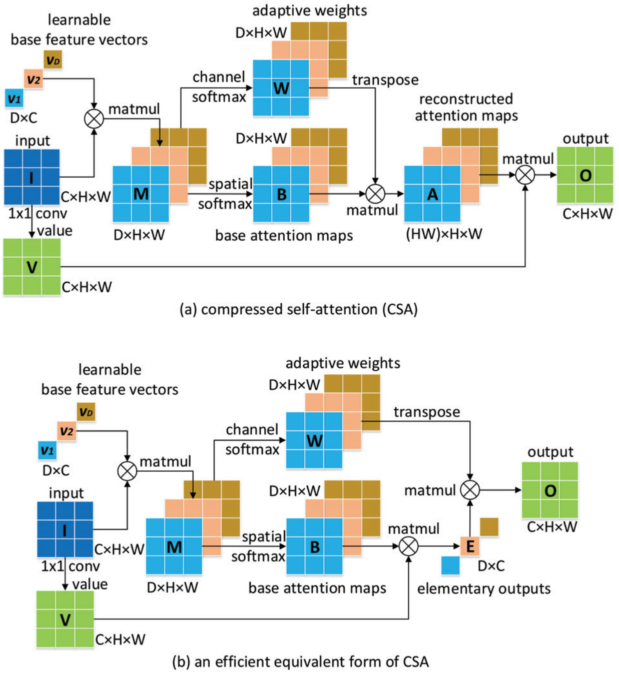
We aim to compress the attention maps to obtain both computational and memory efficiency. The motivation of our method is based on the observation that the feature vectors in a feature map have significant redundancy and usually form a small number of clusters. This property was extensively studied for image segmentation and grouping (Achanta et al. 2012). We can use a small number of learnable weight vectors $\mathbf{v}_1, \ldots, \mathbf{v}_D$ to learn these clusters, and use them as bridges to establish the relationship between any two feature vectors in the feature map instead of pairwise measurement. We call these weight vectors *base feature vectors*. Thus, instead of computing an attention map for each I_i, we first compute attention maps for a small number of base feature vectors $\mathbf{v}_1, \ldots, \mathbf{v}_D$ by

$$f_b(\mathbf{v}_k, I_j) = \frac{e^{\mathbf{v}_k^\intercal I_j}}{\sum_{j'=1}^{HW} e^{\mathbf{v}_k^\intercal I_{j'}}}, \tag{3}$$

where $f_b(\mathbf{v}_k, I_j)$ represents the value at the j-th location of the k-th attention map B_k corresponding to $\mathbf{v}_k$, as shown in Figure 2 (a).

After obtaining the base attention maps $\mathbf{B}$, the i-th attention map A_i for I_i can be reconstructed from $\mathbf{B}$ according to the relationship between I_i and the base feature vectors $\mathbf{v}_1, \ldots, \mathbf{v}_D$ as follows:

$$f_a(I_i, I_j) = \sum_{k=1}^{D} f_b(\mathbf{v}_k, I_j) w(\mathbf{v}_k, I_i), \tag{4}$$



(a) compressed self-attention (CSA)

(b) an efficient equivalent form of CSA

Figure 2: The illustration of compressed self-attention.

where $w(\mathbf{v}_k, I_i) = \frac{e^{\mathbf{v}_k^\intercal I_i}}{\sum_{k'=1}^{D} e^{\mathbf{v}_{k'}^\intercal I_i}}$ is an adaptive weight representing the relationship between I_i and $\mathbf{v}_k$.

As shown in Figure 2 (a), the base attention maps $\mathbf{B} \in \mathbb{R}^{D \times H \times W}$ and the adaptive weights $\mathbf{W} \in \mathbb{R}^{D \times H \times W}$ can be efficiently computed by first applying 1×1 convolutions to the original input $\mathbf{I}$ and then performing softmax operations on the spatial and channel dimensions respectively. The original attention maps $\mathbf{A}$ can be easily reconstructed by performing matrix multiplication between $\mathbf{B}$ and $\mathbf{W}$. Then we can obtain the final output $\mathbf{O}$ by performing matrix multiplication between the attention maps $\mathbf{A}$ and the new feature map $\mathbf{V}$.

In fact, we can exchange the execution orders of the last two matrix multiplications to implement CSA efficiently, which is shown in Figure 2 (b). In detail, we can plug (4) into (1) and obtain the final output of CSA as follows:

$$\begin{aligned} O_i &= \sum_{j=1}^{HW} f_a(I_i, I_j) g(I_j) \\ &= \sum_{j=1}^{HW} \sum_{k=1}^{D} f_b(\mathbf{v}_k, I_j) w(\mathbf{v}_k, I_i) g(I_j) \\ &= \sum_{k=1}^{D} w(\mathbf{v}_k, I_i) E_k, \end{aligned} \tag{5}$$

where $E_k = \sum_{j=1}^{HW} f_b(\mathbf{v}_k, I_j) g(I_j)$ is the k-th elementary output that is obtained by applying the k-th base attention map B_k to the new feature map $\mathbf{V}$. Then the final output $\mathbf{O}$ can be obtained by a weighted average of the D elementary outputs.

As shown in Figure 2 (b), we can first apply the base attention maps $\mathbf{B}$ to the new feature map $\mathbf{V}$ to obtain the elementary outputs $\mathbf{E} \in \mathbb{R}^{C\times D}$, which selects the most relevant spatial feature vectors for each cluster. Then we can obtain the final output $\mathbf{O}$ by applying the adaptive weights $\mathbf{W}$ to the elementary outputs $\mathbf{E}$, which selects the most relevant cluster for each spatial feature vector. The calculations and the size of the base attention maps $\mathbf{B}$ both become $\frac{D}{HW}$ of the original, which greatly reduces the computation and memory costs.

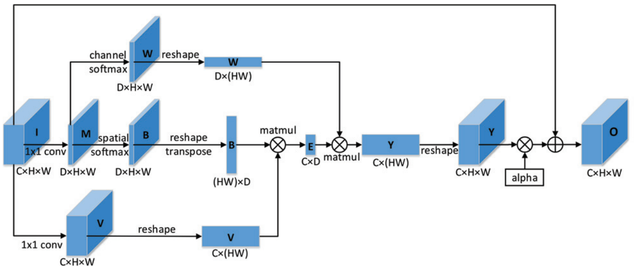


Figure 3: The concrete implementation of CSA module.

Implementation of CSA module The implementation of CSA module is shown in Figure 3, where the algorithmic scheme is detailed as follows. Given the input $\mathbf{I} \in \mathbb{R}^{C\times H\times W}$: (i) it is fed into two 1×1 convolution layers to generate two new feature maps $\mathbf{M} \in \mathbb{R}^{D\times H\times W}$ and $\mathbf{V} \in \mathbb{R}^{C\times H\times W}$, respectively; (ii) two softmax operations are performed on the spatial and channel dimensions of $\mathbf{M}$ to generate the base attention maps $\mathbf{B} \in \mathbb{R}^{D\times H\times W}$ and the adaptive weights $\mathbf{W} \in \mathbb{R}^{D\times H\times W}$, respectively; (iii) $\mathbf{B}$ is reshaped and transposed to $\mathbb{R}^{HW\times D}$, $\mathbf{V}$ and $\mathbf{W}$ are reshaped to $\mathbb{R}^{C\times HW}$ and $\mathbb{R}^{D\times HW}$, respectively; (iv) a matrix multiplication is performed between $\mathbf{V}$ and $\mathbf{B}$ to produce the elementary outputs $\mathbf{E} \in \mathbb{R}^{C\times D}$ as follows:

$$E_k = \sum_{j=1}^{HW} V_j \cdot \frac{e^{M_{kj}}}{\sum_{j'=1}^{HW} e^{M_{kj'}}}; \tag{6}$$

(v) a matrix multiplication is performed between $\mathbf{E}$ and $\mathbf{W}$ to produce the feature map $\mathbf{Y} \in \mathbb{R}^{C\times HW}$; (vi) $\mathbf{Y}$ is reshaped to $\mathbb{R}^{C\times H\times W}$, which is then multiplied by a learnable or fixed parameter α; (vii) an element-wise summation operation is performed between the result obtained in the previous step and the input $\mathbf{I}$ to obtain the final output $\mathbf{O} \in \mathbb{R}^{C\times H\times W}$ as follows:

$$O_i = \alpha \sum_{k=1}^{D} E_k \cdot \frac{e^{M_{ki}}}{\sum_{k'=1}^{D} e^{M_{k'i}}} + I_i, \tag{7}$$

In order to further boost the performance of the proposed CSA module, we partition the feature maps into several groups along the channel dimension and perform CSA within each group independently. By doing so, CSA is able to select the features that optimally describe specific meaning of an image in any given context, which increases the diversity of long-range interactions. The grouped CSA is very similar to the original CSA, except the channel grouping, and we will not elaborate on this in more detail.

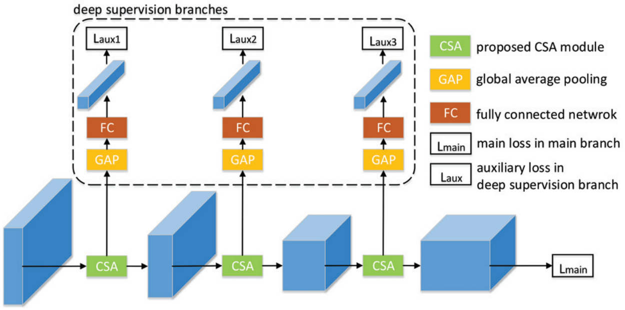


Figure 4: The general framework of applying our CSA module in deep metric learning.

Application in Deep Metric Learning Our CSA module can be easily combined with existing methods to make use of the attention mechanism, and we propose a general framework to apply our CSA module in deep metric learning. As shown in Figure 4, the framework includes a backbone network, CSA modules, a main branch, and deep supervision branches. We add the CSA modules to the representative algorithms in specific metric learning tasks to boost the original performance, without changing the backbone networks and the main branches of the original algorithms.

We insert the CSA module between the two adjacent stages of the backbone to utilize both the local detailed information in the shallow feature maps and the rich semantic information in the deep feature maps. In order to learn more meaningful base feature vectors and base attention maps, we apply deep supervision to each CSA module. The deep supervision branch includes a global average pooling layer and fully connected layers. The final loss function is as follows:

$$L_{total} = L_{main} + \lambda \sum_{i=1}^{N} L_{aux_i} \tag{8}$$

where L_{main} represents the main loss of the main branch, and L_{aux_i} represents the ith auxiliary loss in the ith deep supervision branch. The main loss L_{main} is the same as the original method. The auxiliary loss L_{aux} can be the cross entropy loss. λ is a balance factor between L_{main} and L_{aux}. N is the number of CSA modules applied to the backbone.

Experiments

We evaluate the performance of CSA via extensive experiments on two metric learning tasks: person re-identification and local descriptor learning. We apply the proposed framework with CSA to the representative methods (Sun et al. 2018; Mishchuk et al. 2017; Xu et al. 2019) in the two tasks to boost their performance. Qualitative and quantitative comparisons demonstrate that the inclusion of CSA within those methods leads to significant improvement in performance.

Models	Market-1501				DukeMTMC-reID				CUHK03-NP			
	R-1	R-5	R-10	mAP	R-1	R-5	R-10	mAP	R-1	R-5	R-10	mAP
HA-CNN (Li, Zhu, and Gong 2018)	91.2	—	—	75.7	80.5	—	—	63.8	41.7	—	—	38.6
DuATM (Si et al. 2018)	91.4	97.1	—	76.6	81.8	90.1	—	64.6	—	—	—	—
Mancs (Wang et al. 2018a)	93.1	—	—	82.3	84.9	—	—	71.8	65.5	—	—	60.5
PCB (Sun et al. 2018)	92.3	97.2	98.2	77.4	81.7	89.7	91.9	66.1	59.7	77.7	85.2	53.2
PCB + SA	93.8	97.7	98.5	82.2	85.3	92.7	**94.8**	73.4	65.9	82.3	88.4	62.7
PCB + CSA	93.7	**98.2**	**98.8**	82.3	**85.5**	92.8	94.5	**73.5**	67.2	83.9	88.9	63.7
PCB-RPP (Sun et al. 2018)	93.8	97.5	98.5	81.6	83.3	90.5	92.5	69.2	62.8	79.8	86.8	56.7
PCB-RPP + SA	93.3	97.5	98.7	83.2	84.9	92.8	94.7	72.9	66.1	**84.0**	89.0	64.6
PCB-RPP + CSA	**93.9**	97.8	**98.8**	**83.5**	85.4	**93.1**	94.5	73.1	**67.4**	83.6	**89.1**	**65.0**

Table 1: Comparison of the models with and without the proposed CSA. SA means the original self-attention.

In the following subsections, we will introduce the datasets, experimental settings, and results in more detail on the two tasks. In addition, we perform a set of ablation studies on the person re-identification task to show the sensitivity of our CSA module with respect to those key parameters / settings: the number of base attention maps, the number of groups, CSA on different stages, and deep supervision.

Person Re-Identification

For person re-identification, we consider two representative algorithms, **PCB** (Sun et al. 2018) and **PCB-RPP** (Sun et al. 2018), as the base models, and conduct the experiments on **Market-1501** (Zheng et al. 2015), **DukeMTMC-reID** (Ristani et al. 2016; Zheng, Zheng, and Yang 2017), and **CUHK03-NP** (Zhong et al. 2017; Li et al. 2014) datasets.

Datasets Market-1501 has 12,936 images of 751 identities for training, and has 3,368 query images and 19,732 gallery images of 750 other identities for testing. DukeMTMC-ReID has 16,522 images of 702 identities for training, and has 2,228 query images and 17,661 gallery images of 702 other identities for testing. CUHK03-NP is re-formulated from the old CUHK03 dataset (Li et al. 2014) with the new training/testing protocol proposed in (Zhong et al. 2017). It contains 7,368 images of 767 identities for training and 5,328 images of 700 other identities for testing. CUHK03-NP offers both hand-labeled and DPM-detected bounding boxes, and we use the latter.

The evaluation protocol proposed in (Zheng et al. 2015) is used for Market-1501 and DukeMTMC-ReID. The new protocol from (Zhong et al. 2017) is employed for CUHK03-NP. The Cumulative Matching Characteristic (CMC) for rank-1, rank-5, rank-10 and the mean average precision (mAP) are measured. All the following results are evaluated under the single-query mode. The results are reported without re-ranking (Zhong et al. 2017).

Implementation Details For PCB and PCB-RPP, we follow the training settings in (Sun et al. 2018). The input images are resized to 384×128 and augmented with random horizontal flipping. We use ResNet50 with the pre-trained weights from ImageNet as the backbone. Optimization is done by SGD with momentum of 0.9 and weight decay of 0.0001. The batch size is set to 64. We train the model for 100 epochs. The base learning rate is initialized at 0.1 and multiplied by 0.1 after every 40 epochs. The learning rate for the backbone is set to 0.1 times the base learning rate. For PCB-RPP, we apply first 40 epochs for training PCB, and append another 60 epochs when employing RPP for boosting.

When applying the proposed CSA for further boosting, we insert the CSA module between the two adjacent stages of ResNet50 and apply deep supervision to each CSA module. Thus, the number of CSA modules N is 3. For the hyper-parameters of CSA, we set the number of base attention maps in each group to 32 and the number of groups to 2. The main loss L_{main} is the same as the original method, and L_{aux} is the cross entropy loss. The balance factor λ is set to 1.0. When comparing SA with CSA, we just replace CSA with SA in the framework of Figure 4 without changing other settings, such as the deep supervision branches and the forms of L_{main} and L_{aux}. The channel grouping is not included in SA, which is too expansive for SA.

Main Results As shown in Table 1, the inclusion of CSA leads to performance improvement for all those original models on Market-1501, DukeMTMC-ReID and CUHK03-NP regarding both rank-1 accuracy and mAP. For the basic model of PCB, it shows an improvement of 1.4% for R-1 and 4.9% for mAP on Market-1501, 3.8% for R-1 and 7.4% for mAP on DukeMTMC-ReID, 7.5% for R-1 and 10.5% for mAP on CUHK03-NP. This indicates that CSA can model rich long-range contextual dependencies and increase the discriminative ability of the features.

We also compare the proposed CSA with the original SA. As shown in Table 1, the performance of CSA is slightly better than that of SA, despite of the compression of attention maps. This is because the channel grouping helps to diversify the long-range dependencies, the positive affect of which is higher than the performance degradation caused by compression. In addition, it is worth noting that SA consumes a large amount of time and memory, especially for the shallow feature maps, due to which it is difficult to apply channel grouping strategy to SA.

Influence of the Number of Base Attention Maps We study the influence of the number of base attention maps in each group on the performance when fixing the number of groups to 2. As shown in Table 2, when we increase the number of base attention maps from 8 to 32, the performance is improved. This is because the more base attention maps, the

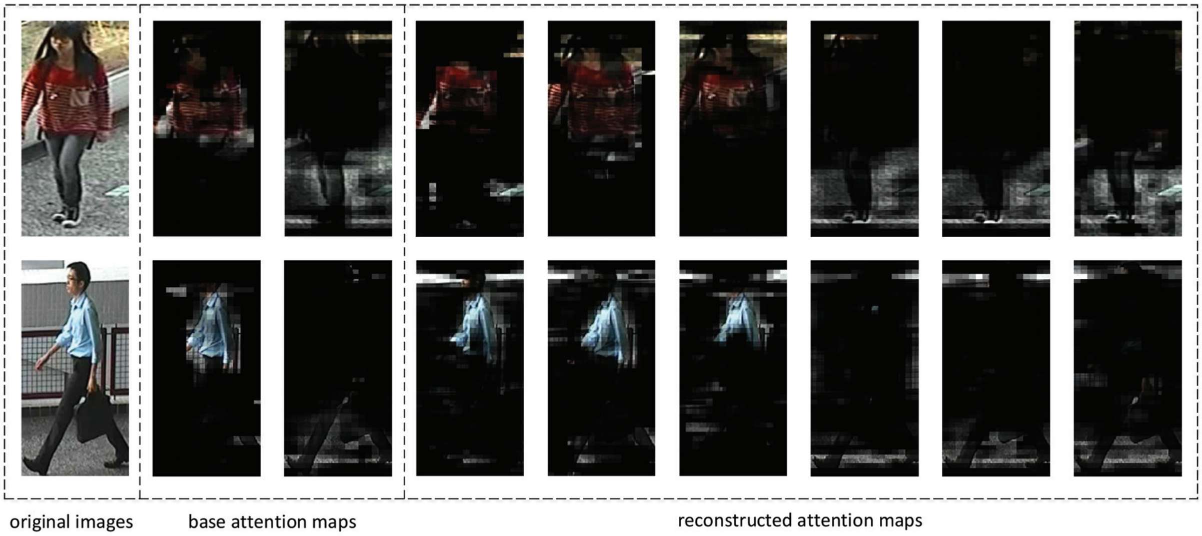


Figure 5: Visualization of the base attention maps and the reconstructed attention maps.

less compression of the original attention maps, and CSA has a finer description of the images' specific meaning in the given context. However, when we further increase the number of base attention maps from 32 to 64, the performance slightly decreases, which is very likely to be caused by overfitting.

Models	Map Num	Market-1501		DukeMTMC-reID	
		R-1	mAP	R-1	mAP
PCB + CSA	8	93.6	82.1	85.0	73.2
PCB + CSA	16	93.6	82.2	**85.5**	73.4
PCB + CSA	32	93.7	**82.3**	**85.5**	**73.5**
PCB + CSA	64	**93.9**	82.2	85.3	73.3

Table 2: Comparison of the models with different numbers of base attention maps in each group of CSA.

Influence of the Number of Groups We study the influence of the number of groups on the performance when fixing the number of base attention maps in each group to 32. As shown in Table 3, when we increase the number of groups from 1 to 2, the performance is improved. This suggests that the channel grouping helps to increases the diversity of long-range interactions and makes the features more discriminative. However, the performance slightly decreases when further increasing the number of groups from 4 to 8, very possibly due to the fact that the number of channels in each group is not adequate enough to express specific semantics in the datasets.

Influence of CSA on Different Stages We study the influence of placing the CSA modules on different stages of the backbone on the performance. As shown in Table 4, when we place the CSA modules simultaneously on the 1st, 2nd, 3rd stages of the backbone, the performance is best. The performance of placing CSA merely on the shallow feature maps (i.e., the 1st stage) is better than that of placing CSA on the deep feature maps (i.e., the 2nd, 3rd stages). This is because the shallow feature maps contain more local detailed information and lack long-range dependencies, which benefits more from self-attention mechanism.

Models	Group Num	Market-1501		DukeMTMC-reID	
		R-1	mAP	R-1	mAP
PCB + CSA	1	93.6	81.9	85.2	73.1
PCB + CSA	2	93.7	**82.3**	85.5	**73.5**
PCB + CSA	4	**93.9**	82.1	**85.6**	73.4
PCB + CSA	8	93.8	**82.3**	85.5	73.3

Table 3: Comparison of the models with different numbers of groups in CSA.

Models	Attn Stages	Market-1501		DukeMTMC-reID	
		R-1	mAP	R-1	mAP
PCB	no	92.3	77.4	81.7	66.1
PCB + CSA	1st stage	93.5	81.6	84.6	72.8
PCB + CSA	2nd,3rd stages	93.3	81.2	84.7	72.6
PCB + CSA	1st,2nd,3rd stages	**93.7**	**82.3**	**85.5**	**73.5**

Table 4: Comparison of the models with CSA on different stages of the backbone.

Influence of Deep Supervision We study the influence of deep supervision on the CSA module on the performance. As shown in Table 5, the performance of the CSA module with deep supervision is always better than that without deep supervision. This is because deep supervision provides a stronger supervision signal, which helps the CSA module learn more meaningful base feature vectors and base attention maps.

Comparison of Speed and Memory Cost We also compare the speed and memory cost of the models with and without SA or CSA. For CSA, we also consider the impact of different numbers of groups. As shown in Table 6, the

Train	Notredame	Yosemite	Liberty	Yosemite	Liberty	Notredame	Mean
Test	Liberty		Notredame		Yosemite		
HardNet (Mishchuk et al. 2017)	1.49	2.51	0.53	0.78	1.96	1.84	1.51
HardNet + SA	1.23	1.98	0.40	0.71	**1.37**	1.06	1.13
HardNet + CSA	1.23	1.92	0.42	0.73	1.39	1.09	1.13
RALNet (Xu et al. 2019)	1.30	2.39	**0.37**	0.67	1.52	1.31	1.26
RALNet + SA	1.23	**1.84**	0.38	**0.66**	1.40	1.05	**1.09**
RALNet + CSA	**1.16**	1.95	0.40	0.71	1.38	**1.02**	1.10

Table 7: Comparison of the models with and without the proposed CSA on patch correspondence verification performance on Brown dataset. We report false positive rate at true positive rate equal to 95% (FPR95).

Models	DS	Market-1501		DukeMTMC-reID	
		R-1	mAP	R-1	mAP
PCB + CSA	no	93.5	81.3	85.3	72.8
PCB-RPP + CSA	no	93.6	82.7	84.9	72.7
PCB + CSA	yes	93.7	82.3	**85.5**	**73.5**
PCB-RPP + CSA	yes	**93.9**	**83.5**	85.4	73.1

Table 5: Comparison of the models with and without deep supervision.

method with SA has high computation and memory costs. The memory cost is 242% of the original and FPS is 59% of the original. The methods with CSA consume less time and memory. When the number of groups is 1, the increase of memory is 11% and the decrease of FPS is 5%, which demonstrates the benefit of CSA. Even when we increase the number of groups to 8, the increase of memory is 26% and the decrease of FPS is 17%, which is still tolerable.

Models	Group Num	Memory (MB/image)	FPS
PCB	—	125.64	380.24
PCB + SA	1	304.33	224.30
PCB + CSA	1	139.60	360.90
PCB + CSA	2	142.27	350.36
PCB + CSA	4	149.02	335.66
PCB + CSA	8	158.02	314.80

Table 6: Comparison of the speed and memory cost of the models with and without SA or CSA.

Visualization of Attention Maps In order to prove the effectiveness of the CSA module, we select two base attention maps and six reconstructed attention maps for each image to display. As shown in Figure 5, for each image, the two base attention maps focus on the parts with different semantics and each reconstructed attention map is similar to one of the base attention maps. This indicates that the total attention maps are redundant, and we can compress them into a small number of base attention maps, with a few base feature vectors which can be viewed as the clusters of the feature vectors in a feature map.

Local Descriptor Learning

For local descriptor learning, we consider two representative algorithms, **HardNet** (Mishchuk et al. 2017) and **RALNet** (Xu et al. 2019), as the basic models, and conduct the experiments on **Brown** (Brown and Lowe 2007) and **Hpatches** (Balntas et al. 2017) datasets.

Datasets Brown dataset (Brown and Lowe 2007) consists of three subsets: Liberty, Notredame, and Yosemite with about 400k patches in each subset. For evaluations on Brown dataset, the models are trained on one subset and tested on the other two. We follow the standard evaluation protocol in (Brown and Lowe 2007) which uses the provided 100K pairs and report the false positive rate at the recall of 95%.

Hpatches dataset (Balntas et al. 2017) consists of 116 sequences of 6 images. It includes three evaluation tasks, patch verification, patch retrieval, and image matching, which are implemented with three levels of difficulty. Following (Xu et al. 2019), we report the average performance of all different factors. We use the models trained on Liberty subset of Brown dataset and test their generalization performance on Hpatches dataset, which is a common practice.

Implementation Details For HardNet and RALNet, we follow the same implementation details as in (Mishchuk et al. 2017) and (Xu et al. 2019). The input images are resized to 32×32, and per-batch normalized. We apply data augmentation by random flipping and 90° rotation. We use L2Net (Tian, Fan, and Wu 2017) as the main network. Optimization is done by SGD with momentum of 0.9 and weight decay of 0.0001. We use the same positive pairs and negative pairs sampling strategy and extract 5000K pairs. We train the model for 10 epochs with learning rate initialized at 10 and linearly decayed to 0. When training the network with CSA, we place the CSA module on the 3rd layer of L2Net without deep supervision and channel grouping. The main loss L_{main} is the same as the original method.

Task	Verification	Matching	Retrieval
HardNet (Mishchuk et al. 2017)	87.12	51.37	69.74
HardNet + SA	87.76	**53.17**	71.45
HardNet + CSA	87.67	52.66	71.14
RALNet (Xu et al. 2019)	87.43	53.16	69.70
RALNet + SA	87.85	52.89	**71.75**
RALNet + CSA	**88.11**	52.62	71.72

Table 8: Comparison of the models with and without CSA on verification, matching and retrieval results on HPatches dataset. All the descriptors are trained on Liberty subset of Brown dataset.

Main Results As shown in Table 7 and Table 8, the inclusion of the proposed CSA leads to a large performance improvement for all the models, in the experiments conducted on both Brown dataset and Hpathces dataset. It is interesting that the improvement of HardNet is larger than that of RALNet. This is because RALNet has a more reasonable loss function and thus leads to a better feature representation. This demonstrates that CSA is still able to greatly improve the feature representation even without an elaborate loss function. When compared to the original SA, the performance of our CSA decreases slightly in some cases. This is because the image patch data is relatively simple and can be represented by feature maps with fewer channels, which cannot further benefit from channel grouping.

Conclusion

In this paper, we aim to enhance self-attention (SA) mechanism for deep metric learning in visual perception, by capturing richer contextual dependencies in visual data. We propose a novel mechanism, named *compressed self-attention (CSA)*, which significantly reduces the computation and memory cost with a neglectable decrease in accuracy with respect to the original SA mechanism. Experimental results demonstrate that the proposed method achieves competitive results compared to those state-of-the-art methods at significantly lower computational cost.

References

Achanta, R.; Shaji, A.; Smith, K.; Lucchi, A.; Fua, P.; and Süsstrunk, S. 2012. Slic superpixels compared to state-of-the-art superpixel methods. *IEEE transactions on pattern analysis and machine intelligence* 34(11):2274–2282.

Balntas, V.; Lenc, K.; Vedaldi, A.; and Mikolajczyk, K. 2017. Hpatches: A benchmark and evaluation of handcrafted and learned local descriptors. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 5173–5182.

Brown, M., and Lowe, D. G. 2007. Automatic panoramic image stitching using invariant features. *International journal of computer vision* 74(1):59–73.

Deng, J.; Guo, J.; Xue, N.; and Zafeiriou, S. 2019. Arcface: Additive angular margin loss for deep face recognition. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 4690–4699.

Fu, J.; Liu, J.; Tian, H.; Li, Y.; Bao, Y.; Fang, Z.; and Lu, H. 2019. Dual attention network for scene segmentation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 3146–3154.

Gong, M.; Xu, Y.; Li, C.; Zhang, K.; and Batmanghelich, K. 2019. Twin auxiliary classifiers gan. *arXiv preprint arXiv:1907.02690*.

Krizhevsky, A.; Sutskever, I.; and Hinton, G. E. 2012. Imagenet classification with deep convolutional neural networks. In *Advances in neural information processing systems*, 1097–1105.

Li, W.; Zhao, R.; Xiao, T.; and Wang, X. 2014. Deepreid: Deep filter pairing neural network for person re-identification. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, 152–159.

Li, W.; Zhu, X.; and Gong, S. 2018. Harmonious attention network for person re-identification. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2285–2294.

Lin, S.; Ji, R.; Chen, C.; Tao, D.; and Luo, J. 2018. Holistic cnn compression via low-rank decomposition with knowledge transfer. *IEEE transactions on pattern analysis and machine intelligence*.

Liu, W.; Lin, R.; Liu, Z.; Liu, L.; Yu, Z.; Dai, B.; and Song, L. 2018. Learning towards minimum hyperspherical energy. In *Advances in Neural Information Processing Systems*, 6222–6233.

Mishchuk, A.; Mishkin, D.; Radenovic, F.; and Matas, J. 2017. Working hard to know your neighbor's margins: Local descriptor learning loss. In *Advances in Neural Information Processing Systems*, 4826–4837.

Ramachandran, P.; Parmar, N.; Vaswani, A.; Bello, I.; Levskaya, A.; and Shlens, J. 2019. Stand-alone self-attention in vision models. *arXiv preprint arXiv:1906.05909*.

Ristani, E.; Solera, F.; Zou, R.; Cucchiara, R.; and Tomasi, C. 2016. Performance measures and a data set for multi-target, multi-camera tracking. In *European Conference on Computer Vision*, 17–35. Springer.

Si, J.; Zhang, H.; Li, C.-G.; Kuen, J.; Kong, X.; Kot, A. C.; and Wang, G. 2018. Dual attention matching network for context-aware feature sequence based person re-identification. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 5363–5372.

Sun, Y.; Zheng, L.; Yang, Y.; Tian, Q.; and Wang, S. 2018. Beyond part models: Person retrieval with refined part pooling (and a strong convolutional baseline). In *Proceedings of the European Conference on Computer Vision (ECCV)*, 480–496.

Tian, Y.; Fan, B.; and Wu, F. 2017. L2-net: Deep learning of discriminative patch descriptor in euclidean space. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 661–669.

Vaswani, A.; Shazeer, N.; Parmar, N.; Uszkoreit, J.; Jones, L.; Gomez, A. N.; Kaiser, Ł.; and Polosukhin, I. 2017. Attention is all you need. In *Advances in neural information processing systems*, 5998–6008.

Wang, C.; Zhang, Q.; Huang, C.; Liu, W.; and Wang, X. 2018a. Mancs: A multi-task attentional network with curriculum sampling for person re-identification. In *Proceedings of the European Conference on Computer Vision (ECCV)*, 365–381.

Wang, X.; Girshick, R.; Gupta, A.; and He, K. 2018b. Non-local neural networks. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 7794–7803.

Xu, Y.; Gong, M.; Liu, T.; Batmanghelich, K.; and Wang, C. 2019. Robust angular local descriptor learning. *arXiv preprint arXiv:1901.07076*.

Yu, X.; Liu, T.; Wang, X.; and Tao, D. 2017. On compressing deep models by low rank and sparse decomposition. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 7370–7379.

Zhang, H.; Goodfellow, I.; Metaxas, D.; and Odena, A. 2018. Self-attention generative adversarial networks. *arXiv preprint arXiv:1805.08318*.

Zheng, L.; Shen, L.; Tian, L.; Wang, S.; Wang, J.; and Tian, Q. 2015. Scalable person re-identification: A benchmark. In *Proceedings of the IEEE international conference on computer vision*, 1116–1124.

Zheng, Z.; Zheng, L.; and Yang, Y. 2017. Unlabeled samples generated by gan improve the person re-identification baseline in vitro. In *Proceedings of the IEEE International Conference on Computer Vision*, 3754–3762.

Zhong, Z.; Zheng, L.; Cao, D.; and Li, S. 2017. Re-ranking person re-identification with k-reciprocal encoding. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 1318–1327.