IEEE Access
Multidisciplinary | Rapid Review | Open Access Journal

Received June 29, 2021, accepted July 9, 2021, date of publication July 26, 2021, date of current version August 3, 2021.

Digital Object Identifier 10.1109/ACCESS.2021.3099856

Teaching Yourself: A Self-Knowledge Distillation Approach to Action Recognition

DUC-QUANG VU[1], NGAN LE[2], (Member, IEEE), AND JIA-CHING WANG[1], (Senior Member, IEEE)

[1]Department of Computer Science and Information Engineering, National Central University, Taoyuan 320317, Taiwan
[2]Department of Computer Science and Computer Engineering, University of Arkansas, Fayetteville, AR 72701, USA

Corresponding author: Jia-Ching Wang (jcw@csie.ncu.edu.tw)

This work was supported in part by the Ministry of Science and Technology under Grant 110-2634-F-008-004 and Grant 110-2218-E-A49-018, and in part by the National Science Foundation under Award OIA-1946391.

ABSTRACT Knowledge distillation, which is a process of transferring complex knowledge learned by a heavy network, i.e., a teacher, to a lightweight network, i.e., a student, has emerged as an effective technique for compressing neural networks. To reduce the necessity of training a large teacher network, this paper leverages the recent self-knowledge distillation approach to train a student network progressively by distilling its own knowledge without a pre-trained teacher network. Far from the existing self-knowledge distillation methods, which mainly focus on still images, our proposed Teaching Yourself is a self-knowledge distillation technique that targets at videos for human action recognition. Our proposed Teaching Yourself is not only designed as an effective lightweight network but also a high generalization capability model. In our approach, the network is able to update itself using the best past model, termed the preceding model, which is then utilized to guide the training process to update the present model. Inspired by consistency training in state-of-the-art semi-supervised learning methods, we also introduce an effective augmentation strategy to increase data diversity and improve network generalization and consistent predictions for our proposed Teaching Yourself approach. Our benchmark has been conducted on both the 3D Resnet-18 and 3D ResNet-50 backbone networks and evaluated on various standard datasets such as UCF101, HMDB51, and Kinetics400 datasets. The experimental results have shown that our teaching yourself method significantly improves the action recognition performance in terms of accuracy compared to existing supervised learning and knowledge distillation methods. We also have conducted an expensive ablation study to demonstrate that our approach mitigates overconfident predictions on dark knowledge and generates more consistent predictions in input variations of the same data point. The code is available at https://github.com/vdquang1991/Self-KD.

INDEX TERMS Self-knowledge distillation, self-learning, knowledge distillation, action recognition, deep learning, convolutional neural network.

I. INTRODUCTION

Human action recognition is one of the most fundamental research problems in computer vision and machine learning due to its prevalence in real life. The recognized actions can be used for other tasks such as security surveillance, abnormalities detection, etc. The goal of action recognition is to identify many different actions from given video clips. This requires considering the temporal structure of input data by aggregating information from multiple frames. In which each frame is an RGB image or an optical flow image. To accomplish this goal, many deep learning models have been proposed with various architectures such as the 2D convolutional neural networks (CNNs) [1], 3D CNNs [2], and LSTM combined with the 2D CNN [1], [3]. Several methods have used more than one network (two streams) with two different inputs to increase the model's learning ability. For example, the input contains an image and an optical flow

The associate editor coordinating the review of this manuscript and approving it for publication was Joewono Widjaja.

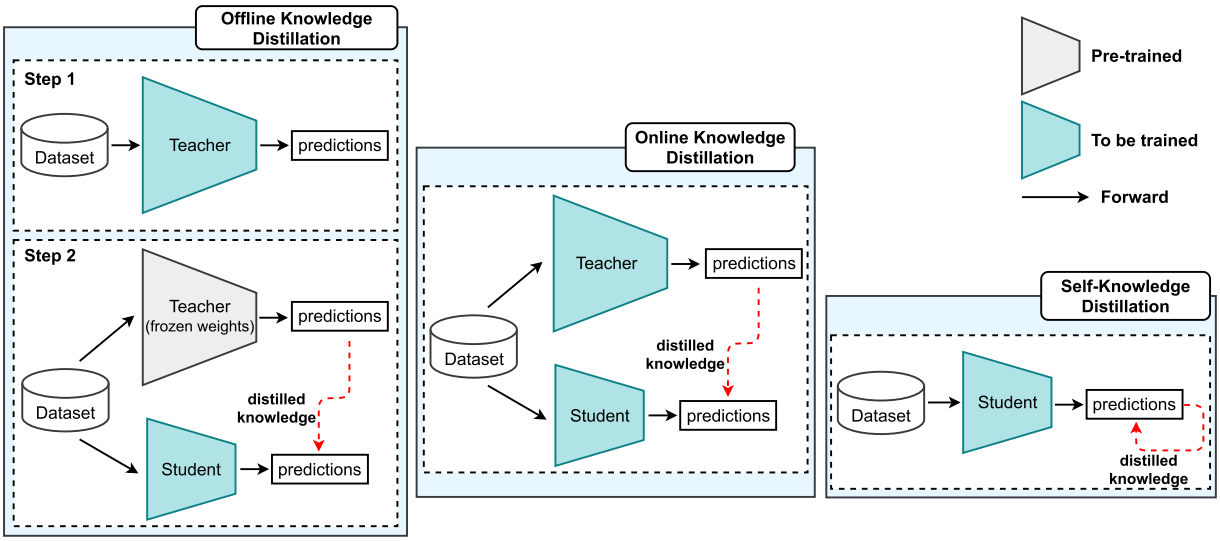


FIGURE 1. The comparison among three KD mechanisms including offline knowledge distillation (left), online KD (middle), and self-KD (right). In offline KD, the teacher presented by a large network is first offline trained on a large-scale dataset. Then, the teacher's knowledge is then transferred to a student presented by a small network. In online KD, both a large teacher network and a small student network are simultaneously trained. In self-KD, a large teacher network is no longer required. Only a small student network is defined, and it is able to distil the knowledge by itself.

clip [3], the input contains an RGB video clip and an optical flow clip [4], and the input contains an RGB slow pathway and an RGB fast pathway [5].

By combining 3D CNNs with either RGB or optical flow produces state-of-the-art (SOTA) performance, however, it has some limitations such as: (i) it requires optical flow extraction from RGB frames, which is computationally expensive and time-consuming; (ii) these aforementioned networks require a large number of model parameters that need to be learned, which could make the model more prone to overfitting. Despite the fact that 3D CNN networks with either RGB or optical flow are powerful models and achieve appealing results on many tasks, they are too large to be deployed on edge devices like smartphones or embedded sensor nodes. Thus, such approaches may not be suitable for real-world applications.

To address the above limitations, knowledge distillation (KD) has become a promising approach because of its ability to transfer a heavy network's interpretation capability to other lighter architectures without reducing the performance capability. Given a heavy and powerful network (i.e., the *teacher*), conventional distillation approaches [6] encouraged to transfer the teacher's knowledge to a lightweight student network by minimizing the differences between two outputs from two networks. The student network can effectively inherit the knowledge of the teacher network. Various KD approaches have been proposed to transfer different knowledge such as feature magnitude [7], feature flow [8], activation maps [9], gradient maps [10], and other factors [11]. Particularly in action recognition, KD methods have proposed to transfer knowledge from different domains such as from the image domain to the video domain [12], [13] or from optical flow models to RGB models [14].

Based on the distillation scheme, KD approaches can be divided into two main categories: offline knowledge distillation (Offline-KD), online knowledge distillation (Online-KD). The Offline-KD [6], [7], [12], [13] usually contains two successive stages, namely (i) training a heavy teacher network, (ii) the teacher network is then frozen and utilized to extract the knowledge to guide the student network (as shown in Figure. 1 left). However, a big capacity gap between a large pre-trained network (teacher) and a smaller network (student) is one of the notable limitations in the Offline-KD approach. As shown in [15], [16], the student network performance degrades when the gap between student and teacher is large. Furthermore, a two-stage training process in Offline-KD will increase both training cost and pipeline complexity. Unlike Offline-KD, Online-KD [14], [17], [18] is a one-phase end-to-end training scheme. In Online-KD, both the teacher and student networks are simultaneously updated (as shown in Figure. 1 middle). Although Online-KD is able to address the Offline-KD's limitations with a flexible teacher, it still requires a teacher network, and the teacher network needs to be trained. Recently, Self-knowledge distillation (Self-KD) [19]–[21], which is a special case of Online-KD, has been proposed to remove the heavy and expensive teacher network. In Self-KD, the student is learned and updated without any teacher (as illustrated in Figure. 1 right).

Many methods based on Self-KD have been proposed and achieved SOTA performance on image and natural language processing tasks in recent years [19]–[22]. For example, in [19], the authors present an approach to distil the predictive

distribution between different samples of the same label during training. Ji *et al.* [21] utilizes an auxiliary self-teacher network to transfer refined knowledge for the classifier network. Kim *et al.* [22] used the previous network as the pseudo-teacher to guide the current network. Far apart from the existing Self-KD methods on images, our approach is inspired by the consistent training [23], [24]. We explore the relationship between the preceding network and the present network in our approach. We utilize the best-performing student network at past epochs to distil knowledge to itself, i.e., the current student network during the training process. Moreover, a robust and suitable data augmentation strategy is proposed for the video domain to generate the heavily distorted versions of a given video clip. This helps increase data diversity, improve network generalization, and generate more consistent predictions between the preceding network and the present network in input variations of the same video clip. Finally, we provide a summary of state-of-the-art methods on different distillation schemes for both image and video domains and highlight the differences of our proposed method with the other KD methods in Table. 1.

TABLE 1. A summary of several common methods on different distillation schemes for both the image and video domain.

Distillation Scheme	Method	Input modality	Comments
Offline-KD	KD [6]	images	Relying on the last layer output to transfer knowledge.
	Fitnets [7]	images	Distillation via intermediate layers.
	TAKD [15]	images	Using teacher assistant to improve the network performance.
	STC [13]	videos	Distillation from pre-trained 2D CNNs to 3D CNNs.
	T3D [12]	videos	Distillation from pre-trained 2D CNNs to 3D CNNs.
	MERS [14]	videos	Distillation from flow networks to RGB networks.
Online-KD	OKDDip [17]	images	Distillation with diverse peers.
	DML [18]	images	Mutual learning via an ensemble of students.
	MARS [14]	videos	Distillation from flow networks to RGB networks.
Self-KD	CS-KD [19]	images	Distillation the predictive distribution between different samples of the same label.
	FRSKD [21]	images	Utilizing an auxiliary self-teacher network.
	Self-KD [22]	images	Using the previous network as the pseudo teacher.
	Ours	**videos**	**The best-performing preceding model is utilized to guide the present model.**

Our contribution is summarized as follows:

(1) Far from the existing Self-KD methods [19]–[22] which target at still images, our teaching yourself is the first Self-KD framework which aims to address human action recognition.

(2) In our proposed self-knowledge distillation method, the predictive distributions of the preceding student are utilized to guide the present student for ''dark knowledge'' (i.e., the knowledge on incorrect predictions) via the Kullback–Leibler Divergence (KLD) loss. This forces the network at the current epoch (assume at epoch n) to produce consistent predictions for dark knowledge similar to the preceding model in the past (i.e., the best performing model from epoch 1 to n-1). Furthermore, it avoids overconfident predictions and prevents the present model from being worse than the best-performing model trained in the past epochs.

(3) Due to the difference between image and video domains, thus not every data augmentation strategy that is good for images is suitable for videos. In this paper, we introduce a robust and suitable set of data augmentations for videos. This increases the diversity of data in the training set and improves the generalization performance of the network. Combining with our proposed self-knowledge distillation method, we found that our approach prevents the model's overconfident predictions and generates more consistent predictions in input variations of the same data point.

(4) Experimental results show that our approach has significantly improved accuracy compared to the supervised learning baselines regardless of the backbone networks. Moreover, our method outperforms the SOTA methods, including self-supervised [25]–[31], supervised learning [4], [32]–[35], and KD approaches [12]–[14].

The remaining of the paper is organized as follows: Section II provides a review on various action recognition approaches. The proposed Teaching Yourself approach is described in Section III. The experimental results, comparisons and ablation studies are presented in Section IV and the conclusion is given in Section V.

II. RELATED WORK

Action recognition has always been one of the most important topics in computer vision. The traditional methods proposed to solve this problem are based on efficient spatio-temporal feature representations and motion propagation across frames in videos such as the HOG3D [36], SIFT3D [37], ESURF [38], MBH [39], iDTs [40], and so on.

A popular approach today is to use CNNs [1]–[3]. In [4], the authors presented five popular strategies to design a CNN for action recognition, including LSTM [1], 3D-ConvNet (or 3D CNN) [2], Two-Stream [3], 3D-Fused Two-Stream [41], Two-Stream 3D-ConvNet [4]. In this paper, we focus on 3D CNNs, which outperforms 2D CNNs in large-scale video datasets [2]. Due to kernel size being 3D, i.e., (t, s, s) where t and s denote temporal and spatial kernel size, these models can be used to extract spatio-temporal features from raw videos [35] directly. This section introduces three main types of action recognition approaches, including

supervised learning, self-supervised learning, and knowledge distillation.

A. SUPERVISED LEARNING

The models in supervised learning are built to predict classes based on annotated data. Tran *et al.* [2] proposed a simple linear model named C3D and found a $3 \times 3 \times 3$ convolutional kernel to work best among a limited set of explored 3D CNN architectures. Instead of using only RGB frames, Simonyan *et al.* [3] proposed a two-stream network where one stream contains RGB images and the other contains optical flow images. The I3D [4] is a new approach to transform a 2D pre-trained network into a 3D network. In the I3D, the 3D filters are replaced by a set of repeated 2D filters. Inspired by the success of the ResNet in the image classification, Hara *et al.* [35] extended the ResNet architecture to a 3D CNN and examined the architectures of various 3D CNNs including the ResNet-18, ResNet-34, ResNet-50, ResNet-101, etc.

Following the success of the C3D and I3D networks, many new architectures have been proposed in recent years. For example, a new method has been proposed to explicitly factorize a 3D convolution into two separate and successive operations, namely a 2D spatial convolution and a 1D temporal convolution referred to as the (2+1)D convolution ([33], [42]). Recently, the SlowFast network [43] is a variation of the 3D CNN networks category. Two parallel pathways are utilized to explicitly capture the appearances and object motion in a video. Instead of using one stream for RGB, and the other for optical flow, the SlowFast network utilizes RGB for all streams. A slow pathway operates at a low frame rate, capturing spatial semantics, and a fast pathway captures motion with fine temporal resolution at a high frame rate. The SlowFast network has been proposed to tackle the action recognition and action spatial localization tasks. It has achieved the highest scores in many benchmark datasets, such as Kinetics, Charades, AVA, etc.

B. SELF-SUPERVISED LEARNING

Self-supervised learning aims at learning visual features from unlabeled data in pretext tasks. The learned visual representation model in the pretext task is then transferred to the downstream task. The objective of self-supervised learning focuses on extracting good feature representations without annotation; thus, it targets designing an effective pretext task component. Inspired by frame reordering tasks, Dahun *et al.* [28] shown that ambiguity in time direction when we hardly distinguish between a ''catch'' or a ''throw'' action from given shuffled frames. The authors introduced a self-supervised task called Space-Time cubic puzzles. Given a randomly permuted sequence of 3D spatio-temporal pieces cropped from a video clip. The 3D CNN is used to learn both spatial and temporal relations from the input video frames and predict their original arrangement.

Different from the above method, a model based on deep reinforcement learning was introduced in [44]. The authors observed that there had been unused potential in self-supervision based on ordering. The diverse permutations will affect CNN differently. How can we find permutations of higher utility to improve a CNN representation than the random set? The authors presented a reinforcement learning algorithm that helps to create permutations in the training phase. To learn the function for proposing permutations, the authors simultaneously train a policy and self-supervised network by utilizing the CNN network's improvement over time as a reward signal. Jiangliu *et al.* [30] presented a self-supervised learning method by predicting motion and appearance statistics. Each video frame is divided into several spatial regions. Then a model predicts the region with the largest motion and its direction. Vu *et al.* [45] introduced seven different transformations for videos, and a model is trained to predict which transformations are applied to the input video.

C. KNOWLEDGE DISTILLATION

This method is first introduced by Hinton *et al.* [6]. The method helps the student model (a small network) learn knowledge from teacher models (a large network). In [12], the authors proposed a new model named 'Temporal 3D ConvNet' (T3D). In this model, 3D dense blocks and Temporal Transition Layers (TTL) are arranged alternately. The TTL layers use kernels with different sizes for temporal dimensions to increase the model's ability to learn temporal features. Additionally, the T3D model uses knowledge transferred from a pre-trained 2D ConvNet (DenseNet-169) on ImageNet. Like T3D, Diba *et al.* [13] proposed Spatio-Temporal Channel Correlation (STC) model based on ResNet architecture, and the authors also used the teacher models are 2D ResNet and ResNext pre-trained on ImageNet. The main contribution of this method is to propose STC blocks alternating 3D Residual blocks. The STC block behaves similarly to the squeeze and excitation block in [46]. Girdhar *et al.* [47] proposed a distillation model based on ResNet architecture. In their model, ResNet50 pre-trained on image datasets as ''teachers'' to train video models in a distillation framework. This is a new approach for learning spatiotemporal representations from unlabeled video data.

III. PROPOSED METHOD

In this section, we first introduce an overview of our proposed Teaching Yourself for action recognition. We then discuss the training paradigm, the loss function, network architecture, and data augmentation in our approach.

A. TEACHING YOURSELF

As mentioned in Section I, Self-KD is proposed to overcome offline and online distillation limitations such as avoiding the high-capacity teacher and reducing the capacity gap between large teachers and small students. Different from the existing

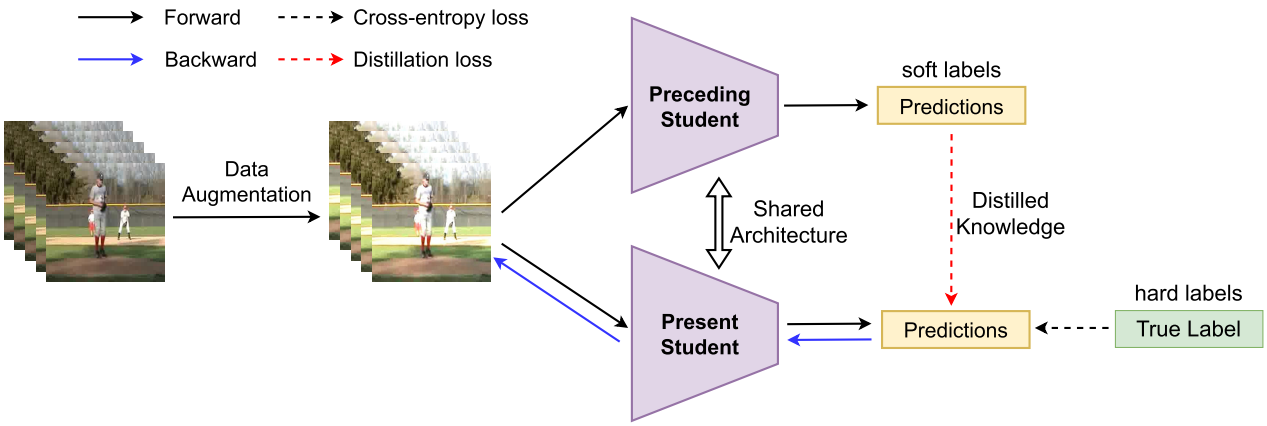


FIGURE 2. Overview of our proposed Teaching Yourself approach for action recognition where the preceding student as the pseudo-teacher denotes the best performing model checkpoint from epoch 1 to epoch n-1, guides training the present student via the last output layer, and the present student denotes the training model at the current epoch (epoch n).

Self-KD methods [19]–[21] for image and natural language processing tasks, in our work, we utilize the model at the past epochs (i.e., preceding model) to guide the model at the current epoch (i.e., present model) during the training phase. Our proposed Teaching Yourself (TY) method is arguably the simplest way to save time in the training process, minimize the competency gap between teacher and student, and preventing overconfident predictions. The proposed method is illustrated in Figure. 2.

1) SAVING TIME IN THE TRAINING PROCESS

Due to no teacher network in our approach, so the best performing model checkpoint at the past epochs (i.e., the preceding student network) is utilized as the pseudo teacher to "teach" the model at the current epoch (i.e., the present student network). Therefore our approach doesn't require calculating the loss function, performing the backpropagation, and updating the weights for the teacher network. As a result, the proposed method saves a lot of processing time during the training phase.

2) MINIMIZING THE COMPETENCY GAP BETWEEN TEACHER AND STUDENT

In conventional knowledge distillation, including offline and online distillation, the complex high-capacity teacher networks are sometimes ineffective for the student models. Mirzadeh *et al.* [15] have shown that the student network performance degrades when the gap between student and teacher is large. In our approach, the teacher and student networks are shared the same architecture; hence, there is no capacity gap between the student and teacher in the proposed TY. As a result, our approach addresses the high-capacity teacher issue. Moreover, the teacher network in our approach is not a fixed model with weights but dynamically evolves as the training proceed.

3) PREVENTING OVERCONFIDENT PREDICTIONS

To avoid this issue, the TY approach utilizes the predictions of the past model as the soft-labels, and a rich set of data augmentations is proposed suitable for the video domain and increases the diversity of data in the training set. Furthermore, our proposed method exploits the relationship between the preceding student network and the present student network) via two predictive distributions of both networks. This forces the network at the current epoch to provide more consistent predictions in the input variations of the same data point by minimizing the distance between the two logits within the same sample.

Far apart from the Siamese networks [23], [24] that usually uses a weight-sharing neural network to maximize the similarity object in two or more inputs, our proposed method uses only one input and maximizes the similarity between two networks (preceding and present networks). Our approach is the combination of the Self-KD method and the data augmentation strategy for video. Given the input video clip, we randomly apply data augmentation operators to the video clip, such as color distortion, noise addition, Gaussian blur, contrast and brightness adjustments, etc. The transformed video clip is then passed into two student models (including the preceding student and the present student) with the same network architecture. During the training phase, the present student model is the model checkpoint at the current epoch. The preceding student is the model checkpoint with the best performance for several past epochs on the validation set. The predictive distributions of the preceding student network are utilized as the knowledge distilled to guide the present student network.

B. TRAINING PARADIGM

Given a set of N training samples is denote as $\{(x_i, y_i)\}_{i=1}^{N}$ where x_i is a video clip and y_i is label (i.e., the action). Note

that y_i is a K-dimensional one-hot vector where K is the numbers of classes. We assume that $\mathcal{F}^T$ and $\mathcal{F}^S$ correspond to the preceding student (pseudo-teacher) and present student models. Let $x'_i = \mathcal{G}(x_i)$ denote the transformed video from the original video x_i where $\mathcal{G}(\cdot)$ is the set of transformation operators. Let $\mathbf{z}^T(x'_i) = \mathcal{F}^T(x'_i; \theta^T)$ and $\mathbf{z}^S(x'_i) = \mathcal{F}^S(x'_i; \theta^S)$ denote the output of the preceding student model and the present student model, respectively. $\mathbf{z}(x'_i)$ represents the logit vector $[z_1(x'), z_2(x'), \ldots, z_K(x')]$ where $z_k(x'_i)$ is the logit value for the k^{th} class (with $k = 1, 2, \ldots, K$); θ^T and θ^S correspond to the set of parameters for the preceding student and the present student models with only θ^S being the set of trainable parameters, and θ^T being the set of frozen weights during the backpropagation process. Let $\mathbf{p}(x'_i) = [p_1(x'_i), \ldots, p_K(x'_i)]$ denote the probability that the input belongs to the K classes can be estimated by a softmax function given by:

$$p_k(x'_i) = \frac{\exp(z_k(x'_i))}{\sum_{j=1}^{K} \exp(z_j(x'_i))} \tag{1}$$

The predictions of the teacher models' soft targets contain dark knowledge, i.e., the knowledge for the wrong predictions. They can be used as supervisors to transfer knowledge to the student. Hinton *et al.* [6] suggest utilizing a temperature factor τ to scale the probabilities:

$$\widetilde{p_k}(x'_i, \tau) = \frac{\exp(z_k(x'_i)/\tau)}{\sum_{j=1}^{K} \exp(z_j(x'_i)/\tau)} \tag{2}$$

where the temperature factor τ is introduced to control the importance of each soft target. The distillation loss is defined to match two logits between the preceding student model and the present student model by:

$$\mathcal{L}_{\mathrm{KL}}(\widetilde{\mathbf{p}}^T | \widetilde{\mathbf{p}}^S; \tau, \theta^S) = \sum_{i=1}^{N} \widetilde{\mathbf{p}}^T(x'_i) \log\left(\frac{\widetilde{\mathbf{p}}^T(x'_i)}{\widetilde{\mathbf{p}}^S(x'_i)}\right) \tag{3}$$

In Eq. 3, $\mathcal{L}_{\mathrm{KL}}$ denotes the Kullback-Leibler divergence loss, $\widetilde{\mathbf{p}}^T(x'_i)$ and $\widetilde{\mathbf{p}}^S(x'_i)$ are the prediction probabilities of the soft targets by the preceding student and the present student, respectively. The cross-entropy (CE) loss between the ground truth y and the present student model's predictions is defined by:

$$\mathcal{L}_{\mathrm{CE}}(y_i | \mathbf{p}^S; \theta^S) = -\sum_{i=1}^{N} y_i \log(y_i, \mathbf{p}^S(x'_i)) \tag{4}$$

where $\mathbf{p}^S(x'_i)$ is the predictions probability of the student model. From Eq. 3 and Eq. 4, the total training loss $\mathcal{L}_{\mathrm{Self-KD}}$ for the proposed method is defined as:

$$\mathcal{L}_{\mathrm{Self-KD}}(y_i, \widetilde{\mathbf{p}}^T, \widetilde{\mathbf{p}}^S, \mathbf{p}^S, \tau, \theta^S) = \mathcal{L}_{\mathrm{CE}}(y_i | \mathbf{p}^S; \theta^S) + \lambda \mathcal{L}_{\mathrm{KL}}(\widetilde{\mathbf{p}}^T | \widetilde{\mathbf{p}}^S; \tau, \theta^S) \tag{5}$$

The loss $\mathcal{L}_{\mathrm{Self-KD}}$ is then backpropagated to optimize the whole framework. In Eq. 5, λ a hyper-parameter representing the weight of $\mathcal{L}_{\mathrm{KL}}$. The detailed procedure of the self-knowledge distillation in the training phase is illustrated in Algorithm 1, where each data augmentation operator has an execution probability of 0.5 ($p = 0.5$).

Algorithm 1: Teaching Yourself: Self-KD for Action Recognition

Input: $(\mathcal{F}^S, \theta^S)$: The present student model where θ^S is the set of trainable parameters.
$(\mathcal{F}^T, \theta^T)$: The preceding student model where θ^T is the set of frozen parameters.
N: numbers of epochs.
λ, τ: loss weight and temperature factor.
Output: Return $\theta^\star$ is optimal weights

for *epoch* $= 1..N$ **do**
  for *batch in training set* **do**
    Sample a batch (x, y) from the training set
    $x = \text{Resize(size}= (-1 : 128))(x)$
    $x = \text{Random\_Crop(size}= (112, 112))(x)$
    $x' = \text{Flip}(p = 0.5)(x)$
    $x' = \text{Adjust\_Contrast}(p = 0.5)(x')$
    $x' = \text{Adjust\_Brightness}(p = 0.5)(x')$
    $x' = \text{Adjust\_Hue}(p = 0.5)(x')$
    $x' = \text{Gaussian\_Blur}(p = 0.5)(x')$
    $x' = \text{Channel\_Splitting}(p = 0.5)(x')$
    $x' = \text{Add\_Noise}(p = 0.5)(x')$
    $\mathbf{z}^T = \mathcal{F}^T(x')$
    $\mathbf{z}^S = \mathcal{F}^S(x')$
    $\widetilde{\mathbf{p}}^T = softmax(\mathbf{z}^T, \tau)$
    $\widetilde{\mathbf{p}}^S = softmax(\mathbf{z}^S, \tau)$
    $\mathbf{p}^S = softmax(\mathbf{z}^S)$
    Calculate the loss function $\mathcal{L}_{\mathrm{Self-KD}}(y, \widetilde{\mathbf{p}}^T, \widetilde{\mathbf{p}}^S, \mathbf{p}^S, \tau, \theta^S)$
    Calculate the backpropagation and update θ^S
  if *Val_Accuracy*$(\mathcal{F}^S, \theta^S) >$ *Val_Accuracy*$(\mathcal{F}^T, \theta^T)$ **then**
    $\theta^T = \theta^\star = \theta^S$
  if θ^S *has converged* **then**
    Early Stopping and return $\theta^\star$

C. DATA AUGMENTATION

Data augmentation is an important component in training deep learning models. Data augmentation aims to generate new training data by applying transformations to an example without changing its label. There are many data augmentation strategies proposed for images, such as RandAugment [48], Data Augmentation for Object Detection [49]. However, data augmentation strategies for videos are still minimal. By experiments, we found several good data augmentation operators for images, but they are not good for videos, especially for action recognition such as ShearX. In other words, not every operator that is good for images is suitable for video. This section introduces a simple yet robust data augmentation

TABLE 2. The architecture of the two networks including 3D ResNet-18 and 3D ResNet-50. For the 3D ResNet-18 network, C_2, C_3, C_4, C_5 are 64, 128, 256, and 512, respectively. For the 3D ResNet-50 network, C_2, C_3, C_4, C_5 corresponds to 256, 512, 1024, and 2048.

Layer	3D ResNet-18	3D ResNet-50	Output size
Input			$T \times 112 \times 112 \times 3$
Conv 1	$7 \times 7 \times 7, 64$ stride $= 1 \times 2 \times 2$		$T \times 56 \times 56 \times 64$
Conv block 2	$\begin{bmatrix} 3 \times 3 \times 3, 64 \\ 3 \times 3 \times 3, 64 \end{bmatrix} \times 2$	$\begin{bmatrix} 1 \times 1 \times 1, 64 \\ 3 \times 3 \times 3, 64 \\ 1 \times 1 \times 1, 256 \end{bmatrix} \times 3$	$T \times 56 \times 56 \times C_2$
Conv block 3	$\begin{bmatrix} 3 \times 3 \times 3, 128 \\ 3 \times 3 \times 3, 128 \end{bmatrix} \times 2$	$\begin{bmatrix} 1 \times 1 \times 1, 128 \\ 3 \times 3 \times 3, 128 \\ 1 \times 1 \times 1, 512 \end{bmatrix} \times 4$	$\frac{T}{2} \times 28 \times 28 \times C_3$
Conv block 4	$\begin{bmatrix} 3 \times 3 \times 3, 256 \\ 3 \times 3 \times 3, 256 \end{bmatrix} \times 2$	$\begin{bmatrix} 1 \times 1 \times 1, 256 \\ 3 \times 3 \times 3, 256 \\ 1 \times 1 \times 1, 1024 \end{bmatrix} \times 6$	$\frac{T}{4} \times 14 \times 14 \times C_4$
Conv block 5	$\begin{bmatrix} 3 \times 3 \times 3, 512 \\ 3 \times 3 \times 3, 512 \end{bmatrix} \times 2$	$\begin{bmatrix} 1 \times 1 \times 1, 512 \\ 3 \times 3 \times 3, 512 \\ 1 \times 1 \times 1, 2048 \end{bmatrix} \times 3$	$\frac{T}{8} \times 7 \times 7 \times C_5$
Average Pooling	$\frac{T}{8} \times C_5$		
FC	K classes		

strategy for the video domain that includes temporal and spatial augmentations.

1) TEMPORAL AUGMENTATION

is a sampling perspective [50]. Typically, the model input is the short video clips with T frames of length (e.g. $T = \{8, 16, 32, 64, \ldots\}$). There are two common ways to sample the data. In the first sampling method, a clip with T continuous frames is trimmed from the raw video and used to represents the entire video [2]. In the second sampling method, a sequence of short snippets sparsely sampled from the entire video represents the input video [51]. We adopt the first method in our work. A clip of length T is extracted with a randomly selected starting frame from the input video. If the length of a video is less than T frames, the last frame is repeated.

2) SPATIAL AUGMENTATION

is a temporally consistent design. Spatial augmentation has been being widely adopted for images in many tasks such as image classification, object detection, object segmentation, etc. Unlike the images, a video is a continuous sequence of frames where each frame is an image; thus, we cannot use existing image-based augmentation operators on the frames individually, such as vertically flip for the first frame, adjust contrast for the second frame, etc. This could negatively affect the motion features across the frames. Therefore, we design a simple method to address this issue. Instead of utilizing augmentation operators individually on the frames, we consistently apply the spatial augmentations across the entire input video clip. Given an input clip of length T, our data augmentation strategy takes two steps. We first scale the shorter edge of the frames in the clip to 128 and the other edge is calculated to maintain the frame original aspect ratio. A random cropping window with dimensions of 112×112 is generated and applied to all of the frames. We then randomly choose several data augmentation operators for the clip, i.e., perform consistently for all frames in the clip. The list of all augmentation operators used in our work includes flip, contrast adjustment, brightness adjustment, hue adjustment, Gaussian blur, channel splitting, and noise addition. The probability chosen for each augmentation operator is set to 0.5. For the channel splitting operator, we randomly select one of the three-channel (RGB channel) and then replace it with another (e.g., RGR or GGB or RBB, etc.). For noise addition, we create a random noise matrix using a Gaussian distribution with the mean as 0 and the standard deviation as 0.1. The noise matrix is added to all of the frames in the input clip. The other augmentation operators such as flip, contrast adjustment, brightness adjustment, hue adjustment, Gaussian blur follow the concepts from the image domain.

D. NETWORK ARCHITECTURE

ResNet [52] is one of the most popular architectures in computer vision. The ResNet significantly increases the performance of many image-related tasks such as classification, detection, and segmentation. The 3D ResNet is an extension of the ResNet for videos [35]. In our work, we consider two SOTA CNN architectures including the 3D ResNet-18, and the 3D ResNet-50 in [35]. The details of the two networks are described in Table 2 where FC represents a fully connected

layer. Both networks are mainly composed of a stack of Conv layers grouped into four blocks. The stride value is set to 2 for the first convolution layer in each block (except the first Conv block). Each convolution layer in the networks is followed by a BatchNormalization layer and a ReLU layer.

IV. EXPERIMENT

In this section, we first introduce the datasets used in this work and our implementation in detail. We then discuss ablation studies, visualization, and performance comparison with different benchmarks.

A. DATASETS AND IMPLEMENTATION

We have conducted experiments on three standard datasets including HMDB51 [53], UCF101 [54], and Kinetics400 [55].

1) UCF101

is an action recognition dataset, including 101 action categories. All videos from this dataset are real action videos collected from YouTube. UCF101 gives diversity in terms of actions, with 13,320 videos containing large variations in camera motion, object appearance and pose, object scale, viewpoint, cluttered background, illumination conditions, etc. The average duration of each video is about 7 seconds. Three train/test splits (70% training and 30% testing) are provided in this dataset.

2) HMDB51

is a small dataset collected from various sources, mostly from movies, and a small proportion from public databases such as the Prelinger archive, YouTube, and Google videos. The dataset contains 6,849 clips divided into 51 action categories, each containing a minimum of 101 clips. Three train/test splits (70% training and 30% testing) are also provided in this dataset.

3) KINETICS400

(K400) is a large dataset that has 400 human action classes [55]. Kinetics is a standard benchmark for action recognition in videos. The videos in this dataset were temporally trimmed and last around 10 seconds and 200–1000 clips for each action. The total has 306,245 videos in Kinetics400. The number of training, validation, and testing sets are about 240,000, 20,000, and 40,000, respectively.

4) IMPLEMENTATION DETAILS

Both 3D ResNet-18 and 3D ResNet-50 are trained from scratch and optimized by stochastic gradient descent (SGD) with a momentum of 0.9 and an initial learning rate of 0.01. The weight decay is set to 5×10^{-4}. The model input is a video clip with 16 frames ($T = 16$), each frame has $112 \times 112 \times 3$ of dimension and normalized to be [-1, 1]. We use the mini-batch of 16 clips per GPU, and training is done in 200 epochs. The learning rate is dropped by $10\times$ after 10 epochs if the validation accuracy not improving.

B. PERFORMANCE COMPARISON

In this section, we conduct the comparison with different benchmarks as follows:

- Compare with independently training (including with and without data augmentation).
- Compare with the KD action recognition approaches with both 3D ResNet-18 and 3D ResNet-50.
- Compare with the SOTA action recognition approaches including supervised, self-supervised approaches under both 3D ResNet-18 and 3D ResNet-50.

1) COMPARE WITH INDEPENDENTLY TRAINING

To examine the effectiveness of the proposed TY method, we have compared our approach to the baseline method (independently training) with cross-entropy loss on the standard datasets and standard learning paradigm settings using the 3D ResNet-18 and 3D ResNet-50. As can be seen in Table. 4, our approach significantly improves the accuracies for both the large and small-scale datasets. Specifically, our method increases the accuracies by 21.9% and 11.9% for the 3D ResNet-18 and 3D ResNet-50 networks, respectively, on the UCF101 dataset. For the HMDB51 dataset, our approach increases the accuracies by 12.6% and 10.8% for the 3D ResNet-18 and 3D ResNet-50 networks, respectively. In the large-scale dataset, i.e., the Kinetics400 dataset, the TY method increases the accuracies by 11.1% and 12.2% for the 3D ResNet-18 and 3D ResNet-50 networks, respectively. From these results, we confirm that our approach can significantly improve generalization compared to independently training only with hard labels on regardless of the backbone networks.

We compare in detail the TY with the baseline method combined with data augmentation to demonstrate the effectiveness of the proposed self-knowledge distillation method combines with our data augmentation strategy. The results of the comparisons are given in Table. 3 and Figure. 3.

TABLE 3. Performance as a function of epochs. All experiments use the 3D ResNet-18 network and are trained on the UCF101 dataset.

Method	Clip@1	Video@1
Baseline	44.2	46.5
Baseline + Data Aug	54.5	57.8
TY (ours)	**64.3**	**68.4**

Table. 3 shows that the proposed data augmentation method significantly improves the model's performance. Specifically, the performance of the baseline method with data augmentation (Baseline + Data Aug) is better than the baseline method without data augmentation by up to 10.3% in terms of clip accuracy and 11.3% in terms of video accuracy. In comparison to Baseline + Data Aug method, the TY method achieves improvements of 9.8%, and 10.6% for clip and video accuracies, respectively.

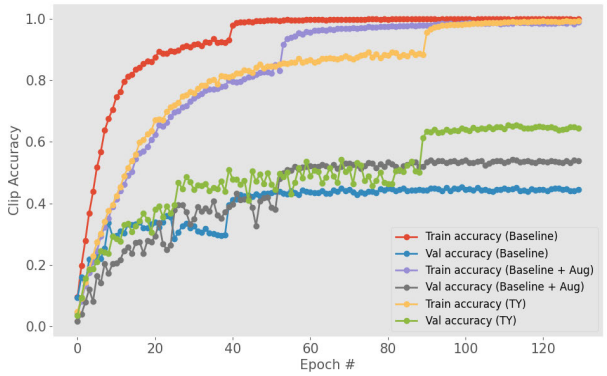


FIGURE 3. Performance as a function of epochs. All experiments use the 3D ResNet-18 network and are trained on the UCF101 dataset.

TABLE 4. The performance of the TY compares to the baseline method in both the 3D ResNet-18 and the 3D ResNet-50 network architectures on the UCF101 and HMDB51 datasets. All networks are trained from scratch.

Method	Backbone	UCF101	HMDB51	Kinetics400
Baseline	3D ResNet-18	46.5	17.1	54.2
TY (ours)	3D ResNet-18	**68.4**	**29.7**	**65.3**
Baseline	3D ResNet-50	59.2	22.0	61.3
TY (ours)	3D ResNet-50	**71.1**	**32.8**	**73.5**

As shown in Figure. 3, the convergence is different between methods. Specifically, the Baseline method has converged after epoch 40, while the Baseline + Data Aug method has converged after epoch 55, and the TY method has converged after epoch 80. The reason for the difference in the convergence speed of the methods is due to the diversity of data augmentation and the variation in the loss function. Data augmentation is the simplest approach to generate more diverse videos in the training set, fit the model on more training data, and reduce the model variance. Moreover, the loss function of our method is more complex, it requires minimizing two losses, including the CE loss and KL loss, so it affects the convergence speed of the TY. As a result, overfitting occurs more slowly than the baseline with and without data augmentation.

2) COMPARISON WITH OTHER KD MECHANISMS

Table 5 compares our TY method with other training mechanisms. As expected, the student performance in distillation approaches actually improves compared to independently training. However, the training cost of KD approaches also increases due to the training of two networks (teacher and student). Unlike KD approaches, our proposed TY method doesn't require training the teacher networks so our approach reduces a lot of time training (decrease 10.1 hours compared to Teacher to Student method and 2.8 hours compared to Student to Student method). This is extremely useful when training deep models on large datasets such as the Kinetics400. Besides, the combination between our TY method and data augmentation strategy significantly improves the generalization ability of the single network without additional models.

TABLE 5. Comparison with several distillation mechanisms on the UCF101 dataset. The student network in all mechanisms is the 3D ResNet-18 network. The column "Model size" denotes the number of parameters in each method. The "Training time" indicates the total training time of each approach. T and S denote teacher and student, respectively. All mechanisms are trained on the same batch size, GPU, and PC.

Mechanism	Teacher Network	Model size	Top-1 Accuracy	Training time (in hours)
Independently training	None	33.4M	46.5	29.8
Teacher to Student	3D ResNet-50	33.4M (S) + 63.7M (T)	55.4	41.6
Student to Student	3D ResNet-18	33.4M (S) + 33.4M (T)	52.8	34.3
Multi-Teachers	3D ResNet-34 3D ResNet-50	33.4M (S) + 46.9M (T1) + 63.7M (T2)	56.5	50.4
Flow-to-RGB	3D ResNet-18	33.4M (S) + 33.4M (T)	54.1	35.8
TY	itself	33.4M	**68.4**	31.5

3) SMALL-SCALE DATASET EXPERIMENT

We present our experimental studies of the TY method for small-scale datasets. We compare the TY method against the SOTA self-supervised learning methods on standard benchmarks such as the UCF101 and HMDB51 datasets.

As shown in Table 6, all results are top-1 accuracy in action recognition on two standard datasets. The results in the table are grouped into three categories.

+ The first category includes the accuracies of the various networks (C3D, 3D ResNet-18, and 3D ResNet-50)

TABLE 6. Top-1 test accuracy (%) of the TY method against state-of-the-art self-supervised learning methods on the UCF101 and HMDB51 datasets. The best performing model is indicated as bold.

Method	Backbone	Pretraining dataset	UCF101	HMDB51
Random Init	3D ResNet-18	None	46.5	17.1
Random Init	C3D	None	45.9	19.7
Random Init	3D ResNet-50	None	59.2	22.0
TCE [25]	3D ResNet-18	K400	68.8	34.2
DPC [26]	3D ResNet-18	K400	68.2	**34.5**
VCP [27]	3D ResNet-18	UCF101	66.0	31.5
VCP [27]	C3D	UCF101	68.5	32.5
VCP [27]	(2+1)D ResNet-18	UCF101	66.3	32.2
3D Cubic Puzzles [28]	3D ResNet-18	K400	65.8	33.7
Video Clip Ordering [29]	3D ResNet-18	UCF101	64.9	29.5
Motion & Appearance [30]	C3D	K400	61.2	33.4
Video Jigsaw [31]	C3D	K400	55.4	27.0
TY (ours)	3D ResNet-18	None	68.4	29.7
TY (ours)	3D ResNet-50	None	**71.1**	32.8

trained from scratch (random-initialization weights) on the UCF101 and HMDB51 datasets.

+ The second category shows the accuracies of the SOTA self-supervised methods with various backbone networks. All self-supervised methods have been pre-trained on different pretext tasks with different pre-training datasets. After that, these models are fine-tuned for action recognition via supervised learning.

+ The third category shows our method's performance on the 3D ResNet-18 and 3D ResNet-50 networks. All networks are trained from scratch on the UCF101 and HMDB51 datasets.

Our proposed method obtains the SOTA performance in terms of accuracy on the UCF101 dataset compared to the existing approaches with various backbone networks and different pretext tasks. In particular, when compared to the several methods that use the Kinetics dataset in the pretext task, the proposed TY method outperforms 2.3% to TCE [25], 2.9% to DPC [26] and 5.3% to 3D Cubic Puzzles [28] in the UCF101 dataset. For the HMDB51 dataset, our accuracy is lower than several self-supervised methods such as TCE [25], DPC [26], 3D Cubic Puzzles [28]. However, these methods are pre-trained on large-scale datasets such as the Kinetics400 dataset; meanwhile, our method is trained from scratch.

4) LARGE-SCALE DATASET EXPERIMENT

Since TY has shown its effectiveness for training various networks on these small datasets, in this part, we scale up the TY method to train on a large dataset, i.e., the Kinetics400 dataset. We then compare the TY method to the SOTA methods, including the supervised learning and knowledge distillation methods.

Table. 7 shows top-1 accuracy results in action recognition, which are split into two groups. The top group shows the performances of the supervised learning-based methods, and the bottom group shows the performances of the knowledge distillation-based methods. Several methods pre-trained on large-scale datasets have either ImageNet or Sport1M entered under ''Pretraining dataset''.

As shown in Table. 7, the proposed TY method has obtained higher accuracy while utilizing fewer frames and lower frame resolutions. In particular, the TY method uses RGB frames without the optical flow, which significantly reduces the cost of optical flow calculation and model training on the optical flow domain. Compared to the SOTA supervised learning methods (the top group), the TY method achieves performances that are on par with the bLVNet [34] (73.5%) with fewer frames (16 vs. 24) and lower frame resolution (112 vs. 224). Moreover, our method outperforms the I3D [4] and R(2+1)D [33] by 2.4% and 0.3%, respectively, even though these methods are pre-trained on large-scale datasets such as the ImageNet and Sport-1M. Our proposed TY has obtained a better performance with a shallower model compared to other knowledge distillation methods. Specifically, even if the STC, T3D and MARS methods utilize deep neural networks such as the ResNeXt-101 and DenseNet-169, the TY method still outperforms the STC [13], T3D [12], MARS [14] by 4.8%, 11.3%, and 4.6% respectively. It implies that the TY method is also effective on the large-scale dataset. It further demonstrates that our approach drives the single network to learn more spatio-temporal features and improve the generalization capability. So, compared to the conventional distillation methods, our proposed TY method has better generality and extensibility for any network backbone and dataset.

TABLE 7. Top-1 test accuracy (%) of the TY method on the Kinetics400 dataset compared to the SOTA methods, including supervised learning and knowledge distillation. # frames denotes the number of frames used in the models. * indicates that these methods use both RGB and optical flow frames in the training phase. The best performing model is indicated as bold.

Method	Backbone	Pretraining dataset	#frames	Frame resolution	Top-1 Accuracy
I3D [4]	InceptionNet	ImageNet	64	224×224	71.1
FASTER [32]	R(2+1)D-50	None	32	224×224	71.7
R(2+1)D* [33]	ResNet-34	Sport-1M	32	112×112	73.3
bLVNet [34]	ResNet-50	None	24	224×224	73.5
R3D [35]	ResNet-152	None	16	112×112	63.0
R3D [35]	ResNeXt-101	None	16	112×112	65.1
STC [13]	ResNeXt-101	ImageNet	32	112×112	68.7
T3D [12]	DenseNet-169	ImageNet	32	224×224	62.2
MARS* [14]	ResNeXt-101	None	16	112×112	68.9
TY (ours)	ResNet-18	None	16	112×112	65.3
TY (ours)	ResNet-50	None	16	112×112	**73.5**

TABLE 8. The effect of two main hyper-parameters τ and λ on our approach. All experiments use the 3D ResNet-18 network and are trained from scratch on the UCF101 dataset. The best performing model is indicated as bold while the worst model is indicated as underline.

τ	λ		
	0.1	0.2	0.5
5	66.1	**68.4**	65.1
10	67.1	66.9	65.3
20	66.3	63.3	<u>46.1</u>

C. ABLATION STUDY

To examine the effect of the main hyper-parameters τ and λ, we conduct a detailed ablation study in this section. Specifically, we test the hyper-parameters across an array of $\tau \in \{5, 10, 20\}$ and $\lambda \in \{0.1, 0.2, 0.5\}$ on the 3D ResNet-18 using the UCF101 dataset. The results are presented in Table. 8. Except for the hyper-parameters τ and λ under consideration, all of the other settings are the same as in Algorithm 1. Overall, we have found that our method is fairly robust with respect to τ and λ, except for the extreme case where the large values of $\tau = 20$ and $\lambda = 0.5$.

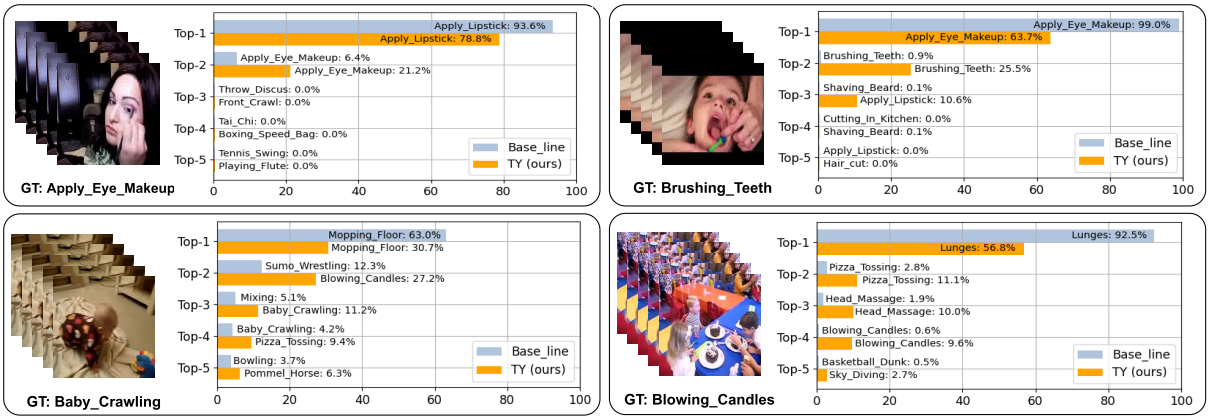


FIGURE 4. Probability distributions (top-5 softmax scores) on misclassified samples between our proposed TY method and baseline method from the UCF101 dataset. All samples are tested on the 3D ResNet-18 network. GT denotes the ground-truth labels of these videos.

D. VISUALIZATION FOR OVERCONFIDENT PREDICTION

In this section, we examine whether the proposed method generates more meaningful predictions and mitigates overconfident predictions. To this end, we investigate prediction values in softmax scores from the 3D ResNet-18 network, i.e., $\mathbf{p}(x)$. Specifically, we analyze the predictions of two methods, namely the Baseline and the proposed TY methods, with four concrete misclassified samples in the UCF101 dataset in Figure. 4.

As shown in Figure. 4, the baseline method outputs overconfident prediction on misclassification. In contrast, the TY method relaxes the overconfident predictions and provides the class probabilities distributed over the classes with similar visual characteristics. For example, with the sample clip labeled "Brushing_Teeth" (top right), the Baseline method provides the predictive distribution of 99.0% for "Apply_Eye_Makeup" class; meanwhile, the "Brushing_Teeth" class has the only 0.9%. Unlike the Baseline method, the TY method generates a predictive distribution of 63.7% for the "Apply_Eye_Makeup" class and 25.5% for the "Brushing_Teeth" class. This demonstrates that the TY method mitigates overconfident predictions and generates more consistent predictions by forcing the networks to produce similar predictions despite the input variations of the same data point.

V. CONCLUSION

In this work, we have introduced a simple yet effective self-knowledge distillation that utilizes knowledge distilled from itself to enhance the generalization capability. Our approach utilized the preceding model (best performing model in the past) as a pseudo-teacher to guide the model at the current epoch during the training phase. We have also introduced a robust and effective data augmentation method for the video domain. Combining Self-knowledge distillation and data augmentation, we have demonstrated that our approach mitigates overconfident predictions, provides more consistent predictions in input variations of the same data point and enhances the generalization performance of deep neural networks without additional networks. Experiments conducted across different network architectures have shown that our proposed method achieves state-of-the-art performance compared to supervised learning, self-supervised learning, and knowledge distillation methods on both small-scale and large-scale datasets. In addition, to action recognition, the proposed method can be adapted and applied to other tasks involving the generalization and calibration of neural networks.

REFERENCES

[1] A. Karpathy, G. Toderici, S. Shetty, T. Leung, R. Sukthankar, and L. Fei-Fei, "Large-scale video classification with convolutional neural networks," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, Jun. 2014, pp. 1725–1732.

[2] D. Tran, L. Bourdev, R. Fergus, L. Torresani, and M. Paluri, "Learning spatiotemporal features with 3D convolutional networks," in *Proc. IEEE Int. Conf. Comput. Vis. (ICCV)*, Dec. 2015, pp. 4489–4497.

[3] K. Simonyan and A. Zisserman, "Two-stream convolutional networks for action recognition in videos," in *Proc. NIPS*, 2014, pp. 568–576.

[4] J. Carreira and A. Zisserman, "Quo vadis, action recognition? A new model and the kinetics dataset," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jul. 2017, pp. 6299–6308.

[5] C. Feichtenhofer, H. Fan, J. Malik, and K. He, "SlowFast networks for video recognition," in *Proc. IEEE/CVF Int. Conf. Comput. Vis. (ICCV)*, Oct. 2019, pp. 6202–6211.

[6] E. G. Hinton, O. Vinyals, and J. Dean, "Distilling the knowledge in a neural network," *CoRR*, vol. abs/1503.02531, pp. 1–9, Mar. 2015.

[7] A. Romero, N. Ballas, S. E. Kahou, A. Chassang, C. Gatta, and Y. Bengio, "FitNets: Hints for thin deep nets," in *Proc. ICLR*, 2015.

[8] J. Yim, D. Joo, J. Bae, and J. Kim, "A gift from knowledge distillation: Fast optimization, network minimization and transfer learning," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jul. 2017, pp. 4133–4141.

[9] S. Zagoruyko and N. Komodakis, "Paying more attention to attention: Improving the performance of convolutional neural networks via attention transfer," in *Proc. ICLR*, 2017.

[10] T. Guo, C. Xu, S. He, B. Shi, C. Xu, and D. Tao, "Robust student network learning," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 31, no. 7, pp. 2455–2468, Aug. 2019.

[11] S. Srinivas and F. Fleuret, "Knowledge transfer with Jacobian matching," in *Proc. ICML*, 2018, vol. 80, pp. 4730–4738.

[12] A. Diba, M. Fayyaz, V. Sharma, A. H. Karami, M. M. Arzani, R. Yousefzadeh, and L. Van Gool, "Temporal 3D ConvNets: New architecture and transfer learning for video classification," 2017, *arXiv:1711.08200*. [Online]. Available: http://arxiv.org/abs/1711.08200

[13] A. Diba, M. Fayyaz, V. Sharma, M. M. Arzani, R. Yousefzadeh, J. Gall, and L. V. Gool, "Spatio-temporal channel correlation networks for action classification," in *Proc. ECCV*, 2018, pp. 284–299.

[14] N. Crasto, P. Weinzaepfel, K. Alahari, and C. Schmid, "MARS: Motion-augmented RGB stream for action recognition," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2019, pp. 7882–7891.

[15] S. I. Mirzadeh, M. Farajtabar, A. Li, N. Levine, A. Matsukawa, and H. Ghasemzadeh, "Improved knowledge distillation via teacher assistant," in *Proc. AAAI Conf. Artif. Intell.*, vol. 34, 2020, pp. 5191–5198.

[16] J. Gou, B. Yu, S. J. Maybank, and D. Tao, "Knowledge distillation: A survey," 2020, *arXiv:2006.05525*. [Online]. Available: http://arxiv.org/abs/2006.05525

[17] D. Chen, J.-P. Mei, C. Wang, Y. Feng, and C. Chen, "Online knowledge distillation with diverse peers," in *Proc. AAAI Conf. Artif. Intell.*, 2020, vol. 34, pp. 3430–3437.

[18] Y. Zhang, T. Xiang, T. M. Hospedales, and H. Lu, "Deep mutual learning," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, Jun. 2018, pp. 4320–4328.

[19] S. Yun, J. Park, K. Lee, and J. Shin, "Regularizing class-wise predictions via self-knowledge distillation," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2020, pp. 13876–13885.

[20] S. Hahn and H. Choi, "Self-knowledge distillation in natural language processing," 2019, *arXiv:1908.01851*. [Online]. Available: http://arxiv.org/abs/1908.01851

[21] M. Ji, S. Shin, S. Hwang, G. Park, and I.-C. Moon, "Refine myself by teaching myself: Feature refinement via self-knowledge distillation," 2021, *arXiv:2103.08273*. [Online]. Available: http://arxiv.org/abs/2103.08273

[22] K. Kim, B. Ji, D. Yoon, and S. Hwang, "Self-knowledge distillation with progressive refinement of targets," 2020, *arXiv:2006.12000*. [Online]. Available: http://arxiv.org/abs/2006.12000

[23] Q. Xie, Z. Dai, E. Hovy, M.-T. Luong, and Q. V. Le, "Unsupervised data augmentation for consistency training," 2019, *arXiv:1904.12848*. [Online]. Available: http://arxiv.org/abs/1904.12848

[24] K. Sohn, D. Berthelot, C.-L. Li, Z. Zhang, N. Carlini, E. D. Cubuk, A. Kurakin, H. Zhang, and C. Raffel, "FixMatch: Simplifying semi-supervised learning with consistency and confidence," 2020, *arXiv:2001.07685*. [Online]. Available: http://arxiv.org/abs/2001.07685

[25] J. Knights, B. Harwood, D. Ward, A. Vanderkop, O. Mackenzie-Ross, and P. Moghadam, "Temporally coherent embeddings for self-supervised video representation learning," 2020, *arXiv:2004.02753*. [Online]. Available: http://arxiv.org/abs/2004.02753

[26] T. Han, W. Xie, and A. Zisserman, "Video representation learning by dense predictive coding," in *Proc. IEEE/CVF Int. Conf. Comput. Vis. Workshop (ICCVW)*, Oct. 2019, pp. 1483–1492.

[27] D. Luo, C. Liu, Y. Zhou, D. Yang, C. Ma, Q. Ye, and W. Wang, "Video cloze procedure for self-supervised spatio-temporal learning," in *Proc. AAAI Conf. Artif. Intell.*, 2020, vol. 34, pp. 11701–11708.

[28] D. Kim, D. Cho, and I. S. Kweon, "Self-supervised video representation learning with space-time cubic puzzles," in *Proc. AAAI Conf. Artif. Intell.*, 2019, vol. 33, pp. 8545–8552.

[29] D. Xu, J. Xiao, Z. Zhao, J. Shao, D. Xie, and Y. Zhuang, "Self-supervised spatiotemporal learning via video clip order prediction," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2019, pp. 10334–10343.

[30] J. Wang, J. Jiao, L. Bao, S. He, Y. Liu, and W. Liu, "Self-supervised spatio-temporal representation learning for videos by predicting motion and appearance statistics," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2019, pp. 4006–4015.

[31] U. Ahsan, R. Madhok, and I. Essa, "Video jigsaw: Unsupervised learning of spatiotemporal context for video action recognition," in *Proc. IEEE Winter Conf. Appl. Comput. Vis. (WACV)*, Jan. 2019, pp. 179–189.

[32] L. Zhu, D. Tran, L. Sevilla-Lara, Y. Yang, M. Feiszli, and H. Wang, "Faster recurrent networks for efficient video classification," in *Proc. AAAI Conf. Artif. Intell.*, 2020, vol. 34, pp. 13098–13105.

[33] D. Tran, H. Wang, L. Torresani, J. Ray, Y. LeCun, and M. Paluri, "A closer look at spatiotemporal convolutions for action recognition," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, Jun. 2018, pp. 6450–6459.

[34] Q. Fan, C.-F. Chen, H. Kuehne, M. Pistoia, and D. Cox, "More is less: Learning efficient video representations by big-little network and depth-wise temporal aggregation," in *Proc. Adv. Neural Inf. Process. Syst.*, vol. 32, H. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alché-Buc, E. Fox, and R. Garnett, Eds. Red Hook, NY, USA: Curran Associates, 2019.

[35] K. Hara, H. Kataoka, and Y. Satoh, "Can spatiotemporal 3D CNNs retrace the history of 2D CNNs and ImageNet?" in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, Jun. 2018, pp. 6546–6555.

[36] A. Klaeser, M. Marszałek, and C. Schmid, "A spatio-temporal descriptor based on 3D-gradients," in *Proc. Brit. Mach. Vis. Conf.*, 2008, pp. 1–275.

[37] P. Scovanner, S. Ali, and M. Shah, "A 3-dimensional sift descriptor and its application to action recognition," in *Proc. 15th Int. Conf. Multimedia (MULTIMEDIA)*, 2007, pp. 357–360.

[38] G. Willems, T. Tuytelaars, and L. V. Gool, "An efficient dense and scale-invariant spatio-temporal interest point detector," in *Proc. ECCV*. Berlin, Germany: Springer, 2008, pp. 650–663.

[39] N. Dalal, B. Triggs, and C. Schmid, "Human detection using oriented histograms of flow and appearance," in *Proc. ECCV*. Berlin, Germany: Springer, 2006, pp. 428–441.

[40] H. Wang and C. Schmid, "Action recognition with improved trajectories," in *Proc. IEEE Int. Conf. Comput. Vis.*, Dec. 2013, pp. 3551–3558.

[41] C. Feichtenhofer, A. Pinz, and A. Zisserman, "Convolutional two-stream network fusion for video action recognition," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2016, pp. 1933–1941.

[42] Z. Qiu, T. Yao, and T. Mei, "Learning spatio-temporal representation with pseudo-3D residual networks," in *Proc. IEEE Int. Conf. Comput. Vis. (ICCV)*, Oct. 2017, pp. 5533–5541.

[43] C. Feichtenhofer, H. Fan, J. Malik, and K. He, "SlowFast networks for video recognition," in *Proc. IEEE/CVF Int. Conf. Comput. Vis. (ICCV)*, Oct. 2019, pp. 6201–6210.

[44] U. Buchler, B. Brattoli, and B. Ommer, "Improving spatiotemporal self-supervision by deep reinforcement learning," in *Proc. ECCV*, 2018, pp. 770–786.

[45] D. Q. Vu, N. T. H. Le, and J.-C. Wang, "Self-supervised learning via multi-transformation classification for action recognition," 2021, *arXiv:2102.10378*. [Online]. Available: http://arxiv.org/abs/2102.10378

[46] J. Hu, L. Shen, and G. Sun, "Squeeze-and-excitation networks," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, Jun. 2018, pp. 7132–7141.

[47] R. Girdhar, D. Tran, L. Torresani, and D. Ramanan, " DistInit: Learning video representations without a single labeled video," in *Proc. IEEE/CVF Int. Conf. Comput. Vis. (ICCV)*, Oct. 2019, pp. 852–861.

[48] E. D. Cubuk, B. Zoph, J. Shlens, and Q. V. Le, "Randaugment: Practical automated data augmentation with a reduced search space," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. Workshops (CVPRW)*, Jun. 2020, pp. 702–703.

[49] B. Zoph, E. D. Cubuk, G. Ghiasi, T.-Y. Lin, J. Shlens, and Q. V. Le, "Learning data augmentation strategies for object detection," in *Proc. Eur. Conf. Comput. Vis.* Cham, Switzerland: Springer, 2020, pp. 566–583.

[50] R. Qian, T. Meng, B. Gong, M.-H. Yang, H. Wang, S. Belongie, and Y. Cui, "Spatiotemporal contrastive video representation learning," 2020, *arXiv:2008.03800*. [Online]. Available: http://arxiv.org/abs/2008.03800

[51] L. Wang, Y. Xiong, Z. Wang, Y. Qiao, D. Lin, X. Tang, and L. V. Gool, "Temporal segment networks: Towards good practices for deep action recognition," in *Proc. Eur. Conf. Comput. Vis.* Cham, Switzerland: Springer, 2016, pp. 20–36.

[52] 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.

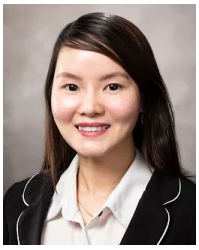
[53] H. Kuehne, H. Jhuang, E. Garrote, T. Poggio, and T. Serre, "HMDB: A large video database for human motion recognition," in *Proc. Int. Conf. Comput. Vis.*, Nov. 2011, pp. 2556–2563.

[54] K. Soomro, A. R. Zamir, and M. Shah, "A dataset of 101 human action classes from videos in the wild," *Center Res. Comput. Vis.*, vol. 2, no. 11, pp. 1–7, Nov. 2012.

[55] 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," 2017, *arXiv:1705.06950*. [Online]. Available: http://arxiv.org/abs/1705.06950

DUC-QUANG VU was born in Nam Dinh city, Nam Dinh, Vietnam, in 1991. He received the B.S. degree in education in information technology from Thai Nguyen University of Education, Vietnam, in 2013, and the M.S. degree in information system from the University of Engineering and Technology, Vietnam National University, Hanoi (VNU), in 2016. He is currently pursuing the Ph.D. degree with the Department of Computer Science and Information Engineering, National Central University, Taiwan.

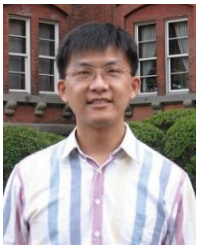
His research interests include the machine learning, deep learning, computer vision, and bioinformatics. He received several awards and honors, including the 2nd in National Informatics Olympiad for Universities, Vietnam, in 2009, the 3rd in National Informatics Olympiad for Universities, Vietnam, in 2011, the 4th in Young Scientist Talent Contest for Universities, Vietnam, in 2012, the Certificate of Achievement in the ACM-ICPC 2010 Asia Hanoi Regional Contest, and the Certificate of Achievement in the 2011 ACM-ICPC Vietnam National Programming Contest.



NGAN LE (Member, IEEE) received the bachelor's and master's degrees in CS from the University of Science, Vietnam, in 2005 and 2009, respectively, and the master's and Ph.D. degrees in ECE from Carnegie Mellon University (CMU), in 2015 and 2018, respectively.

From 2018 to 2019, she was a Research Associate with the Department of Electrical and Computer Engineering (ECE), CMU. She is currently an Assistant Professor with the Department of Computer Science and Computer Engineering, University of Arkansas. Her publications appear in top conferences, including CVPR, MICCAI, ICCV, SPIE, IJCV, and ICIP, and premier journals, including *IJCV*, *JESA*, IEEE TRANSACTIONS ON IMAGE PROCESSING, *PR*, *JDSP*, and IEEE TRANSACTIONS ON INFORMATION FORENSICS AND SECURITY. She has coauthored more than 55 journal articles, conference papers, and book chapters, more than 7 patents and inventions. Her past research interests include biometrics, compressed sensing, image processing, data hiding, watermarking, document analysis, and handwriting recognition. Her current research interests include image understanding, video understanding, computer vision, robotics, machine learning, deep learning, reinforcement learning, biomedical imaging, and single cell-RNA.

Dr. Le is also a Guest Editor of *Scene Understanding in Autonomous* (Frontier) and *Artificial Intelligence in Biomedicine and Healthcare* (MDPI). She co-organized the Deep Reinforcement Learning Tutorial for Medical Imaging at MICCAI 2018, Medical Image Learning with Less Labels and Imperfect Data workshop at MICCAI 2019, 2020. She has served as a Reviewer for more than ten top-tier conferences and journals, including IEEE TRANSACTIONS ON PATTERN ANALYSIS AND MACHINE INTELLIGENCE, AAAI, CVPR, NIPS, ICCV, ECCV, MICCAI, IEEE TRANSACTIONS ON IMAGE PROCESSING, *PR*, *TAI*, and *IVC*.



JIA-CHING WANG (Senior Member, IEEE) received the Ph.D. degree in electrical engineering from the National Cheng Kung University, Taiwan, in 2002.

He was an Honorary Fellow with the Department of Electrical and Computer Engineering, University of Wisconsin-Madison, in 2008 and 2009. He is currently a Professor with the Department of Computer Science and Information Engineering, National Central University, Taiwan. His research interests include signal processing, deep learning, machine learning, and VLSI architecture design. He is an Honorary Member of the Phi Tau Phi Scholastic Honor Society and a member of ACM and IEICE.