Self-Supervised Video Representation Learning with Constrained Spatiotemporal Jigsaw

Yuqi Huo[1,2] , **Mingyu Ding**[3] , **Haoyu Lu**[1] , **Ziyuan Huang**[4] ,
Mingqian Tang[5] , **Zhiwu Lu**[2*] and **Tao Xiang**[6]
[1]School of Information, Renmin University of China, Beijing, China
[2]Gaoling School of Artificial Intelligence, Renmin University of China, Beijing, China
[3]The University of Hong Kong, Pokfulam, Hong Kong, China
[4]National University of Singapore, Singapore
[5]Alibaba Group, Hangzhou, China
[6]University of Surrey, Surrey, UK
luzhiwu@ruc.edu.cn

Abstract

This paper proposes a novel pretext task for self-supervised video representation learning by exploiting spatiotemporal continuity in videos. It is motivated by the fact that videos are spatiotemporal by nature and a representation learned by detecting spatiotemporal continuity/discontinuity is thus beneficial for downstream video content analysis tasks. A natural choice of such a pretext task is to construct spatiotemporal (3D) jigsaw puzzles and learn to solve them. However, as we demonstrate in the experiments, this task turns out to be intractable. We thus propose Constrained Spatiotemporal Jigsaw (CSJ) whereby the 3D jigsaws are formed in a constrained manner to ensure that large continuous spatiotemporal cuboids exist. This provides sufficient cues for the model to reason about the continuity. Instead of solving them directly, which could still be extremely hard, we carefully design four surrogate tasks that are more solvable. The four tasks aim to learn representations sensitive to spatiotemporal continuity at both the local and global levels. Extensive experiments show that our CSJ achieves state-of-the-art on various benchmarks.

1 Introduction

Self-supervised learning (SSL) has achieved tremendous successes recently for static images [He *et al.*, 2020; Chen *et al.*, 2020] and shown to be able to outperform supervised learning on a wide range of downstream image understanding tasks. However, such successes have not yet been reproduced for videos. Since different SSL models differ mostly on the pretext tasks employed on the unlabeled training data, designing pretext tasks more suitable for videos is the current focus for self-supervised video representation learning [Han *et al.*, 2020a; Wang *et al.*, 2020].

Spatiotemporal analysis is the key to many video content understanding tasks. A good video representation learned from the self-supervised pretext task should capture discriminative information jointly along both spatial and temporal dimensions. It is thus somewhat counter-intuitive to note that most existing SSL pretext tasks for videos do not explicitly require joint spatiotemporal video understanding. For example, some spatial pretext tasks have been borrowed from images without any modification [Jing *et al.*, 2018], ignoring the temporal dimension. On the other hand, many recent video-specific pretext tasks typically involve speed or temporal order prediction [Lee *et al.*, 2017; Benaim *et al.*, 2020], *i.e.*, operating predominately along the temporal axis.

A natural choice for a spatiotemporal pretext task is to solve 3D jigsaw puzzles, whose 2D counterpart has been successfully used for images [Noroozi and Favaro, 2016]. Indeed, solving 3D puzzles requires the learned model to understand spatiotemporal continuity, a key step towards video content understanding. However, directly solving a 3D puzzle turns out to be intractable: a puzzle of $3\times3\times3$ pieces (the same size as a Rubik's cube) can have 27! possible permutations. Video volume even in a short clip is much larger than that. Nevertheless, the latest neural sorting model [Paumard *et al.*, 2020; Du *et al.*, 2020] can only handle permutations a few orders of magnitude less, so offer no solution. This is hardly surprising because such a task is daunting even for humans: Most people would struggle with a standard Rubik's cube, let alone a much larger one.

In this paper, we propose a novel Constrained Spatiotemporal Jigsaw (CSJ) pretext task for self-supervised video representation learning. The key idea is to form 3D jigsaw puzzles in a constrained manner so that it becomes solvable. This is achieved by factorizing the permutations into the three spatiotemporal dimensions and then applying them sequentially. This ensures that for a given video clip, large continuous spatiotemporal cuboids exist after the constrained shuffling to provide sufficient cues for the model to reason about spatiotemporal continuity (see Figure 1(b)(c)). Such large continuous cuboids are also vital for human understanding of video as revealed in neuroscience and visual studies [Stringer

[*]Corresponding author.

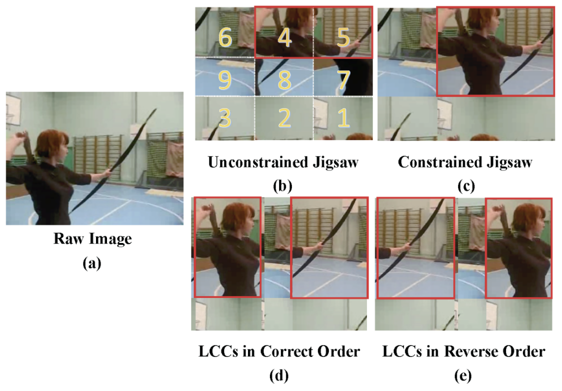

Figure 1: Illustration of our constrained jigsaw using an image example (only spatial for clarity). (a): The raw image. (b),(c): Comparing an unconstrained puzzle (b) and our constrained one (c), it is clear that ours is much more continuous (hence interpretable) reflected by the size of the largest continuous cuboids (LCCs, rectangles in images here) shown in red. (d),(e): Illustration of the importance of the relative order of the top-2 LCCs for determining the global continuity level of the shuffled image. (d) and (e) have the same top-2 LCCs, but only (d) keeps the correct relative order between them.

et al., 2006; Chen *et al.*, 2019]. Even with the constrained puzzles, solving them directly could still be extremely hard. Consequently, instead of directly solving the puzzles (*i.e.*, recovering the permutation matrix so that each piece can be put back), four surrogate tasks are carefully designed. They are more solvable but meanwhile still ensure that the learned representation is sensitive to spatiotemporal continuity at both the local and global levels. Concretely, given a video clip shuffled with our constrained permutations, we make sure that the top-2 largest continuous cuboids (LCCs) dominate the clip volume. The level of continuity in the shuffle clip as a whole is thus determined mainly by the volumes of these LCCs, and whether they are at the right order (see Fig. 1(d)(e)) both spatially and temporally. Our surrogate tasks are thus designed to locate these LCCs and predict their order so that the model learned with these tasks can be sensitive to spatiotemporal continuity both locally and globally.

Our main contributions are three-fold: (1) We introduce a new pretext task for self-supervised video representation learning called Constrained Spatiotemporal Jigsaw (CSJ). To our best knowledge, this is the first work on self-supervised video representation learning that leverages spatiotemporal jigsaw understanding. (2) We propose a novel constrained shuffling method to construct easy 3D jigsaws containing large LCCs. Four surrogate tasks are then formulated in place of the original jigsaw solving tasks. They are much more solvable yet remain effective in learning spatiotemporal discriminative representations. (3) Extensive experiments show that our approach achieves state-of-the-art on two downstream tasks across various benchmarks.

2 Related Work

Self-supervised Learning with Pretext Tasks. According to the transformations used by the pretext task, existing SSL methods for video presentation learning can be divided into three groups: **(1) Spatial-Only Transformations**: Derived from the original image domain [Gidaris *et al.*, 2018], [Jing *et al.*, 2018] leveraged the spatial-only transformations for self-supervised video presentation learning. **(2) Temporal-Only Transformations**: [Misra *et al.*, 2016; Lee *et al.*, 2017] obtained shuffled video frames with the temporal-only transformations and then distinguished whether the shuffled frames are in chronological order. [Xu *et al.*, 2019] chose to shuffle video clips instead of frames. [Benaim *et al.*, 2020; Yao *et al.*, 2020] exploited the speed transformation via determining whether one video clip is accelerated. **(3) Spatiotemporal Transformations**: There are only a few recent approaches [Ahsan *et al.*, 2019; Kim *et al.*, 2019] that leveraged both spatial and temporal transformations by permuting 3D spatiotemporal cuboids.

However, due to the aforementioned intractability of solving the spatiotemporal jigsaw puzzles, they only leveraged either temporal or spatial permutations as training signals, *i.e.*, the crops are extracted from a 4-cell grid of shape 2x2x1 or 1x1x4. Therefore, *no true spatiotemporal permutations* have been considered in [Ahsan *et al.*, 2019; Kim *et al.*, 2019]. In contrast, given that both spatial appearances and temporal relations are important cues for video representation learning, the focus of this work is on investigating how to exploit the 3D permutations jointly for self-supervised video presentation learning, *i.e.*, if we follow the statement in [Kim *et al.*, 2019], 16 crops are actually sampled from a grid of shape 2x2x4. To that end, our CSJ presents the first spatiotemporal continuity based pretext task for video SSL, thanks to a novel constrained 3D jigsaw and four surrogate tasks to reason about the continuity in the 3D jigsaw puzzles without solving them directly.

Self-supervised Learning with Contrastive Learning. Contrastive learning is another self-supervised learning approach that has become increasingly popular in the image domain [He *et al.*, 2020; Chen *et al.*, 2020]. Recently, it has been incorporated into video SSL as well [Han *et al.*, 2020b]. Contrastive learning and transformation based pretext tasks are orthogonal to each other and often combined in that different transformed versions of a data sample form the positive set used in contrastive learning. Recent works [Han *et al.*, 2019; 2020a; Zhuang *et al.*, 2020] leveraged features from the future frame embeddings or with the memory bank. They modeled spatiotemporal representations using only contrastive learning without transformations. Contrastive learning is also exploited in one of our surrogate pretext tasks. Different from existing works, we explore the spatiotemporal transformations in the form of CSJ and employ contrastive learning to distinguish different levels of spatiotemporal continuity in shuffled jigsaws. This enables us to learn more discriminative spatiotemporal representations.

3 Constrained Spatiotemporal Jigsaw

3.1 Problem Definition

The main goal of self-supervised video representation learning is to learn a video feature representation function $f(\cdot)$ without using any human annotations. A general approach to achieving this goal is to generate a supervisory signal y

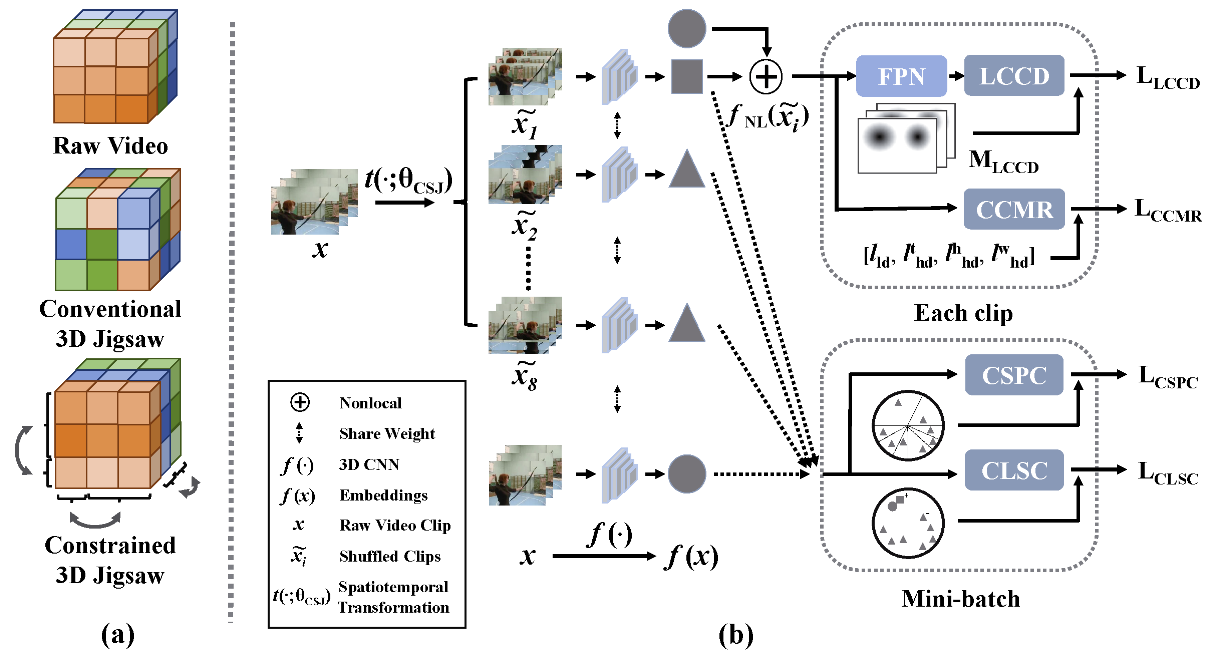

Figure 2: (a) Illustration of our Constrained Spatiotemporal Jigsaw (CSJ) (see Sec. 3.2). (b) The pipeline of our proposed framework for self-supervised video representation learning (see Sec. 3.3).

from an unlabeled video clip x and construct a pretext task P to predict y from $f(x)$. The process of solving the pretext task P encourages $f(\cdot)$ to learn discriminative spatiotemporal representations from videos.

The pretext task P is constructed by applying a transformation function $t(\cdot;\theta)$ parameterized by θ and then automatically deriving y from θ, *e.g.*, y can be the type of the transformation. Based on this premise, P is defined as the prediction of y using the feature map of the transformed video clip $f(\widetilde{x})$, *i.e.*, $P : f(\widetilde{x}) \rightarrow y$, where $\widetilde{x} = t(x;\theta)$. For example, in [Lee *et al.*, 2017], $t(\cdot;\theta)$ denotes a temporal transformation that permutes the four frames of video clip x in a temporal order θ, $\widetilde{x} = t(x;\theta)$ is the shuffled clip, and the pseudo-label y is defined as the permutation order θ (*e.g.*, 1324, 4312, etc.). The pretext task P is then a classification problem of 24 categories because there are $4! = 24$ possible orders.

3.2 Constrained Permutations

Solving spatiotemporal video jigsaw puzzles seems to be an ideal pretext task for learning discriminative representation as it requires an understanding of spatiotemporal continuity. After shuffling the pixels in a video clip using a 3D permutation matrix, the pretext task is to recover the permutation matrix. However, as explained earlier, this task is intractable given even moderate video clip sizes. Our solution is to introduce constraints on the permutations. As a result, a new pretext task P_{CSJ} based on **Constrained Spatiotemporal Jigsaw** (see Fig. 2(a)) is formulated, which is much easier to solve than a random/unconstrained jigsaw.

Specifically, our goal is to introduce constraints to the permutations so that the resultant shuffled video clip is guaranteed to have large continuous cuboids (see Fig. 2(a)). Similar to humans [Stringer *et al.*, 2006], having large continuous cuboids is key for a model to understand a 3D jigsaw and therefore to have any chance to solve it. Formally, the volume of a shuffled video clip $\widetilde{x}$ are denoted as $\{T, H, W\}$, measuring its sizes along the temporal, height, and width dimensions, respectively. A cuboid is defined as a crop of $\widetilde{x}$: $c = \widetilde{x}_{t_1:t_2,h_1:h_2,w_1:w_2}$, where $t_1, t_2 \in \{1, 2, \ldots, T\}, h_1, h_2 \in \{1, 2, \ldots, H\}, w_1, w_2 \in \{1, 2, \ldots, W\}$. If all the jigsaw pieces (smallest video clip unit, *e.g.*a pixel or a 3D pixel block) in c keep the same relative order as they were in x (before being shuffled), we call the cuboid c as a continuous cuboid c^{cont}. The cuboid's volume equals $(t_2 - t_1) \times (h_2 - h_1) \times (w_2 - w_1)$, and the largest continuous cuboid (LCC) $c^{\mathrm{cont}}_{\mathrm{max}}$ is the c^{cont} with the largest volume.

We introduce two permutation strategies to ensure that the volumes of LCCs are large in relation to the whole video clip volume after our shuffling transformation $t(\cdot;\theta_{\mathrm{CSJ}})$. First, instead of shuffling x in three spatiotemporal dimensions simultaneously, $t(\cdot;\theta_{\mathrm{CSJ}})$ factorizes the permutations into the three spatiotemporal dimensions and then utilizes them sequentially to generate shuffled clips, *e.g.*, in the order of T, W, H and only once. Note that the volume of the generated $\widetilde{x}$ stays the same with different permutation orders (*e.g.*, TWH and HTW). Second, we shuffle a group of jigsaw pieces together instead of each piece individually along each dimension.

Given that a video clip has eight frames, taking the temporal dimension as an example, we make an arbitrarily grouped permutation. θ_{CSJ} could be represented as the permutation from $\{12345678\}$ to $\{84567123\}$. The longest and the second-longest index ranges are: $[2, 5]$ for coordinates $\{4567\}$, and $[6, 8]$ for coordinates $\{123\}$. Note that we make 3 groups of frames only as an example. Since the permutations are not deterministic, in practice, different numbers of groups (*e.g.*., 3, 4, or 5) will be selected for different video clips. With these two permutation strategies, not only do we have large LCCs, but also they are guaranteed to have clearly separable boundaries (see Fig. 2(b)) with surrounding pieces due to the factorized and grouped permutation design. This means that they are easily detectable.

3.3 Surrogate Tasks

Having permutation constraints preserves more spatiotemporal continuity in the shuffled clip and reduces the amount of possible permutations. But exploiting them to make a neural sorting model tractable is still far from trivial. Instead of solving the jigsaw directly, our P_{CSJ} is thus formulated as four surrogate tasks: Largest Continuous Cuboid Detection (LCCD), Clip Shuffling Pattern Classification (CSPC), Contrastive Learning over Shuffled Clips (CLSC), and Clip Continuity Measure Regression (CCMR). As illustrated in Fig. 2(b), given an unlabeled clip x, we first construct a mini-batch of 8 clips $\{\widetilde{x}_1, \widetilde{x}_2, ..., \widetilde{x}_8\}$ by shuffling x with different but related constrained permutations (to be detailed later). These shuffled clips and the raw clip x are then fed into a 3D CNN model $f(\cdot)$ for spatiotemporal representation learning with a non-local operation [Wang *et al.*, 2018]:

$$f_{\text{NL}}(\widetilde{x}_i) = \text{NL}(f(\widetilde{x}_i), f(x)), \tag{1}$$

where $\text{NL}(\cdot, \cdot)$ denotes the non-local operator, and $f(\widetilde{x}_i)$ and $f(x)$ denote the feature map of $\widetilde{x}_i$ and x from the last convolutional layer of $f(\cdot)$, respectively. The resultant feature map $f_{\text{NL}}(\widetilde{x}_i)$ is further passed through a spatial pooling layer followed by a separately fully-connected layer for each surrogate task. Note that the raw video feature map $f(x)$ is used as guidance through the non-local based attention mechanism to help fulfill the tasks. This is similar to humans needing to see the completed jigsaw picture to help solve the puzzle.

We first explain how the eight permutations from the same raw clip are generated. First, the factorized and grouped permutations are applied to x to create one shuffled clip. By examining the largest and the second-largest continuous puzzle piece numbers of each dimension ($\{T, H, W\}$), we can easily identify the top-2 largest continuous cuboids (LCCs). Next, by varying the relative order of the top-2 LCCs either in the correct (original) order or the reverse order in each dimension, 2×2×2=8 permutations are obtained. By controlling the group size in permutation, we can make sure that the top-2 LCCs account for a large proportion, saying 80% of the total clip volume. Our four tasks are thus centered around these two LCCs as they largely determine the overall spatiotemporal continuity of the shuffled clip.

The first task **LCCD** is to locate the top-2 LCCs $\{c^{\text{cont}}_{\text{max}}(j) : j = 1, 2\}$ and formulated as a regression problem. Given a ground-truth LCC $c^{\text{cont}}_{\text{max}}(j)$, a Gaussian kernel is applied to its center to depict the possibility of each pixel in $\widetilde{x}$ belonging to the LCC. This leads to a soft mask M^j_{LCCD} with the same size of $\widetilde{x}$: M^j_{LCCD} is all 0 outside the region of $c^{\text{cont}}_{\text{max}}(j)$, and $\exp(-\frac{||a-a_{\text{c}}||^2}{2\sigma_g^2})$ inside the region, where a, a_{c} denote any pixel and the center point, respectively. σ_g is the hyper-parameter which is set as 1 empirically. In the training stage, FPN [Lin *et al.*, 2017] is used for multi-level feature fusion. LCCD is optimized using the MSE loss in each point:

$$L_{\text{LCCD}} = \sum_{j\in\{1,2\}} \sum_{a\in\widetilde{x}} \text{MSE}(M^j_{\text{LCCD}}(a), M^j_{\text{LCCD}}(a)'), \tag{2}$$

where $\text{MSE}(\cdot, \cdot)$ denotes the MSE loss function, and $M^j_{\text{LCCD}}(a)'$ is the prediction of each pixel a.

CSPC is designed to recognize the shuffling pattern of a shuffled clip. As mentioned early, the eight shuffled clips in each mini-batch are created from the same raw clip and differ only in the relative order of the top-2 LCCs along each of the three dimensions. There are thus eight permutations depending on the order (correct or reverse) in each dimension. Based on this understanding, CSPC is formulated as a multi-class classification task to recognize each shuffled clip into one of these eight classes, which is optimized using the Cross-Entropy (CE) loss:

$$L_{\text{CSPC}} = \sum_{i\in\{0,1,\ldots,7\}} \text{CE}(l_{\text{CSPC}}[i], l'_{\text{CSPC}}[i]), \tag{3}$$

where $\text{CE}(\cdot, \cdot)$ is the CE loss function and $l'_{\text{CSPC}}[i]$ is the predicted class label of i-th shuffled clip in each mini-batch.

The two tasks above emphasize on local spatiotemporal continuity understanding. In contrast, **CLSC** leverages the contrastive loss to encourage global continuity understanding. In particular, since the top-2 LCCs dominate the volume of a clip, it is safe to assume that if their relative order is correct in all three dimensions, the shuffled clip largely preserve continuity compared to the original clip, while all other 7 permutations feature large discontinuity in at least one dimension. We thus form a contrastive learning task with the original video x and the most continuous shuffled video $\widetilde{x}_i$ as a positive pair, and x and the rest $\widetilde{x}_j$ $(j \neq i)$ as negative pairs. CLSC is optimized using the Noise Contrastive Estimation (NCE) [Oord *et al.*, 2018] loss:

$$L_{\text{CLSC}} = -\log \frac{\exp(\text{sim}(f(x), f(\widetilde{x}_i))/\tau)}{\exp(\text{sim}(f(x), f(\widetilde{x}_i))/\tau) + \sum_j \exp(\text{sim}(f(x), f(\widetilde{x}_j))/\tau)}, \tag{4}$$

where $\text{sim}(\cdot, \cdot)$ is defined by the dot product: $f(x)^\top f(\widetilde{x}_i)$, and τ is the temperature hyper-parameter. Note that the non-local operator is not used in CLSC.

CCMR is similar to CLSC in that it also enforces global continuity understanding, but differs in that it is a regression task aimed at predicting a global continuity measure. We consider two such measures. Since the total size of the top-2 LCCs $\{c^{\text{cont}}_{\text{max}}(j) : j = 1, 2\}$ is a good indicator of how continuous a shuffle video clip is, the first measure l_{ld} directly measures the relative total size of the top-2 LCCs: $l_{ld} = \frac{\text{v}(c^{\text{cont}}_{\text{max}}(1))+\text{v}(c^{\text{cont}}_{\text{max}}(2))}{\text{v}(\widetilde{x})}$, where $\text{v}(\cdot)$ represents the volume of a clip/cuboid. The second measure $l^{\text{t/h/w}}_{\text{hd}}$ examines the shuffling degree of $\widetilde{x}$ in each dimension, computed as the normalized hamming distance: $\frac{\text{hamming}(\widetilde{x})}{N_c(N_c-1)/2}$, where $\text{hamming}(\cdot)$ denotes the hamming distance in each dimension between the original piece sequence and the permuted one, and N_c represents the number of pieces in each dimension so that $N_c(N_c-1)/2$ indicates the maximum possible hamming distance in the dimension. CCMR is optimized using the Mean

Squared Error (MSE) loss:

$$L_{\text{CCMR}} = \text{MSE}([l_{\text{ld}}, l^{\text{t}}_{\text{hd}}, l^{\text{h}}_{\text{hd}}, l^{\text{w}}_{\text{hd}}], [l^{'}_{\text{ld}}, l^{\text{t}'}_{\text{hd}}, l^{\text{h}'}_{\text{hd}}, l^{\text{w}'}_{\text{hd}}]), \quad (5)$$

where $l^{'}_{\text{ld}}, l^{\text{t}'}_{\text{hd}}, l^{\text{h}'}_{\text{hd}}, l^{\text{w}'}_{\text{hd}}$ are the prediction of the model.

3.4 Overall Learning Objective

Our entire CSJ framework is optimized end-to-end with the learning objective defined as:

$$L = \sigma_1 L_{\text{LCCD}} + \sigma_2 L_{\text{CSPC}} + \sigma_3 L_{\text{CLSC}} + \sigma_4 L_{\text{CCMR}}, \quad (6)$$

where $\sigma_1, \sigma_2, \sigma_3, \sigma_4$ are the weights for these losses. We deploy the adaptive weighting mechanism [Kendall *et al.*, 2018] to weight these tasks, resulting no free hyper-parameters to tune. We also adopt curriculum learning [Bengio *et al.*, 2009] to train our network by shuffling clips from easy to hard.

4 Experiments

4.1 Datasets and Settings

We select three benchmark datasets for performance evaluation: UCF101 [Soomro *et al.*, 2012], HMDB51 [Kuehne *et al.*, 2011], and Kinetics-400 (K400) [Kay *et al.*, 2017], containing 13K/7K/306K video clips from 101/51/400 action classes, respectively. In the self-supervised pre-training stage, we utilize the first training split of UCF101/HMDB51 and the training split of K400 without using their labels. As in [Han *et al.*, 2020a], we adopt R2D3D as the backbone network, which is modified from R3D [Hara *et al.*, 2018] with fewer parameters. By fine-tuning the pre-trained model, we can evaluate the SSL performance on a downstream task (*i.e.*, action classification). Following [Han *et al.*, 2019; He *et al.*, 2020], two evaluation protocols are used: comparisons against state-of-the-arts follow the more popular fully fine-tuning evaluation protocol, but ablation analysis takes both the linear evaluation and fully fine-tuning protocols. For the experiments on supervised learning, we report top-1 accuracy on the first test split of UCF101/HMDB51 as the standard [Han *et al.*, 2020a].

4.2 Implementation Details

Raw videos in these datasets are decoded at a frame rate of 24-30 fps. From each raw video, we start from a randomly selected frame index and sample a consecutive 16-frame video clip with a temporal stride of 4. For data augmentation, we first resize the video frames to 128×171 pixels, from which we extract random crops of size 112×112 pixels. We also apply random horizontal flipping to the video frames during training. Random color jittering is utilized to avoid learning shortcuts.We exploit *only the raw RGB video frames* as input, and do not leverage optical flow or other auxiliary signals for self-supervised pre-training.

4.3 Model Evaluations

Results of Directly Solving CSJ. We first demonstrate the results of solving the CSJ task directly in Table 1. We randomly shuffle video clips into 4 × 4 × 4 jigsaw puzzles. To recognize the correct permutation, the model solve a (4! × 4! × 4!)-way classification task in the pre-training stage.

Rand. Init.	CSJ	LCCD+CCMR (4x4x4)	LCCD+CCMR (16x28x28)
8.3	13.8	18.2	23.1

Table 1: Evaluation of pre-training tasks with the backbone R2D3D-18 under the linear evaluation protocol on UCF101. For computation efficiency, CSJ is only defined on 4 × 4 × 4 cells.

Tasks	Rand. Init.	LCCD	CSPC	CLSC	CCMR
Linear	8.3	21.8	22.6	18.9	22.7
Fully	63.6	67.8	68.1	68.1	68.1
Tasks	(CCMR)+CSPC	+CLSC	+LCCD	+AW	+CL
Linear	24.7	25.5	27.9	28.2	**28.5**
Fully	69.2	69.3	69.5	70.0	**70.4**

Table 2: Evaluation of pre-training tasks with the backbone R2D3D-18 under linear and fully fine-tuning protocols on UCF101. AW: Adaptive Weighting. CL: Curriculum Learning.

We compare the CSJ task with the joint LCCD+CCMR task under the same setting for fair comparison. Linear evaluation is adopted to show the effectiveness of different tasks. We can observe from the table that solving LCCD+CCMR jointly is more effective than solving CSJ directly.

Ablation Study. We conduct ablative experiments to validate the effectiveness of four CSJ surrogate tasks and two additional learning strategies. From Table 2, we can observe that: (1) Self-supervised learning with each of the four tasks shows better generalization than fine-tuning the network from scratch (random initialization). (2) By training over all the four tasks jointly, we can achieve large performance gains (see '+LCCD' vs. 'CCMR'). (3) Each additional learning strategy (*i.e.*, adaptive weighting or curriculum learning) leads to a small boost to the performance by 0.3-0.5%. (4) Our full model achieves a remarkable classification accuracy of 70.4%, demonstrating the effectiveness of our proposed CSJ with only the RGB video stream (without additional optical flow, audio, or text modalities).

Visualization of LCCD Predictions. We also demonstrate the visualization of the LCCD predictions from the pre-trained models in Fig. 3. We can observe that solving the LCCD task indeed enables the model to learn the locations of LCCs and understand spatiotemporal continuity, which is a key step towards video content understanding.

4.4 Main Results

Comparison in Action Recognition. A standard way to evaluate a self-supervised video representation learning model is to use it to initialize an action recognition model on a small dataset. Specifically, after self-supervised pre-training on UCF101/HMDB51/K400, we exploit the learned backbone for fully fine-tuning on UCF101 and HMDB51, following [Han *et al.*, 2020a; Wang *et al.*, 2020]. We consider one baseline: fully-supervised learning with pre-training on K400. Note that this baseline is commonly regarded as the upper bound of self-supervised representation learning [Alwassel *et al.*, 2020]. From Table 3, we have the following

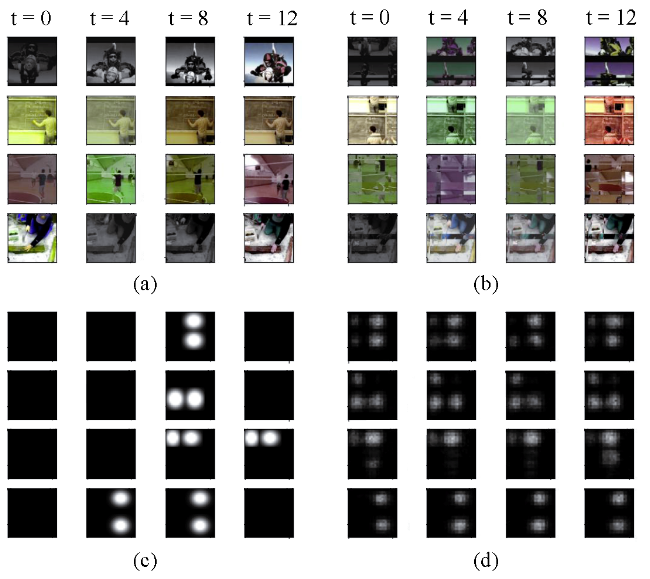

Figure 3: Visualization of the LCCD predictions from pre-trained models. Each row denotes the frames at time stamp = (0, 4, 8, 12) from one video clip. (a) raw frames (with color jittering); (b) shuffled frames; (c) ground truth of LCCD; (d) network's prediction.

Methods	**Backbone**	**Pre-trained**	**UCF101**	**HMDB51**
OPN	C2D(VGG)	U	59.8	23.8
CMC	C2D	U	55.3	–
VCOP†	R3D-18	U	64.9	29.5
VCP†	R3D-18	U	66.0	31.5
PRP†	R3D-18	U	66.5	29.7
MemDPC*	R2D3D-18	U	69.2	–
CSJ* (ours)	R2D3D-18	U	**70.4**	**36.0**
Video-Jigsaw‡	C2D	K400	55.4	27.0
Statisics*	C3D	K400	61.2	33.4
ST-Puzzle‡	R3D-18	K400	63.9	33.7
DPC*	R2D3D-18	K400	68.2	34.5
SpeedNet†	I3D	K400	66.7	43.7
VIE*	R3D-18	K400	75.5	44.6
Pace†	R(2+1)D-18	K400	**77.1**	36.6
CSJ* (ours)	R2D3D-18	K400	<u>76.2</u>	**46.7**
MemDPC*	R2D3D-34	K400	78.1	41.2
CBT	S3D	K600+HT	**79.5**	44.6
CSJ* (ours)	R2D3D-34	K400	**79.5**	**50.9**
Upper Bound:				
Fully-Supervised	R3D-34	K400	87.7	59.1

Table 3: Comparison to the state-of-the-arts on UCF101(U) and HMDB51 (only with the RGB modality). †: temporal-only transformations are used. ‡: both spatial and temporal transformations are used. ∗: spatiotemporal representations are used. HT: HowTo100M. The underline represents the second-best result.

Methods	**Top1**	**Top5**	**Top10**	**Top20**	**Top50**
VCOP	14.1	30.3	40.0	51.1	66.5
VCP	18.6	33.6	42.5	53.5	68.1
SpeedNet	13.1	28.1	37.5	49.5	65.0
PRP	**22.8**	38.5	46.7	55.2	69.1
Pace	19.9	36.2	46.1	55.6	69.2
MemDPC	20.2	40.4	52.4	64.7	–
CSJ (ours)	<u>21.5</u>	**40.5**	**53.2**	**64.9**	**70.0**

Table 4: Comparison with state-of-the-art self-supervised learning methods for nearest neighbor video retrieval (top-k recall) on UCF101. The underline represents the second-best result.

observations: (1) Our CSJ achieves state-of-the-art performance on both UCF101 and HMDB51. Particularly, with the backbone R2D3D-18 that is weaker than R(2+1)D-18, our CSJ performs comparably w.r.t. Pace on UCF101 but achieves a 10% improvement over Pace on HMDB51. (2) By exploiting spatiotemporal transformations for self-supervised representation learning, our CSJ beats either methods with only temporal transformations (†) or methods with both spatial and temporal transformations (‡), as well as those learning spatiotemporal representations (∗) via only contrastive learning (w./o. spatiotemporal transformations). (3) Our CSJ also outperforms CBT [Sun *et al.*, 2019], which used ten-times more massive datasets (K600 + Howto100M) and multiple modalities (RGB+Audio). (4) Our CSJ is the closest to the fully-supervised one (upper bound), validating its effectiveness in self-supervised video representation learning.

Comparison in Video Retrieval. We evaluate our CSJ method in the video retrieval task. Following [Xu *et al.*, 2019], we extract each video clips' embeddings with the pre-training model and use each clip in the test set to query the k nearest clips in the training set. The comparative results in Table 4 show that our method outperforms all other self-supervised methods and achieves new state-of-the-art in video retrieval on UCF101. Particularly, our method beats the latest competitor PRP [Yao *et al.*, 2020] on four out of five metrics. This indicates that our proposed CSJ is also effective for video representation learning in video retrieval.

5 Conclusion

We have introduced a novel self-supervised video representation learning method named Constrained Spatiotemporal Jigsaw (CSJ). By introducing constrained permutations, our proposed CSJ is the first to leverage spatiotemporal jigsaw in self-supervised video representation learning. We also propose four surrogate tasks based on our constrained spatiotemporal jigsaws. They are designed to encourage a video representation model to understand the spatiotemporal continuity, a key building block towards video content analysis. Extensive experiments were carried out to validate the effectiveness of each of the four CSJ tasks and also show that our approach achieves the state-of-the-art on two downstream tasks.

Acknowledgements

This work was supported in part by National Natural Science Foundation of China (61976220 and 61832017), Beijing Outstanding Young Scientist Program (BJJWZYJH012019100020098), and Open Project Program Foundation of Key Laboratory of Opto-Electronics Information Processing, CAS (OEIP-O-202006).

References

[Ahsan *et al.*, 2019] Unaiza Ahsan, Rishi Madhok, and Irfan Essa. Video jigsaw: Unsupervised learning of spatiotemporal context for video action recognition. In *WACV*, pages 179–189, 2019.

[Alwassel *et al.*, 2020] Humam Alwassel, Dhruv Mahajan, Lorenzo Torresani, Bernard Ghanem, and Du Tran. Self-supervised learning by cross-modal audio-video clustering. In *NeurIPS*, 2020.

[Benaim *et al.*, 2020] Sagie Benaim, Ariel Ephrat, Oran Lang, Inbar Mosseri, William T. Freeman, Michael Rubinstein, Michal Irani, and Tali Dekel. Speednet: Learning the speediness in videos. In *CVPR*, pages 9922–9931, 2020.

[Bengio *et al.*, 2009] Yoshua Bengio, Jérôme Louradour, Ronan Collobert, and Jason Weston. Curriculum learning. In *ICMLE*, pages 41–48, 2009.

[Chen *et al.*, 2019] Yue Chen, Yalong Bai, Wei Zhang, and Tao Mei. Destruction and construction learning for fine-grained image recognition. In *CVPR*, pages 5157–5166, 2019.

[Chen *et al.*, 2020] Ting Chen, Simon Kornblith, Mohammad Norouzi, and Geoffrey Hinton. A simple framework for contrastive learning of visual representations. In *ICML*, pages 10709–10719, 2020.

[Du *et al.*, 2020] Ruoyi Du, Dongliang Chang, Ayan Kumar Bhunia, Jiyang Xie, Yi-Zhe Song, Zhanyu Ma, and Jun Guo. Fine-grained visual classification via progressive multi-granularity training of jigsaw patches. *arXiv preprint arXiv:2003.03836*, 2020.

[Gidaris *et al.*, 2018] Spyros Gidaris, Praveer Singh, and Nikos Komodakis. Unsupervised representation learning by predicting image rotations. *ICLR*, 2018.

[Han *et al.*, 2019] Tengda Han, Weidi Xie, and Andrew Zisserman. Video representation learning by dense predictive coding. In *ICCV Workshop*, pages 1483–1492, 2019.

[Han *et al.*, 2020a] Tengda Han, Weidi Xie, and Andrew Zisserman. Memory-augmented dense predictive coding for video representation learning. In *ECCV*, pages 312–329, 2020.

[Han *et al.*, 2020b] Tengda Han, Weidi Xie, and Andrew Zisserman. Self-supervised co-training for video representation learning. In *NeurIPS*, 2020.

[Hara *et al.*, 2018] Kensho Hara, Hirokatsu Kataoka, and Yutaka Satoh. Can spatiotemporal 3d cnns retrace the history of 2d cnns and imagenet? In *CVPR*, pages 6546–6555, 2018.

[He *et al.*, 2020] Kaiming He, Haoqi Fan, Yuxin Wu, Saining Xie, and Ross Girshick. Momentum contrast for unsupervised visual representation learning. In *CVPR*, pages 9729–9738, 2020.

[Jing *et al.*, 2018] Longlong Jing, Xiaodong Yang, Jingen Liu, and Yingli Tian. Self-supervised spatiotemporal feature learning via video rotation prediction. *arXiv preprint arXiv:1811.11387*, 2018.

[Kay *et al.*, 2017] Will Kay, João Carreira, Karen Simonyan, Brian Zhang, Chloe Hillier, Sudheendra Vijayanarasimhan, Fabio Viola, Tim Green, Trevor Back, Paul Natsev, Mustafa Suleyman, and Andrew Zisserman. The kinetics human action video dataset. *arXiv preprint arXiv:1705.06950*, 2017.

[Kendall *et al.*, 2018] Alex Kendall, Yarin Gal, and Roberto Cipolla. Multi-task learning using uncertainty to weigh losses for scene geometry and semantics. In *CVPR*, pages 7482–7491, 2018.

[Kim *et al.*, 2019] Dahun Kim, Donghyeon Cho, and In So Kweon. Self-supervised video representation learning with space-time cubic puzzles. In *AAAI*, pages 8545–8552, 2019.

[Kuehne *et al.*, 2011] Hildegard Kuehne, Hueihan Jhuang, Estibaliz Garrote, Tomaso Poggio, and Thomas Serre. Hmdb: A large video database for human motion recognition. In *ICCV*, pages 2556–2563, 2011.

[Lee *et al.*, 2017] Hsin-Ying Lee, Jia-Bin Huang, Maneesh Singh, and Ming-Hsuan Yang. Unsupervised representation learning by sorting sequences. In *ICCV*, pages 667–676, 2017.

[Lin *et al.*, 2017] Tsung-Yi Lin, Piotr Dollár, Ross Girshick, Kaiming He, Bharath Hariharan, and Serge Belongie. Feature pyramid networks for object detection. In *CVPR*, pages 2117–2125, 2017.

[Misra *et al.*, 2016] Ishan Misra, C. Lawrence Zitnick, and Martial Hebert. Shuffle and learn: Unsupervised learning using temporal order verification. In *ECCV*, pages 527–544, 2016.

[Noroozi and Favaro, 2016] Mehdi Noroozi and Paolo Favaro. Unsupervised learning of visual representations by solving jigsaw puzzles. In *ECCV*, pages 69–84, 2016.

[Oord *et al.*, 2018] Aaron van den Oord, Yazhe Li, and Oriol Vinyals. Representation learning with contrastive predictive coding. *arXiv preprint arXiv:1807.03748*, 2018.

[Paumard *et al.*, 2020] Marie-Morgane Paumard, David Picard, and Hedi Tabia. Deepzzle: Solving visual jigsaw puzzles with deep learning and shortest path optimization. *TIP*, 29:3569–3581, 2020.

[Soomro *et al.*, 2012] Khurram Soomro, Amir Roshan Zamir, and Mubarak Shah. Ucf101: A dataset of 101 human actions classes from videos in the wild. *arXiv preprint arXiv:1212.0402*, 2012.

[Stringer *et al.*, 2006] Simon M. Stringer, Gavin Perry, Edmund T. Rolls, and J. H. Proske. Learning invariant object recognition in the visual system with continuous transformations. *Biological Cybernetics*, 94(2):128–142, 2006.

[Sun *et al.*, 2019] Chen Sun, Fabien Baradel, Kevin Murphy, and Cordelia Schmid. Learning video representations using contrastive bidirectional transformer. *arXiv preprint arXiv:1906.05743*, 2019.

[Wang *et al.*, 2018] Xiaolong Wang, Ross Girshick, Abhinav Gupta, and Kaiming He. Non-local neural networks. In *CVPR*, pages 7794–7803, 2018.

[Wang *et al.*, 2020] Jiangliu Wang, Jianbo Jiao, and Yun-Hui Liu. Self-supervised video representation learning by pace prediction. In *ECCV*, pages 504–521, 2020.

[Xu *et al.*, 2019] Dejing Xu, Jun Xiao, Zhou Zhao, Jian Shao, Di Xie, and Yueting Zhuang. Self-supervised spatiotemporal learning via video clip order prediction. In *CVPR*, pages 10334–10343, 2019.

[Yao *et al.*, 2020] Yuan Yao, Chang Liu, Dezhao Luo, Yu Zhou, and Qixiang Ye. Video playback rate perception for self-supervised spatio-temporal representation learning. In *CVPR*, pages 6548–6557, 2020.

[Zhuang *et al.*, 2020] Chengxu Zhuang, Tianwei She, Alex Andonian, Max Sobol Mark, and Daniel Yamins. Unsupervised learning from video with deep neural embeddings. In *CVPR*, pages 9563–9572, 2020.