Self-Supervised Knowledge Transfer via Loosely Supervised Auxiliary Tasks

Seungbum Hong, Jihun Yoon, and Min-Kook Choi
VisionAI, hutom
Seoul, Republic of Korea
{qbration21, jhyoon2020, mkchoi}@hutom.io

Junmo Kim
KAIST
Daejeon, Republic of Korea
junmo.kim@kaist.ac.kr

Abstract

Knowledge transfer using convolutional neural networks (CNNs) can help efficiently train a CNN with fewer parameters or maximize the generalization performance under limited supervision. To enable a more efficient transfer of pretrained knowledge under relaxed conditions, we propose a simple yet powerful knowledge transfer methodology without any restrictions regarding the network structure or dataset used, namely self-supervised knowledge transfer (SSKT), via loosely supervised auxiliary tasks. For this, we devise a training methodology that transfers previously learned knowledge to the current training process as an auxiliary task for the target task through self-supervision using a soft label. The SSKT is independent of the network structure and dataset, and is trained differently from existing knowledge transfer methods; hence, it has an advantage in that the prior knowledge acquired from various tasks can be naturally transferred during the training process to the target task. Furthermore, it can improve the generalization performance on most datasets through the proposed knowledge transfer between different problem domains from multiple source networks. SSKT outperforms the other transfer learning methods (KD, DML, and MAXL) through experiments under various knowledge transfer settings. The source code will be made available to the public[1].

1. Introduction

Knowledge transfer is the most representative training methodology for improving the generalization capability and training efficiency of convolutional neural networks (CNNs). The most widely used knowledge transfer is transfer learning [37, 35], which uses pretrained weights trained on large-scale datasets as initial values for new tasks. Pretrained weights have been used as feature encoders after fine-tuning in different vision tasks such as image classification, object detection, and semantic segmentation [24, 25, 29]. Taskonomy [38, 39] provides an extensive database and analysis approach for studying the effects of transfer learning using pretrained networks on the fine-tuning of target tasks. However, transfer learning using pretrained networks presupposes structural dependencies that must share essentially the same (whole or partial) network structure as the source network. Moreover, questions have been raised on whether transfer learning using pretrained weights can provide appropriate initial weights, which can help in improving convergence or training with less amount of data but not very effective for knowledge transfer to different tasks [12].

Another representative example of knowledge transfer using CNN is knowledge distillation (KD) [14], where pretrained teacher networks distill and deliver dark (hidden) knowledge in the inference output or learned features during student network training. KD methods include loss-based KD, which delivers dark knowledge through loss with a soft

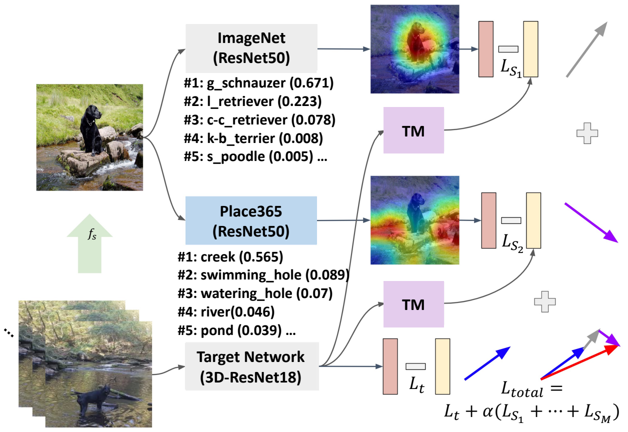

Figure 1. **Motivation of the SSKT.** The CNN responses that can be obtained from the same image vary depending on the type of supervision. SSKT performs auxiliary training using soft labels to convey the previously trained prior knowledge from the source tasks. At this time, the supervision used in the auxiliary training generates a gradient (grey and purple arrows) that can improve the generalization performance of the target task. The gradient to update the weight of the target network is obtained as a linear combination of all losses (red arrow). TM is a transfer module.

[1] https://github.com/generation21/generation6011

label [14], and KD, wherein the similarity between the features extracted are exploited in specific stages of the CNN [30]. Knowledge transfer through KD has the advantage of training student networks that have a better-generalized performance with fewer parameters comparable to teacher networks. In the general KD setup, the training datasets of both teacher and student networks should be the same. However, for a more general knowledge transfer, these constraints must be relaxed.

Figure 1 shows an example in which the information to be transferred can vary depending on the type of supervision of each pretrained CNN through Grad-CAM visualization [32] for the same input image. For this purpose, we propose a simple yet powerful method that aims for better generalization with minimal additional supervision. The proposed knowledge transfer conveys prior knowledge as a soft label from multiple pretrained networks obtained using different types of supervision methods in a self-supervised manner. The proposed knowledge transfer method utilizes a pretrained network but is independent of the structure of the target network and the training dataset of the pretrained network to train the target task. The target network is updated through the linear combination of the target loss and multiple auxiliary losses with soft label-based self-supervision from the pretrained networks by trying to guess the type of knowledge inferred from the source network (Figure 1). We expect that the target network reflects dark knowledge as a gradient from multiple supervisions of existing training data through soft label-based self-supervision. Because it does not refer to previously trained data and performs auxiliary training in a self-supervised manner, the proposed knowledge transfer can be performed within any task while requiring only the final output dimension (number of classes) for calculating each auxiliary loss. We call this form of knowledge transfer *self-supervised knowledge transfer* (SSTK). The technical contributions of the SSKT are as follows:

- Unlike previous knowledge transfer techniques, SSKT does not depend on the structure of the source network or the training dataset. SSKT is even possible between structurally complete heterogeneous networks, such as 2D to 3D CNN knowledge transfer.

- SSKT has the potential for simultaneous knowledge transfer from multiple pretrained networks trained on completely different types of datasets or supervision.

- The SSKT improves the generalization performance in most knowledge transfer scenarios. Particularly in different problem domains, SSKT outperforms existing knowledge transfer methods.

Figure 2 shows the differences between SSKT and other knowledge transfer methods. For clarity, we take the problem domain as a superset of tasks (e.g., image classification, action recognition, object detection, and semantic segmentation), and the task is bounded in the same problem domain (e.g., ImageNet, Places365, and STL10 in image classification).

2. Related Works

Transfer learning with a pretrained initializer. As the most commonly used method of knowledge transfer, transfer learning with a pretrained initializer has been applied to various fields of computer vision using high-performance models [20, 33, 13] with a large amount of training data. Because low-level feature information is shared by most visual recognition problems, fine-tuning is often applied by updating only a specific upper-layer when training new tasks [25, 29]. Generally, this can be used in the same network structure, and many image recognition applications use only the feature encoder part of the entire task architecture [24, 25, 29]. Although transfer learning using ImageNet has been successful so far, questions on transfer learning using ImageNet data have been raised [12], and the experimental results of object recognition and key point recognition problems have been reported as examples where the pretrained initial values do not convey additional knowledge to the different problem domains.

Knowledge distillation. Pretrained teacher networks with larger parameters can improve the generalization performance of student networks by performing knowledge transfer during the training process to a relatively small number of parameters [14]. Student networks often show better generalization performance over teacher networks [8]. Recent knowledge distillation techniques for effective knowledge transfer used the relationship between the inferred posteriors from the teacher and student networks [27] or the dataset as a subclass of the original classes to be learned [26]. In [8], the ensemble model achieved a high recognition performance by distilling the knowledge of student networks with different initialization values using teacher and student networks with the same structure. SSKT uses a knowledge transfer method, such as KD with a soft label. However, unlike existing KD methods, because the SSKT transfers knowledge through an auxiliary task-based self-supervision technique, there is no dataset dependency.

Deep mutual learning (DML). DML is one of the representative variations of the KD-based knowledge transfer method that uses multiple networks for efficient training through KD loss within networks [40]. DML could achieve performance improvement through mutual learning in the middle of training without requiring a powerful pretrained teacher network. Recently, a learning methodology using feature fusion [18], generative adversarial networks [3], and collaborative learning [36] structures has been proposed to improve the efficiency of mutual learning during training. It differs from SSKT in that the DML performs multitask

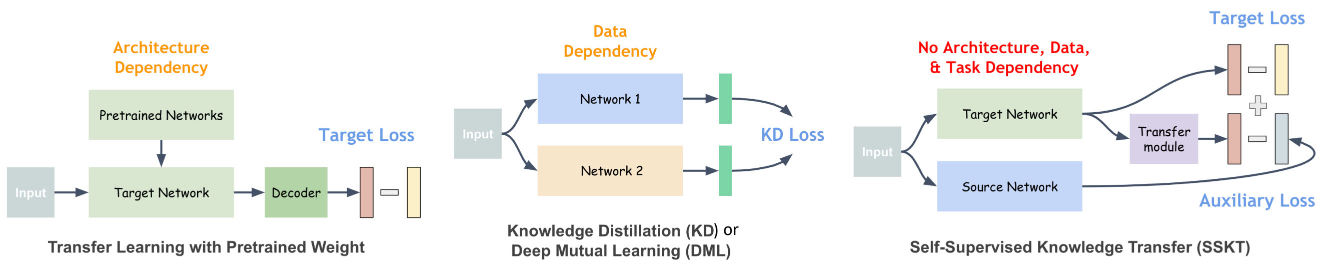

Figure 2. **Schematic of the difference between the proposed SSKT and other knowledge transfer methods.** Unlike existing knowledge transfer methods, such as KD and DML, SSKT is independent of the network structure and dataset. In addition, generalization performance improvements can be achieved without additional supervision during the training process.

training-based learning on the same dataset and updates all the networks during training. In addition, it is fundamentally different from SSKT in that DML does not utilize prerequisite knowledge from former training.

Auxiliary learning. Unlike general multi-task learning, in the case of auxiliary task learning, the purpose of a deep neural network is not to increase the performance of auxiliary tasks but to improve the performance of the target task. It was recently reported that auxiliary task learning could help improve the test performance of the target task, and the cosine similarity between the gradient losses of the target and auxiliary tasks was analyzed to visualize the cause of contribution by the auxiliary task [6]. The basic structure of the SSKT was inspired by meta-learning techniques using auxiliary tasks under self-supervision (MAXL) [23]. However, MAXL requires a hierarchical auxiliary class structure defined manually for the primary class to be trained. SSKT was able to achieve higher performance than MAXL by using only the number of classes of source tasks as additional supervision information.

Self-supervised learning (SSL). Typical self-supervised learning techniques train pretrained weights that perform unsupervised training on pretext tasks to obtain good initial weights [9, 2, 1, 11]. The pretrained network utilized in SSKT was not obtained by unsupervised training, and the target task training differed from the general self-supervised approach, requiring meta information such as the number of classes used in the source network training. However, SSKT can still be considered self-supervised from the viewpoint of training auxiliary tasks using only soft labels from pretrained networks, without requiring additional supervision with hard labels when training the auxiliary task.

Domain adaptation or expansion. Domain adaptation or extension [22, 16] aims to improve the generalization performance for new domains with existing trained networks. However, unlike in the case of the general domain adaptation problem, SSKT has a significant difference in using the prior knowledge trained in the existing domain for new network training without sharing parameters. In addition, because it does not include any update process for the pretrained network, the update is performed regardless of the performance of the source domain. Moreover, there was no change in the inference performance of the source domain after training.

3. Self-Supervised Knowledge Transfer

SSKT supports a training structure that enables knowledge transfer in a variety of scenarios using a CNN. Various scenarios refer to situations not influenced by the network architecture of the target task to be trained and in which the knowledge transfer does not depend on the type of task. We devised a structure that transfers knowledge naturally without compromising the training information of the pretrained network or requiring additional supervision during the target task training process. We achieved this using soft label-based knowledge transfer techniques with auxiliary task learning through self-supervision for the various domains of image recognition variants.

3.1. SSKT with Single Source

We define a multi-task network for auxiliary learning, $h_t(x; \theta_t, D_t, T_t)$, where x is the input, θ_t is a parameter of the target network, D_t is a target dataset, and T_t is the task to be trained. θ_t is updated simultaneously through the target loss and auxiliary loss during training to solve the primary task. $h_s(x; \theta_s, D_s, T_s)$ describes a source network that receives input x and delivers knowledge to the target network. θ_s denotes a parameter trained by the source task T_s for the source dataset (ImageNet and Places365) D_s. θ_s is not updated during target task training. The multi-task network h_t for the primary task training shares up to the top layer of the convolutional block below the primary and auxiliary branches, except for the last feature layer. The last feature layer of the branch for the target task loss outputs $h_t^{prim}(x; \theta_t, D_t, T_t)$, and the branch for the auxiliary task loss outputs $h_t^{aux}(x; \theta_t, D_t, T_s)$. At this time, the auxiliary task branch could have a transfer module composed of the summation of the bottleneck structure from each convolutional block for effective feature encoding for the auxiliary task. Figure 3 shows an example of a multitask network with a transfer module. The ground truth label of the primary task for the total loss of h_t is given by y_t^{prim}, for x

belonging to D_t. The ground truth label for the auxiliary task is given by the softmax output y_s^{aux} from h_s.

The total loss function for a single-task knowledge transfer based on a multitask network is as follows.

$$\underset{\theta_t}{\operatorname{argmin}} \Big(L(h_t^{prim}(x_i; \theta_t.D_t, T_t), y_{t,i}^{prim}) \\ + \alpha L(h_t^{aux}(x_i; \theta_t.D_t, T_s), y_{s,i}^{aux}) \Big), \tag{1}$$

where i is the i^{th} batch of the training data, α is a balanced parameter for the total loss, and $y_{s,i}^{aux} = h_s(x_i; \theta_s, D_t, T_s)$ is the softmax outputted from the pretrained source network and which conveys the dark knowledge of the pretrained dataset using soft labels. We applied two types of losses of cross-entropy (CE) and knowledge distillation (KD) [14] for auxiliary task learning:

$$L_{CE} = -ylog(c(\frac{g(x;\theta)}{T})), \tag{2}$$

$$L_{KD} = KL\Big(c(\frac{g(x;\theta_s)}{T}), c(\frac{g(x;\theta_t)}{T})\Big), \tag{3}$$

where y is a one-hot vector from the ground truth label, $g(x;\theta)$ is the final feature outputted from the network during training, T controls the softening intensity for the softmax output as the temperature parameter, c denotes the softmax function, and KL denotes the KL divergence between the output distributions. The gradient update after $k+1$ iterations is described by

$$\theta_t^{k+1} = \theta_t^k - \eta \nabla_{\theta_t} \Big(L(h_t^{prim}(x_i; \theta_t.D_t, T_t), y_{t,i}^{prim}) \\ + \alpha L(h_t^{aux}(x_i; \theta_t.D_t, T_s), y_{s,i}^{aux}) \Big) \tag{4}$$

where η is the learning rate. Figure 2(a) shows the structural schematics for the SSKT with a single-source scenario.

Transfer module (TM). To encourage the prediction of $y_{s,i}^{aux}$ by h_t, we design a bottleneck structure-based transfer module that supports an auxiliary task using the feature outputted from each convolutional block. For an example of a ResNet-based model, each output of the res block is encoded as a fixed-size output by bottleneck and average pooling, and finally, each feature outputted from the transfer bottleneck is summed (see Figure 3).

3.2. SSKT with Multiple Source

SSKT can be easily extended to multiple h_s using multiple auxiliary losses. We improved the structure by including M source tasks for knowledge transfer to target tasks, as follows:

$$\underset{\theta_t}{\operatorname{argmin}} \Big(L(h_t^{prim}(x_i; \theta_t.D_t, T_t), y_{t,i}^{prim}) \\ + \alpha (L(h_t^{aux}(x_i; \theta_t.D_t, T_{s_1}), y_{s_1,i}^{aux}) \\ + L(h_t^{aux}(x_i; \theta_t.D_t, T_{s_2}), y_{s_2,i}^{aux}) + \dots \\ + L(h_t^{aux}(x_i; \theta_t.D_t, T_{s_M}), y_{s_M,i}^{aux})) \Big), \tag{5}$$

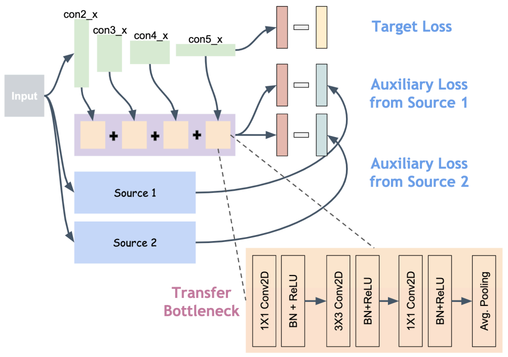

Figure 3. **Schematic of transfer modules for efficient auxiliary learning.** The transfer module used in the SSKT consists of the summation of the feature outputs of the bottleneck layers from each convolutional block. The schematic shows an example of the transfer module with ResNet variants for SSKT using multiple sources.

where s_M denotes the index of the source task for knowledge transfer. Theoretically, if the memory is sufficient, we can perform multiple task knowledge transfers from a large number of M source tasks.

3.3. SSKT within Different Problem Domains

Until now, transfers for single and multiple sources have been performed assuming tasks in the same problem domain that do not specify the forms of D_t and D_s. We set up a scenario to enable knowledge transfer even when the modalities of D_t and D_s are different or when the problem domains of the tasks are different. In this case, because the characteristics of the input data are different and there is a large difference in the structure of the network for training, the preparation process for knowledge transfer involves data conversion or preprocessing. For knowledge transfer with different problem domains, the total loss function can be defined as:

$$\underset{\theta_t}{\operatorname{argmin}} \Big(L(h_t^{prim}(x_i; \theta_t.D_t, T_t), y_{t,i}^{prim}) \\ + \alpha L(h_t^{aux}(f_s(x_i); \theta_t.D_t, T_s), y_{s,i}^{aux}) \Big), \tag{6}$$

where the data transformation function f_s converts the data type to match the source task and infer the recognition information to the task of the source domain. For example, if T_t is an action recognition problem using a 3D-CNN, the input $x^{w \times h \times d} \in D_t$ is defined as a 3D tensor. In this case, if a pretrained network for knowledge transfer is obtained through the image recognition problem T_s using 2D-CNN, $f_s : x^{w \times h \times d} \to \hat{x}^{w \times h}$ should be defined as a function that maps a 3D tensor to a 2D matrix into which h_s can be inputted.

Knowledge transfer to other problem domains can also be done with multiple (M) source tasks:

Table 1. Training details for each task, for knowledge transfer with SSKT. IC denotes image classification, AC denotes action classification, MCIC denotes multi-class image classification, β represents momentum, and λ represents weight decay. CE denotes cross-entropy, KD denotes knowledge distillation, and BCE denotes binary cross-entropy. In the training model, (s) was trained by learning from scratch, and (f) was trained by fine-tuning.

Task (T)	Dataset (D)	# cls	Model	Optimizer	LR Scheduler	β,λ	Loss [Task + Auxiliary]
IC (T_t)	CIFAR10 (C10)	10	ResNet[20, 32]	SGD	step (0.1,[150:250:350])	0.9, 5e-4	[CE + CE or KD]
	CIFAR100 (C100)	100	ResNet[20, 32]	SGD	step (0.1,[60:120:160:200])	0.9, 5e-4	[CE + CE or KD]
	STL10 (S10)	10	ResNet18[20, 32, 44]	SGD	step (0.1,[60:120:160:200])	0.9, 5e-4	[CE + CE or KD]
			MoblieNetV2	SGD	step (0.1,[60:120:160:200])	0.9, 5e-4	[CE + CE or KD]
			DenseNet121	SGD	step (0.1,[60:120:160:200])	0.9, 5e-4	[CE + CE or KD]
	Places365 (P)	365	ResNet18	SGD	step (0.1,[30:60:90])	0.9, 1e-4	[CE + CE or KD]
	ImageNet (I)	1000	ResNet18	SGD	step (0.1,[30:60:90])	0.9, 1e-4	[CE + CE or KD]
MCIC (T_t)	Pascal VOC (VOC)	20	ResNet[18, 34, 50] (s)	SGD	step (0.1,[30:60:90])	0.9, 1e-4	[BCE + CE or KD]
			ResNet18 (f)	SGD	step (0.01,[30:60:90])	0.9, 1e-4	[BCE + CE or KD]
AC (T_t)	UCF101 (U101)	101	3D-ResNet18 (s)	SGD	reduce on plateau (0.1)	0.9, 1e-3	[CE + CE or KD]
			3D-ResNet18 (f)	SGD	reduce on plateau (0.01)	0.9, 1e-3	[CE + CE or KD]
	HMDB51 (H51)	51	3D-ResNet18 (s)	SGD	reduce on plateau (0.1)	0.9, 1e-3	[CE + CE or KD]
			3D-ResNet18 (f)	SGD	reduce on plateau (0.01)	0.9, 1e-3	[CE + CE or KD]
IC (T_s)	Places365 (P)	365	ResNet50	[41]	[41]	[41]	[41]
	ImageNet (I)	1000	ResNet50	[28]	[28]	[28]	[28]

$$\begin{aligned}\underset{\theta_t}{\operatorname{argmin}} \Big(& L(h_t^{prim}(x_i;\theta_t.D_t,T_t), y_{t,i}^{prim}) \\ & +\alpha(L(h_t^{aux}(f_{s_1}(x_i);\theta_t.D_t,T_{s_1}), y_{s_1,i}^{aux}) \\ & +L(h_t^{aux}(f_{s_2}(x_i);\theta_t.D_t,T_{s_2}), y_{s_2,i}^{aux}) + \dots \\ & +L(h_t^{aux}(f_{s_M}(x_i);\theta_t.D_t,T_{s_M}), y_{s_M,i}^{aux}))\Big), \end{aligned} \tag{7}$$

in this case, up to M transformation functions can be defined. Gradient updates for knowledge transfer to other problem domains are defined in the same way as single- or multiple-task knowledge transfers between the same domains.

4. Experimental Results

To verify the SSKT, we designed a series of experimental scenarios. First, we set a knowledge transfer with a single-source task in the same problem domain. A pretrained CNN on image classification tasks (ImageNet and Places365) was used for knowledge transfer for the same problem domain. Second, we performed knowledge transfer for the image classification problem using multiple source networks. In this case, the target network for target task learning had two or more auxiliary tasks. Third, knowledge transfer scenarios using source networks with different problem domains were tested in two ways. The first method involved target task learning with a different purpose and knowledge transfer to a domain within the same image recognition category. For example, to solve the multiclass classification problem for 2D images, the target network (2D-CNN) has the same type of architecture as the source network (2D-CNN). In this case, knowledge transfer between domains is possible without any special definition of the transformation function f_s for the source network. The second scenario involves knowledge transfer using different problem domains using heterogeneous networks. For example, in training the action recognition network using 3D-CNN, we verified whether the network learned by the image classification problem could transfer the knowledge to the action recognition problem.

In addition to verifying the knowledge transfer performance of SSKT for each knowledge transfer scenario, the optimization results for the hyperparameters of the loss functions were included. Moreover, our experiments include the relationship between the change in knowledge transfer performance based on the model architecture and fine tuning, and a performance comparison with other knowledge transfer methods such as KD, DML, and MAXL. All the experimental results included in the table are selected based on the highest performance among combinations of auxiliary loss, transfer module, and source network[2]. Table 1 summarizes the experimental settings for the verification of the SSKT.

4.1. Image Classification → Image Classification

SSKT Setting. To perform SSKT between image classification problems, we used networks pretrained on ImageNet (I) [5] or Places365 (P) [41] dataset as the source network. The target task datasets were CIFAR10, CIFAR100 [19], and STL10 [4], which contain relatively small amounts of data, as well as large datasets such as ImageNet and Places365. As a pretrained source network, we used the ResNet50 network provided in the PyTorch *torchvision* package [28]. The transfer module is composed of a bottleneck and average pooling, as shown in Figure 3. Multiple pretrained networks can be utilized in SSKT to apply for multisource task-based knowledge transfer. We used both a ResNet50 model trained on ImageNet and Places365 (P+I) datasets as our source tasks.

Results. Table 2 shows the performances of learning from scratch and SSKT for the image classification problem on

[2]Additional combinations of test results and architecture details can be found in the supplementary materials.

Table 2. Performance change of SSKT for a single source compared to training from scratch. The best-performing model for each dataset is highlighted in bold. All the experiments evaluated the test performance thrice from the same random seed for the model. TM denotes transfer module, and R[depth] denotes ResNet structure.

T_s	T_t	Model	Method	TM	Loss	acc.
-	C10	R20	scratch	-	CE	92.19±0.09
P		R20	SSKT	o	CE+KD	92.25±0.04
I		R20	SSKT	o	CE+CE	92.44±0.05
P+I		R20	SSKT	x	CE+KD	**92.46±0.15**
-		R32	scratch	-	CE	93.21±0.09
P		R32	SSKT	x	CE+KD	92.87±0.31
I		R32	SSKT	x	CE+CE	93.26±0.08
P+I		R32	SSKT	o	CE+CE	**93.38±0.02**
-	C100	R20	scratch	-	CE	68.26±0.36
P		R20	SSKT	x	CE+KD	68.01±0.42
I		R20	SSKT	o	CE+CE	**68.63±0.12**
P+I		R20	SSKT	o	CE+CE	68.56±0.23
-		R32	scratch	-	CE	70.33±0.19
P		R32	SSKT	x	CE+CE	69.97±0.16
I		R32	SSKT	o	CE+CE	70.75±0.06
P+I		R32	SSKT	o	CE+CE	**70.94±0.36**
-	S10	R20	scratch	-	CE	81.15±0.34
P		R20	SSKT	o	CE+CE	82.76±0.05
I		R20	SSKT	o	CE+CE	83.45±0.07
P+I		R20	SSKT	o	CE+CE	**84.56±0.35**
-		R32	scratch	-	CE	81.19±0.17
P		R32	SSKT	o	CE+CE	83.06±0.27
I		R32	SSKT	o	CE+CE	**83.68±0.28**
P+I		R32	SSKT	o	CE+CE	83.4±0.2
-		R44	scratch	-	CE	80.18±0.54
P		R44	SSKT	o	CE+CE	82.68±0.39
I		R44	SSKT	o	CE+CE	**83.59±0.13**
P+I		R44	SSKT	o	CE+CE	83.44±0.15

CIFAR10 (C10), 100 (C100), and STL10 (S10) datasets. For knowledge transfer between image classification problems, the use of SSKT improves the generalization performance on all the tested datasets. The performance change when the transfer module is applied to efficiently transform the features for auxiliary task learning is also evident. Notably, the highest improvement in the generalization performance was on the SLT10 dataset, which contains a small proportion of the training data. This means that knowledge transfer using SSKT is more effective when the amount of supervision is relatively low. Table 3 shows the performance changes of the SSKT on large-scale image datasets. When SSKT was applied to both Place365 and ImageNet, there was a steady performance improvement. In particular, the highest performance improvement was obtained when the source networks were used simultaneously (P + I). However, for large datasets, it was difficult to confirm the effect of improving the performance of the transfer module.

4.2. Image Classification → Multi-class Image Classification

SSKT Setting. To verify whether knowledge transfer using SSKT is effective between different problem domains,

Table 3. Performance change when applying the SSKT and training from scratch on a large-scale image dataset.

T_s	T_t	Model	Method	TM	Loss	acc.
-	P	R18	scratch	-	CE	50.92
P		R18	SSKT	o	CE+CE	54.5
I		R18	SSKT	o	CE+CE	53.67
P+I		R18	SSKT	x	CE+CE	**54.78**
-	I	R18	scratch	-	CE	64.14
P		R18	SSKT	o	CE+CE	64.99
I		R18	SSKT	x	CE+CE	67.79
P+I		R18	SSKT	x	CE+CE	**70.57**

Table 4. Results of SSKT from the image classification problem to the multi-class image classification problem.

T_s	T_t	Model	Method	TM	Loss	acc.
-	VOC	R18	scratch	-	BCE	67.28±0.25
P		R18	SSKT	o	BCE+CE	74.76±0.17
I		R18	SSKT	x	BCE+CE	74.78±0.09
P+I		R18	SSKT	o	BCE+CE	**76.42±0.06**
-		R34	scratch	-	BCE	66.0±0.49
P		R34	SSKT	o	BCE+CE	75.65±0.12
I		R34	SSKT	o	BCE+CE	75.14±0.14
P+I		R34	SSKT	o	BCE+CE	**77.02±0.02**
-		R50	scratch	-	BCE	61.16±0.34
P		R50	SSKT	o	BCE+CE	74.44±0.06
I		R50	SSKT	o	BCE+CE	74.24±0.05
P+I		R50	SSKT	o	BCE+CE	**77.1±0.14**

knowledge transfer between image classification and multi-class image classification was performed. We tested multi-class image classification using the PASCAL VOC dataset [7] as the target task. The target loss function for solving the multiclass image classification was used as the binary cross-entropy (BCE). Table 1 lists the training details for the target and source tasks.

Results. Table 4 shows the performance of learning from scratch for multiclass image classification and the performance change when SSKT is included in the training process. For the PASCAL VOC multi-class classification task, we used the standard average precision (AP) as the accuracy measurement to evaluate the predictions. The use of SSKT significantly improved the knowledge transfer performance between different problem domains. The improvement was greater than that for knowledge transfer between the same problem domains. In addition, we verified an additional performance improvement when applying SSKT to multi-source networks.

4.3. Image Classification → Action Classification

SSKT Setting. Although there is a difference between problem domains, the performance improvement of knowledge transfer may be relatively predictable because the modalities of the input data of the source and target tasks are the same. We used a pretrained image classification network as the source task and set the target task as an action recognition problem. We employed 3D-CNN to design an experiment for a more extended knowledge transfer. To utilize the dark knowledge of the source network as described in Equation (6), we need to define a transformation function $f_s : x^{w \times h \times d} \rightarrow \hat{x}^{w \times h}$. Because SSKT is not designed to define good f_s, we simply limited the role of f_s in extracting the center frame from a 3D video clip. As a transfer module for inferring auxiliary tasks in a 3D-CNN, the output features of the target model were further processed using a 3D bottleneck structure. We used a 3D ResNet-based baseline model [10] to train the action classification networks. Table 1 lists the training details for 3D ResNet and source networks.

Results. Table 5 shows the performance of learning from scratch with the 3D-ResNet on the UCF101 [34] and HMDB51 [21] datasets, as well as the performance change when SSKT was used in the training process. All the results were obtained from *split 1* for each dataset. Note that the auxiliary task in SSKT prevents overfitting during the training of the 3D-CNN model. Similar to other SSKT settings, the performance improvement was highest when using a multi-source network. We further note that the use of SSKT between heterogeneous problem domains can improve the performance further, depending on the definition of f_s.

4.4. Further Analysis and Discussion

Parameter optimization. We performed a series of experiments for the temperature parameter T included in the auxiliary loss, the balance parameter α included in the total loss, and the presence of the transfer module. Figure 4 shows the graphs of the parameter optimization results on the STL10 and PASCAL VOC datasets. For STL10, the use of a transfer module in the SSKT process yielded a high performance in most cases; however, in the case of PASCAL VOC, the transfer module did not play a significant role.

Model comparison. We applied SSKT to MobileNet V2 (MV2) [31] and DenseNet121 (D121) [15] to evaluate the model performance in CNN architectures other than ResNet variants. The source network used the ResNet50 model, which was previously trained on Places365 and ImageNet, similar to other SSKT settings. In evaluating the STL10 dataset, Table 6 shows that MobileNet V2 and DenseNet121 exhibit performance improvements with the SSKT. The results in the table show the highest performance in the configuration among single-and multi-source SSKT settings.

Relation to fine-tuning. In SSKT scenarios, fine-tuning and learning from scratch can be applied for target network training. We adopted fine-tuning-based SSKT with a ResNet18 model trained in ImageNet for a multiclass image classification problem using the PASCAL VOC dataset. In addition, using the pretrained 3D-ResNet18 model on the Kinetics-400 dataset [17], SSKT was applied to the action classification problem for the UCF101 and HMDB51 datasets. Table 7 shows that even if fine-tuning based on predetermined weights is performed, the performance is improved in all the experimental settings. However, unlike the evaluation results in the case of training from scratch, the performance improvement from multiple sources was not significant, and the best performance was obtained when the KD loss was used.

Table 5. Results of SSKT from image classification to action classification with different modalities of the input data.

T_s	T_t	Model	Method	TM	Loss	acc.
-	U101	3DR18	scratch	-	CE	43.28
P		3DR18	SSKT	o	CE+CE	45.35
I		3DR18	SSKT	x	CE+CE	46.62
P+I		3DR18	SSKT	x	CE+CE	**52.19**
-	H51	3DR18	scratch	-	CE	17.14
P		3DR18	SSKT	o	CE+CE	18.77
I		3DR18	SSKT	o	CE+KD	18.77
P+I		3DR18	SSKT	o	CE+CE	**20.54**

Table 6. SSKT results using MobileNet V2 (MV2) and DenseNet121 (D121).

T_s	T_t	Model	Method	TM	Loss	acc.
-	S10	R20	scratch	-	CE	81.15±0.34
-		MV2	scratch	-	CE	72.26±0.83
-		D121	scratch	-	CE	72.02±0.48
P+I		R20	SSKT	o	CE+ CE	**84.56±0.35**
P+I		MV2	SSKT	o	CE+CE	**76.96±0.39**
P+I		D121	SSKT	x	CE+CE	**77.03±0.17**

Table 7. SSKT results using pretrained weights. *ft* denotes fine-tuning and K denotes Kinetics-400 dataset.

T_s	T_t	Model	Method	TM	Loss	acc.
-	VOC	R18	*ft* (I)	-	CE	90.52±0.11
P		R18	SSKT	x	CE+KD	**92.28±0.06**
I		R18	SSKT	x	CE+KD	92.26±0.07
P+I		R18	SSKT	o	CE+KD	92.25±0.07
-	U101	3DR18	*ft* (K)	-	CE	83.95
P		3DR18	SSKT	x	CE+KD	**84.58**
I		3DR18	SSKT	o	CE+KD	84.37
P+I		3DR18	SSKT	o	CE+KD	84.19
-	H51	3DR18	*ft* (K)	-	CE	56.64
P		3DR18	SSKT	o	CE+KD	**57.82**
I		3DR18	SSKT	o	CE+KD	57.75
P+I		3DR18	SSKT	o	CE+CE	57.29

Comparison with other knowledge transfer methods. For a further analysis of SSKT, we compared its performance with those of typical knowledge transfer methods, namely KD and DML. For KD, the details for learning were set the same as in [14], and for DML, training was performed in the same way as in [40]. Table 8 shows the evaluation performance in terms of knowledge transfer. In the case of 3D-CNN-based action classification, KD was not performed, and the network used for DML had the model with the same conditions as the source and target networks of SSKT. In the case of DML for 3D-CNN, training was performed by replacing ResNet50 with 3D-ResNet50 under the same conditions. In the case of action classification, both learning from scratch and fine-tuning results were included. In most cases, under similar training conditions,

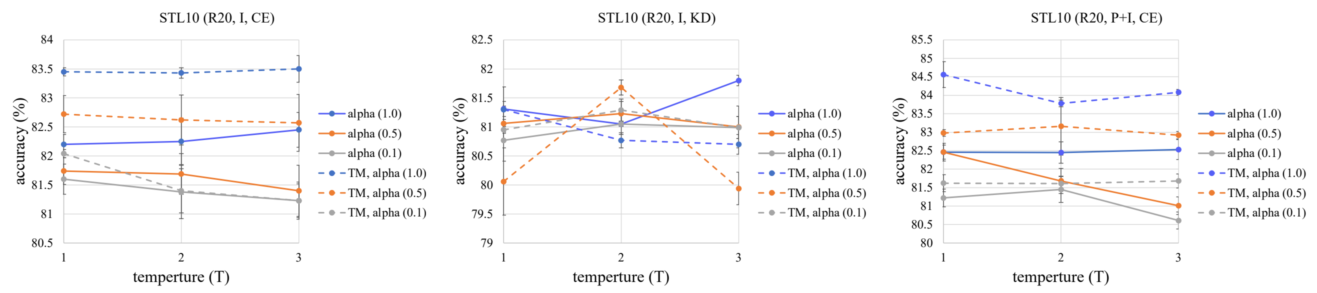

Figure 4. **Hyper parameter optimization of SSKT.** The title of each graph is composed of D_t (target model, T_s, auxiliary loss). T is the temparture parameter of each auxiliary loss, and α is the balance parameter of the total loss.

Table 8. Comparison with other knowledge transfer methods. (s) denote learning from scratch and (f) denote fine-tuning.

T_t	Model	KD	DML	SSKT (T_s)
C10	R20	91.75±0.24	92.37±0.15	**92.46±0.15** (P+I)
	R32	92.61±0.31	93.26±0.21	**93.38±0.02** (P+I)
C100	R20	68.66±0.24	**69.48±0.05**	68.63±0.12 (I)
	R32	70.5±0.05	**71.9±0.03**	70.94±0.36 (P+I)
S10	R20	77.67±1.41	78.23±1.23	**84.56±0.35** (P+I)
	R32	76.07±0.67	77.14±1.64	**83.68±0.28** (I)
VOC	R18	64.11±0.18	39.89±0.07	**76.42±0.06** (P+I)
	R34	64.57±0.12	39.97±0.16	**77.02±0.02** (P+I)
	R50	62.39±0.6	39.65±0.03	**77.1±0.14** (P+I)
U101	3DR18 (s)	-	13.8	**52.19** (P+I)
	3DR18 (f)	-	83.95	**84.58** (P)
H51	3DR18 (s)	-	3.01	**17.91** (P+I)
	3DR18 (f)	-	56.44	**57.82** (P)

Table 9. Comparison with MAXL. V16 denotes VGG16 [33]. F denotes focal loss [23]

T_t	Model (Loss)	MAXL ($\psi[i]$)	SSKT (T_s)	T_s Model
C10	V16 (F)	93.27±0.09 (2)	94.1±0.1 (I)	V16
	V16 (F)	93.47±0.08 (3)	92.72±0.15 (P)	R50
	V16 (F)	93.49±0.05 (5)	93.06±0.2 (P+I)	R50, V16
	V16 (F)	93.10±0.08 (10)	92.54±0.17(P+I)	R50, R50
	V16 (CE)	-	**94.22±0.02** (I)	V16
	V16 (CE)	-	93.12±0.12 (P)	R50
	V16 (CE)	-	93.67±0.17 (P+I)	R50, V16
	V16 (CE)	-	93.26±0.12 (P+I)	R50, R50
	R20 (F)	91.53±0.33 (2)	91.48±0.03 (I)	V16
	R20 (F)	91.52±0.1 (3)	91.26±0.17 (P)	R50
	R20 (F)	91.38±0.47 (5)	90.93±0.01 (P+I)	R50, V16
	R20 (F)	91.56±0.16 (10)	91.11±0.18 (P+I)	R50, R50
	R20 (CE)	-	92.44±0.05 (I)	R50
	R20 (CE)	-	92.25±0.04 (P)	R50
	R20 (CE)	-	**92.46±0.15** (P+I)	R50, R50

SSKT exhibited a higher generalization performance than other knowledge transfer techniques. Moreover, training in problem domains other than the image classification problem for DML was difficult. In particular, in the case of multiclass classification, neither network was sufficiently trained. SSKT achieved improved performance in all problem domains with multiple experimental setups.

Table 9 shows the performance comparison with MAXL, another auxiliary learning-based transfer learning method [23]. In addition to the target task's class label, MAXL requires a manually defined hierarchical auxiliary class structure ($\psi[i] = 2, 3, 5, 10$). We compared the performance on the CIFAR10 dataset by applying the same four-stage hierarchical auxiliary class label as defined in [23]. Despite not requiring additional class labels, SSKT showed better recognition performance than MAXL in all settings. In particular, the same learning scheduler in [23] showed better performance than MAXL (comparisons with VGG16 in Table 9), and SSKT achieved a larger performance gap when changing the source task and hyper-parameter setting.

5. Conclusions

We proposed a simple yet powerful knowledge transfer method, namely SSKT, that enables knowledge transfer between heterogeneous networks and datasets. We validated the SSKT using several knowledge transfer scenarios based on a CNN. For further investigations of the SSKT, depending on the target task where knowledge transfer takes place, additional considerations for the transfer module design must be made. Moreover, the following experiments are required to verify how the performance of the target model varies with the architecture of the source model trained on the same dataset. Variations in the data transformation function f_s for the source networks may be another avenue worth pursuing. Further, there is a need to perform analyses, such as the one proposed in [6], to analyze the factors involved when the backpropagation of the auxiliary task loss affects the performance of the target task. We expect that further research will follow, as SSKT may provide a different perspective on knowledge transfer.

Acknowledgement. This work was supported by the Korea Medical Device Development Fund grant funded by the Korea government (the Ministry of Science and ICT, the Ministry of Trade, Industry and Energy, the Ministry of Health Welfare, the Ministry of Food and Drug Safety) (Project Number: 202012A02-02).

References

[1] T. Chen, S. Kornblith, M. Norouzi, and G. Hinton. Big self-supervised models are strong semi-supervised learners. In *In Proc. of NeurIPS*, 2020. 3

[2] T. Chen, S. Kornblith, M. Norouzi, and G. Hinton. A simple framework for contrastive learning of visual representations. In *In Proc. of ICML*, 2020. 3

[3] I. Chung, S. Park, J. Kim, and N. Kwak. Feature map-level online adversarial knowledge distillation. In *In Proc. of ICML*, 2020. 2

[4] A. Coates, A. Ng, and H. Lee. An analysis of single-layer networks in unsupervised feature learning. In *In Proc. of AISTAT*, 2011. 5

[5] J. Deng, W. Dong, R. Socher, L. J. Li, K. Li, and L. Fei-Fei. Imagenet: A large-scale hierarchical image database. In *In Proc. of CVPR*, 2009. 5

[6] Y. Du, W. M. Czarnecki, S. M. Jayakumar, R. Pascanu, and B. Lakshminarayanan. Adapting auxiliary losses using gradient similarity. In *In Proc. of NeurIPSW*, 2019. 3, 8

[7] M. Everingham, S. M. A. Eslami, L. V. Gool, C. K. I. Williams, Winn J, and A. Zisserman. The pascal visual object classes challenge - a retrospective. *International Journal of Computer Vision*, 111(1):98–136, 2015. 6

[8] T. Furlanello, Z. C. Lipton, M. Tschannen, L. Itti, and A. Anandkumar. Born again neural networks. In *In Proc. of ICML*, 2018. 2

[9] P. Goyal, D. Mahajan, A. Gupta, and I. Misra. A simple framework for contrastive learning of visual representations. In *In Proc. of ICCV*, 2019. 3

[10] K. Hara, H. Kataoka, and Y. Satoh. Can spatiotemporal 3d cnns retrace the history of 2d cnns and imagenet? In *In Proc. of CVPR*, 2018. 7

[11] K. He, H. Fan, Y. Wu, S. Xie, and R. Girshick. Momentum contrast for unsupervised visual representation learning. In *In Proc. of CVPR*, 2020. 3

[12] K. He, R. Girshick, and P. Dollár. Rethinking imagenet pre-training. In *In Proc. of ICCV*, 2019. 1, 2

[13] K. He, X. Zhang, S. Ren, and J. Sun. Deep residual learning for image recognition. In *In Proc. of CVPR*, 2016. 2

[14] G. Hinton, O. Vinyals, and J. Dean. Distilling the knowledge in a neural network. In *In Proc. of NIPSW*, 2014. 1, 2, 4, 7

[15] G. Huang, Z. Liu, L. van der Maaten, and K. Q. Weinberger. Densely connected convolutional networks. In *In Proc. of CVPR*, 2017. 7

[16] H. Jung, J. Ju, M. Jung, and J. Kim. Less-forgetful learning for domain expansion in deep neural networks. In *In Proc. of AAAI*, 2018. 3

[17] W. Kay, J. Carreira, K. Simonyan, B. Zhang, C. Hillier, S. Vijayanarasimhan, F. Viola, T. Green, T. Back, P. Natsev, M. Suleyman, and A. Zisserman. The kinetics human action video dataset. In *CoRR abs/1705.06950*, 2017. 7

[18] J. Kim, M. Hyun, I. Chung, and N. Kwak. Feature fusion for online mutual knowledge distillation. In *In Proc. of ICPR*, 2020. 2

[19] A. Krizhevsky. Learning multiple layers of features from tiny images. In *Technical Report*, 2009. 5

[20] A. Krizhevsky, I. Sutskever, and G. E. Hinton. Imagenet classification with deep convolutional neural networks. In *In Proc. of NIPS*, 2012. 2

[21] H. Kuehne, H. Jhuang, E. Garrote, T. A. Poggio, and T. Serre. Hmdb: a large video database for human motion recognition. In *In Proc. of ICCV*, 2011. 7

[22] Z. Li and D. Hoiem. Learning without forgetting. In *In Proc. of ECCV*, 2016. 3

[23] S. Liu, A. J. Davison, and E. Johns. Self-supervised generalisation with meta auxiliary learning. In *In Proc. of NeurIPS*, 2019. 3, 8

[24] W. Liu, D. Anguelov, D. Erhan, C. Szegedy, S. Reed, C. Y. Fu, and A. C. Berg. Ssd: Single shot multibox detector. In *In Proc. of ECCV*, 2016. 1, 2

[25] J. Long, E. Shelhamer, and T. Darrell. Fully convolutional networks for semantic segmentation. In *In Proc. of CVPR*, 2015. 1, 2

[26] R. Müller, S. Kornblith, and G. Hinton. Subclass distillation. In *CoRR abs/2002.03936*, 2020. 2

[27] W. Park, D. Kim, Y. Lu, and M. Cho. Imagenet classification with deep convolutional neural networks. In *In Proc. of CVPR*, 2019. 2

[28] A. Paszke, S. Gross, F. Massa, A. Lerer, J. Bradbury, G. Chanan, T. Killeen, Z. Lin, N. Gimelshein, L. Antiga, A. Desmaison, A. Köpf, E. Yang, Z. DeVito, M. Raison, A. Tejani, S. Chilamkurthy, B. Steiner, L. Fang, J. Bai, and S. Chintala. Pytorch: An imperative style, high-performance deep learning library. In *In Proc. of NeurIPS*, 2019. 5

[29] S. Ren, K. He, R. Girshick, and J. Sun. Faster r-cnn: Towards real-time object detection with region proposal networks. In *In Proc. of NIPS*, 2015. 1, 2

[30] A. Romero, N. Ballas, S. E. Kahou, A. Chassang, C. Gatta, and Y. Bengio. Fitnets: Hints for thin deep nets. In *In Proc. of ICLR*, 2015. 2

[31] M. Sandler, A. Howard, M. Zhu, A. Zhmoginov, and L. C. Chen. Mobilenetv2: Inverted residuals and linear bottlenecks. In *In Proc. of CVPR*, 2018. 7

[32] R. R. Selvaraju, M. Cogswell, A. Das, R. Vedantam, D. Parikh, and D. Batra. Grad-cam: Visual explanations from deep networks via gradient-based localization. In *In Proc. of ICCV*, 2017. 2

[33] Karen Simonyan and Andrew Zisserman. Very deep convolutional networks for large-scale image recognition. In *In Proc. of ICLR*, 2015. 2, 8

[34] K. Soomro, A. R. Zamir, and M. Shah. Ucf101: A dataset of 101 human actions classes from videos in the wild. In *CoRR abs/1212.0402*, 2012. 7

[35] C. Tan, F. Sun, T. Kong, W. Zhang, C. Yang, and C. Liu. A survey on deep transfer learning. In *In Proc. of ICANN*, 2018. 1

[36] G. Wu and S. Gong. Peer collaborative learning for online knowledge distillation. In *In Proc. of CVPR*, 2020. 2

[37] J. Yosinski, Clune J, Bengio Y, and Lipson H. How transferable are features in deep neural networks? In *In Proc. of NIPS*, 2014. 1

[38] A. Zamir, A. Sax, W. Shen, L. Guibas, J. Malik, and S. Savarese. Taskonomy: Disentangling task transfer learning. In *In Proc. of CVPR*, 2018. 1

[39] A. Zamir, A. Sax, T. Yeo, O. Kar, N. Cheerla, R. Suri, Z. Cao, J. Malik, and L. Guibas. Robust learning through cross-task consistency. In *In Proc. of CVPR*, 2020. 1

[40] Y. Zhang, T. Xiang, T. M. Hospedales, and H. Lu. Deep mutual learning. In *In Proc. of CVPR*, 2018. 2, 7

[41] B. Zhou, A. Lapedriza, A. Khosla, A. Oliva, and A. Torralba. Places: A 10 million image database for scene recognition. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 40(6):1452–1464, 2018. 5