Video-Efficient Foundation Models

Fida Mohammad Thoker

Video-Efficient Foundation Models

Video-Efficient Foundation Models

Fida Mohammad Thoker

This book was typeset by the author using $\LaTeX\,2_\varepsilon$.
The cover was generated by Adobe-FireFly (Model Firefly Image 2 beta) with the following prompts:

- Front: “A 3D render of a glowing sphere inside a data center on a black background and with a prominent artificial deep neural network embedded in the sphere. Concepts: *Futuristic*, Color and Tone: *Monochromatic*, Themes: *Digital art*”
- Back: “Bottom view of GPU clusters in a single-color. Style: *Image generated from the above front prompt*, Concepts: *Futuristic*, Color and Tone: *Monochromatic*, Themes: *Digital art*”

Video-Efficient Foundation Models

ACADEMISCH PROEFSCHRIFT

ter verkrijging van de graad van doctor
aan de Universiteit van Amsterdam
op gezag van de Rector Magnificus
prof. dr. ir. P.P.C.C. Verbeek
ten overstaan van een door het College voor Promoties ingestelde commissie,
in het openbaar te verdedigen in de Agnietenkapel
op vrijdag 8 december 2023, te 10:00 uur

door Fida Mohammad Thoker
geboren te Tral, Jammu Kashmir

Promotiecommissie

Promotor:	prof. dr. C. G. M. Snoek	Universiteit van Amsterdam
Copromotor:	dr. H.R. Doughty	Universiteit van Amsterdam
Overige leden:	dr. Y. M. Asano	Universiteit van Amsterdam
	dr. E. Gavves	Universiteit van Amsterdam
	dr. P.S.M. Mettes	Universiteit van Amsterdam
	prof. dr. ir. H.E. Bal	Vrije Universiteit Amsterdam
	prof. dr. J. Gall	University of Bonn

Faculteit der Natuurwetenschappen, Wiskunde en Informatica

The work described in this thesis has been carried out within the graduate school ASCI, at the Video & Image Sense Lab.

To my cherished son, Ibaad, may this thesis stand as a testament to the power of perseverance and the pursuit of knowledge, inspiring you to reach for the stars in your own journey.

Contents

Chapter 1

Introduction

The world of Artificial Intelligence (AI) has witnessed breathtaking progress, revolutionizing various domains and transforming the way we interact with technology. The current state-of-the-art in AI mainly encompasses developing deep learning algorithms to solve complex tasks. The success of deep learning has empowered machines to accomplish remarkable feats, from visual understanding to auditory learning to natural language processing. Picture this: Deep neural networks can now analyze images and videos, recognizing objects [271], faces [151], and even complex scenes [154]. This has found practical applications in fields like autonomous driving [19], surveillance systems [13, 18], and healthcare [155, 37]. Advances in audio understanding have enabled machines to transcribe spoken words into written text [186], classify sounds [176], monitor noise [194], and recognize speakers [12]. It has empowered voice assistants like Amazon's Alexa to understand and respond to user commands, providing hands-free control and access to information. Similarly, natural language processing has advanced to a point where machines can understand and generate human-like text [43, 20]. This has resulted in advancements such as language translation [168], sentiment analysis [40], and chatbots and virtual assistants. Notably, chatbots like ChatGPT have demonstrated impressive capabilities, simulating human-like interactions and providing valuable assistance across various domains. Moreover, AI has pushed the boundaries of creative expression with audio and visual generation models. AI models can now produce stunningly realistic images and videos [187, 88], compose original music [104, 47], and mimic specific styles of artists [35, 250], blurring the line between natural and computer-generated content. These advances in AI have far-reaching implications, promising a future where technology seamlessly integrates with everything in our lives, providing intelligent solutions and captivating experiences. Conversely, such advancements in AI also have the potential to increase socioeconomic inequality with more automation, increase privacy and security concerns, raise ethical challenges such as autonomous weapons and deep fakes, and make it harder to distinguish between information and misinformation, etc. With such rapid progress and potential for both good and bad, the imperative for more responsible AI becomes evident.

A crucial role in advancing AI in various areas is played by supervised learning. It involves teaching models to learn patterns and relationships between data and their assigned labels. For instance, images are annotated with object classes and

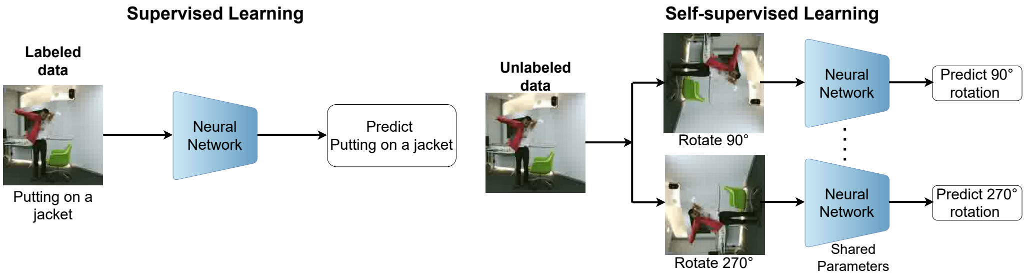


FIGURE 1.1: **Supervised vs Self-supervised Learning**. Supervised learning requires data labeled by humans and aims to learn feature representations by solving the corresponding classification task (here: activity classification). Self-supervised learning learns from unlabeled data and aims to learn feature representations by solving an auxiliary task *e.g.* an N-way classification problem in which the network must predict which rotation angle was used to transform the input video. Self-supervision results in learning high-level semantics directly from the unlabeled data and has become the de facto strategy for training foundation models.

object locations, language data is annotated with sentiment labels, and videos are annotated with activity class labels. By explicitly training models to predict the correct labels for a number of examples, they learn to understand underlying semantic patterns and relationships that exist in the data and associate them with the correct labels. This allows the models to make predictions on new, unseen data. However, meticulously labeling large amounts of data is expensive and time-consuming. Thus, various learning strategies *e.g.* transfer learning, semi-supervised learning, active learning, and self-supervised learning have emerged to address this shortcoming. Rather than relying on manual annotations, self-supervised learning methods design *auxiliary tasks*, that learn meaningful semantic patterns from the inherent structure of the data itself (see Figure 1.1). For instance, models can learn useful representations by learning to recognize the different rotations of an image [72] or by predicting the next word in a sentence [16, 183]. These learned representations can then be transferred to solve different tasks *e.g.* image classification or sentiment analysis with a small amount of task-specific labeled data. The success of self-supervised learning has resulted in the emergence of powerful foundation models [17], *i.e.*, models that are pretrained on broad data and can be adapted to a wide range of downstream tasks. Self-supervised tasks like masked language modeling [43] have become the de facto learning strategy to train large foundation models for text data like GPT [184, 20]. Similarly, in the image domain, self-supervised tasks like contrastive learning [29, 85], masked autoencoding [84, 9] and image-text alignment [185, 105] have achieved the levels of supervised learning or even surpassed it in some cases. However, the same progress has not been made in video self-supervised learning.

Videos are special and distinct due to their unique characteristics and rich information. Firstly, videos capture temporal dynamics, providing a sequence of frames that convey motion, actions, and changes in the scene. This temporal nature allows for the modeling of motion patterns and dependencies, necessary for solving tasks like action recognition [238, 203, 103]. Secondly, videos offer rich contextual information by observing multiple frames, allowing for a broader understanding of objects, events, and their interactions, necessary for modeling tasks such as object detection and tracking [108, 109, 59], temporal action detection [200, 31, 256, 274], spatio-temporal action detection [100, 284], multi-label action classification [202] and temporal repetition counting [276, 190, 127]. Most video self-supervised tasks are adapted from image-based counterparts and often fall short of capturing these intricate aspects that are inherent to video data. Additionally, videos contain large amounts of redundant frames where semantics vary slowly in the temporal dimension, making it computationally expensive to process and train self-supervised models on the same scale as that of images and text. Given these unique characteristics, training video foundation models require self-supervised tasks that can model better motion dynamics and contextual information in videos, while also reducing the associated computational costs. Furthermore, a good video foundation model requires having enhanced generalization capability such that it can be adapted to various downstream contexts *e.g.*, with limited task-specific labels, domains not seen in pretraining, different video-based tasks, etc. Finally, video data can be observed in non-RGB modalities like Audio, Depth Maps, Infrared, and Skeleton Sequences, allowing us to model human actions in a unique way while preserving individuals' privacy, as shown in Figure 1.2. These modalities can offer advantages in challenging scenarios, such as low-light conditions or occlusions, and where privacy concerns are paramount, and have been used for action analysis [263, 235, 260, 199]. Consequently, it is also paramount to explore video data beyond the common RGB modality and build video foundation models that are specific to non-RGB video modalities. Overall, this leads to a core query: Can machines start to understand the dynamic aspects of multi-modal video data, learn spatio-temporal patterns with limited label supervision and limited data, and apply them to unseen contexts in a way that closely resembles how we humans see things?

1.1 Problem Statement

This thesis tackles the problem of ***video-efficiency*** for ***video foundation models***. We define *video-efficiency* as a multifaceted problem that demands video foundation models that are not only accurate but also exhibit *label-efficiency*, *domain-efficiency*, and *data-efficiency*, which we detail one by one.

Label-efficiency refers to using a reduced set of labeled videos for solving video-based tasks. The process of annotating videos involves manually identifying events, delineating temporal boundaries, recognizing objects and their spatial locations, or describing scenes over time. Based on the task, video annotations can span various levels, such as assigning labels to the entire video [113], to specific events within the video [98], or to individual pixels [179]. Thus, it is a resource-intensive endeavor that necessitates substantial capital, human expertise, and time investment. The process

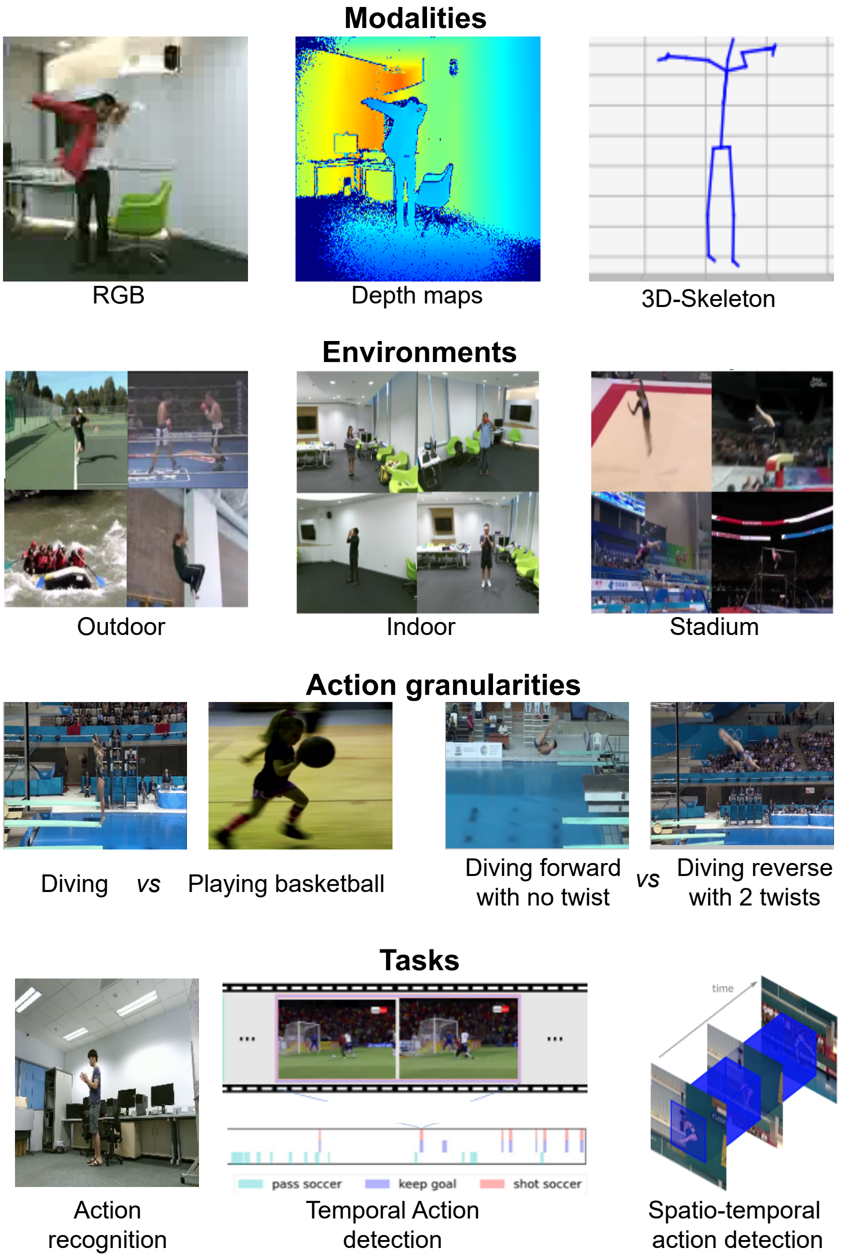

Modalities
RGB
Depth maps
3D-Skeleton
Environments
Outdoor
Indoor
Stadium
Action granularities
Diving vs Playing basketball
Diving forward with no twist vs Diving reverse with 2 twists
Tasks
time
pass soccer
keep goal
shot soccer
Action recognition
Temporal Action detection
Spatio-temporal action detection

FIGURE 1.2: **Multiple facets of video understanding** encompassing different modalities, environments, action granularities, and tasks. Non-RGB modalities represent a privacy-preserving video modality with many potential use cases and therefore require solutions that are specific to these modalities. And, to endow video-efficiency we need models that can be useful for multiple visual environments, granularities, and tasks in video understanding. This thesis addresses all these facets for video-efficient foundation models.

becomes even more challenging for tasks like temporal repetition counting [276], which involves counting the number of repetitions of an event in a period of time. Consequently, there is a critical need to train video foundation models that only require small amounts of labeled video data for the downstream tasks, leading to significant annotation cost reduction.

Domain-efficiency refers to the ability of a video model to be effective for diverse video contexts. For instance, videos originating from diverse environments like indoor setups, outdoor landscapes, or sports arenas, introducing variability in visual contexts (see Figure 1.2). It is important to capture such variability so that the same model can be used for different environments, instead of training a new model for each environment of interest. Similarly, actions in videos can vary in granularity, demanding a distinct level of semantic understanding for coarse-grained actions (diving vs playing basketball) and for fine-grained actions (different types of dives in diving), as shown in Figure 1.2. Furthermore, video-based tasks can encompass a wide range (as shown in Figure 1.2), *e.g.*, action recognition [209], temporal action detection [98], and temporal repetition counting [276], necessitating tailored insights and representations. The conventional approach requires training new models specific to each such domain variability which is time-consuming and resource-intensive. Consequently, the need for video foundation models that adapt to diverse visual contexts, accommodate varying levels of granularity, and apply to multiple tasks becomes imperative, allowing for the same model to be useful for various applications.

Data-efficiency pertains to the capability of a model to learn effectively from a limited amount of video data without experiencing a substantial drop in performance. Traditional deep learning models [82, 110, 112, 256] for video understanding typically demand vast video datasets and extensive computational resources. However, real-world constraints, such as privacy concerns or limited training resources, may impede the ability to train on large volumes of video data. In such scenarios, the ability to extract meaningful insights and achieve desirable results with limited data becomes paramount. Data efficiency allows for more efficient utilization of available resources and facilitates the development of video foundation models in contexts where data collection is challenging or computational resources are limited.

1.2 Research Questions

To address the problem of video-efficiency in video foundation models, we pose the central research question of this thesis:

What enables video-efficient video foundation models?

Our research covers label-efficiency, domain-efficiency, and data-efficiency of video foundation models in standalone and in combination. We examine video-efficient learning for different types of input video modalities: RGB, Depth maps, and 3D-Skeleton sequences, and for different learning strategies: knowledge transfer, and self-supervised learning. We observe that video foundation models are not extensively explored for different dimensions of video-efficiency. In this thesis, we dive into our main research question by answering a specific sub-question in each of the four chapters.

As a starting point for our investigation into video-efficiency, we aim to achieve label-efficiency for non-RGB video modalities. The goal is to recognize and detect actions from non-RGB modalities like Depth maps and 3D-skeleton sequences when only limited modality-specific labeled examples are available. For the RGB modality, many large-scale labeled datasets [113, 25, 111] have been made available. They have become the de facto pretraining choice when recognizing or detecting new actions from RGB datasets that have limited amounts of labeled examples available. Unfortunately, such large-scale labeled action datasets for non-RGB modalities are unavailable for supervised pretraining, highlighting the need for alternate solutions for label-efficient action recognition or detection with non-RGB video data. Therefore, we investigate the feasibility of transferring knowledge from pretrained RGB-based video models to efficiently train new non-RGB video models. Consequently, the following research question is posed:

What transfers efficiently across video modalities?

In Chapter 2, we answer this question and propose a novel method for cross-modal knowledge distillation between RGB and non-RGB video modalities. To train action models with limited labeled examples from non-RGB video modalities we propose a teacher-student approach that leverages knowledge from large-scale labeled RGB data. Our proposal involves a two-step training process: (*i*) extracting action representation knowledge from an RGB-trained teacher network and adapting it to a non-RGB student network, and (*ii*) finetuning the transfer model using available labeled examples of the non-RGB target modality. For knowledge transfer, we introduce feature-supervision strategies that rely on unlabeled paired data from the RGB and the target modality to effectively transfer feature-level representations from the teacher to the student network. Experimental evaluation demonstrates the effectiveness of our approach in improving the label-efficiency for action recognition and detection using Depth maps and 3D-skeleton sequences. Moreover, our method showcases domain-efficiency by effectively transferring knowledge from RGB datasets captured in different environments.

Although our proposed cross-modal transfer improves label- and domain-efficiency for the non-RGB modalities. It still requires a large amount of labeled RGB data for training the teacher network as well as access to unlabeled paired videos from the RGB and the target modalities for knowledge transfer. Recent approaches, increase label efficiency by removing the need for any labels in learning feature representations by relying on self-supervision [85, 29]. The objective of self-supervised learning is to directly learn feature representations from the unlabeled data of a specific modality, which can then be finetuned with a small amount of labeled data from the same modality. However, these self-supervision approaches [169, 181, 173, 94] predominantly focus on the RGB video modality and are less prevalent for non-RGB modalities such as 3D-skeleton sequences. This leads us to the next research question: Can we enhance video efficiency for non-RGB modalities through self-supervision? Specifically, we pose the research question:

What is self-learnable for 3D-skeleton video sequences?

Self-supervised methods applied to the RGB domain [85, 29, 181, 169, 173, 94, 41] have achieved success through contrastive learning, which enables the learning of invariances to various data augmentations. The effectiveness of these methods is attributed to the augmentation techniques that are possible in the RGB domain, which can generate meaningful positive pairs capturing spatio-temporal dynamics in video data. Conversely, self-supervised approaches [286, 165, 211, 136] for 3D-skeleton data rely on pretext tasks such as reconstructing masked input skeletons or predicting joint motions to learn feature representations. Adapting contrastive learning for 3D-skeleton data poses challenges due to the absence of comparable augmentations and the sparse nature of skeleton data, which restricts the input sampling space. In Chapter 3, we propose a novel self-supervised method for 3D-skeleton sequences based on contrastive learning. Our proposal is built upon learning invariances to various skeleton augmentations and input skeleton representations via a noise contrastive estimation. In particular, we contribute several skeleton-specific spatial and temporal augmentations that can generate meaningful positive pairs to learn the spatio-temporal dynamics of the skeleton data. In addition, we propose inter-skeleton contrastive learning, which learns from multiple different input skeleton representations in a cross-contrastive manner. This also enriches the input sampling space for contrastive learning. By learning similarities between different skeleton representations as well as augmented views of the same sequence, the network is encouraged to learn higher-level semantics of the skeleton data than when only using the augmented views. Besides achieving state-of-the-art results on challenging benchmarks for skeleton-based action recognition and retrieval, our approach showcases superior label-efficiency compared to previous methods when evaluated on downstream setups with limited labeled data.

In the previous chapters, our primary focus has centered on improving the video efficiency of non-RGB video foundation models. However, it is important to acknowledge that most video-based tasks [238, 203, 103, 225, 270, 22, 256, 259, 175, 21, 276, 190, 127] are commonly addressed in the RGB domain due to its prevalence and visual nature. This emphasizes the significance of enabling video efficiency of the RGB-based video foundation models. Previous research has addressed various aspects of video efficiency in the RGB domain by building video foundation models that aim to learn general video representations using large-scale video datasets. These representations are subsequently employed to enhance the label and domain efficiency of new video contexts through transfer learning. The simplest way to train a video foundation model is via supervised pretraining *i.e.* by training action classification models on large human-annotated video datasets [82, 110]. Recently, self-supervised learning techniques have shown significant advancements in training video foundation models [64, 216, 181, 173, 139, 8, 4, 158] by directly learning from unlabeled video data. However, the emphasis has primarily been on creating novel self-supervised tasks, leaving a substantial gap in understanding the video efficiency of self-supervised models. For instance, the current evaluation approach of video self-supervised tasks involves training models on a large unlabeled dataset [113] and finetuning them on small labeled action classification datasets [209, 122] to assess the performance. The finetuning datasets consist of videos that exhibit a high degree of domain similarity to those employed in self-supervised training. Specifically, they share comparable environmental conditions and action granularity and only evaluate for action classification tasks. This raises concerns about whether high-performing models will succeed in challenging real-world scenarios. These scenarios may involve a scarcity of labeled videos or videos with different visual environments and action granularities compared to those in the self-supervised training datasets. They may also encompass video tasks that go beyond action classification. Consequently, it is essential to thoroughly investigate the effectiveness of video representations obtained through current self-supervised techniques in terms of both label- and domain-efficiency. This prompts us to formulate the following research question:

What limits video-efficient foundation models?

In Chapter 4, we systematically address the research question by conducting a comprehensive large-scale study to evaluate the performance of existing self-supervised video learning methods across various downstream setups. These setups encompass different environmental conditions, varying amounts of labeled data, diverse levels of granularities, and a range of video tasks. Our study entails over 500 experiments conducted on 7 video datasets, employing 9 self-supervised methods, and evaluating performance on 6 video understanding tasks. The results of our study reveal that current benchmarks in video self-supervised learning fail to adequately capture the generalizability of representations across these diverse downstream factors. We observe that self-supervised methods significantly underperform compared to vanilla supervised pretraining, particularly in scenarios involving substantial domain shifts and limited labeled samples. Through extensive analysis, we distill a subset of our experiments known as the SEVERE-benchmark, which provides insights into video-efficiency of representations obtained by existing and future video self-supervised methods. Our study highlights the limitations of existing video self-supervised foundation models in terms of their ability to generalize effectively to diverse and challenging downstream setups. This finding motivates us to pose the final research question:

What generalizes video-efficient foundation models?

Most video self-supervised methods are based on contrastive learning [181, 173, 169, 158, 171] where the goal is to increase feature similarity between spatially and temporally augmented clips from the same video, known as positive pairs. Despite temporal differences, such positive pairs maintain a high spatial similarity and coarse-grained features can solve the contrastive task without needing to explicitly capture local motion dynamics. This limits the generalizability of the learned video representations, especially to domains that require finer motion understanding, as observed in Chapter 4. To address this limitation, in Chapter 5 we propose a new video self-supervised learning task that explicitly aims to learn motion-focused video representations. In particular, we propose a contrastive method to learn similarities between videos with identical local motion dynamics but an otherwise different appearance. We do so by adding synthetic motion trajectories to videos which we refer to as tubelets. By simulating different tubelet motions and applying transformations, such as scaling and rotation, we introduce motion patterns beyond what is present in the pretraining data. This allows our model to be data-efficient in the pretraining too. In particular, our method demonstrates remarkable data-efficiency in pretraining and can maintain its performance when using only 25% of the pretraining data. Experimental evaluations conducted across diverse downstream settings demonstrate that our approach enables self-supervised learning of video foundation models that are not only data-efficient but also label-efficient and domain-efficient.

To summarize, this thesis focuses on video-efficiency of video foundation models. It begins by exploring the advantages of video-efficient learning. The thesis delves into technological innovations that facilitate various aspects of video-efficiency for non-RGB- and RGB-based video foundation models. In the following section, we present a list of all publications that resulted from the research. Furthermore, each research question introduced above is expanded upon in the subsequent chapters. Finally, we provide our conclusion in the last chapter of the thesis.

1.3 Publications, Co-authorship, and Roles

For each chapter of this thesis, we here declare the authors' contributions:

Chapter 2

Fida Mohammad Thoker, Cees G.M. Snoek (2020). *"Feature-Supervised Action Modality Transfer"*. In: IEEE International Conference on Pattern Recognition [223].

• F.M. Thoker	All aspects
• C.G.M. Snoek	Insight and supervision

Chapter 3

Fida Mohammad Thoker, Hazel Doughty, Cees G. M. Snoek (2021). *"Skeleton-Contrastive 3D Action Representation Learning"*. In: Proceedings of the ACM International Conference on Multimedia [221].

• F.M. Thoker	All aspects
• H. Doughty	Guidance and technical advice
• C.G.M. Snoek	Insight and supervision

Chapter 4

Fida Mohammad Thoker, Hazel Doughty, Piyush Bagad, Cees G. M. Snoek (2022). *"How Severe is Benchmark-Sensitivity in Video Self-Supervised Learning?"*. In: European Conference on Computer Vision [224].

• F.M. Thoker	All aspects
• H. Doughty	Guidance and technical advice
• P. Bagad	Experiments and analysis
• C.G.M. Snoek	Insight and supervision

Chapter 5

Fida Mohammad Thoker, Hazel Doughty, Cees G. M. Snoek (2023). *"Tubelet-Contrastive Self-Supervision for Video-Efficient Generalization"*. In: International Conference on Computer Vision [220].

• F.M. Thoker	All aspects
• H. Doughty	Guidance and technical advice
• C.G.M. Snoek	Insight and supervision

Chapter 2

Feature-Supervised Action Modality Transfer

2.1 Introduction

The goal of this chapter is to recognize an action like *drinking water*, *hugging* or *falling down* in multimodal video content, be it a stream of RGB pixels [62, 113, 83], depth maps [263, 144, 246] or 3D-skeletons [199, 126, 249]. The common approach to action recognition in video is to train a deep convolutional neural network on massively labeled RGB, or derived optical-flow, video datasets like Kinetics [113], Sports-1M [111] or ActivityNet [56]. These pre-trained RGB models are also valuable to recognize or detect new actions from alternative RGB videos, with only limited amounts of labeled action examples available for fine-tuning [209, 122, 98], thereby saving a lot of annotation cost. Unfortunately, for non-RGB video modalities massively labeled action datasets, and the corresponding pre-trained models, are scarce. In this chapter, we strive for limited-example action recognition in non-RGB video modalities, like depth maps and 3D-skeletons, by learning from large-scale RGB video data labeled with other actions.

We take inspiration from the ideas of general knowledge distillation by Hinton *et al.* [87] and cross-modal distillation for action recognition. *e.g.* [66, 65, 38, 222]. The goal of knowledge distillation is to compress a large complex teacher network into a small and simple student network. In cross-modal distillation a teacher network is first trained to recognize a set of actions from the source action modality using many labeled examples. Then the teacher network distills knowledge to the student network to recognize the same set of actions from a different target action modality. We adapt these ideas for a different setting. That is, to train a student network to recognize a set of actions from a target modality while distilling knowledge from a teacher network that has been trained to recognize a different set of actions from a different source action modality. This scenario has a practical application for recognizing or detecting new actions from non-RGB action modalities with limited labeled examples. Instead of relying on labeled large scale datasets of these non-RGB modalities for pre-training (which are scarce), we can distill knowledge from existing RGB trained action models. In summary, we aim to transfer information about recognizing actions across modalities and across classes.

In this chapter, we propose to recognize and detect actions from non-RGB modalities like depth maps and 3D-skeleton sequences, when only limited labeled examples for these modalities are available. To achieve this, we assume an RGB trained action model is given and we also have access to many unlabeled pairs of two modalities (paired RGB and non-RGB actions), along with some labeled examples for the non-RGB action modality. Then, the trained RGB model acts as the teacher network to supervise the learning of the non-RGB student model using unlabeled modality pairs. In contrast to the general knowledge transfer, which distills class probabilities from the teacher to the student network, we distill action representations from the teacher to the student network via *feature-level supervision*. More precisely, for a given unlabeled modality pair, the non-RGB student network is optimized to match its output features with that of the trained RGB teacher. After this cross-modal distillation step, the non-RGB student network is fine-tuned with the available labeled examples of the non-RGB modality for a downstream task. Before presenting our method, we will first discuss in more detail related work.

2.2 Related Work

2.2.1 Modalities for Action Recognition

Modern action recognition, *e.g.*, [113, 83, 230, 228, 245] relies on deep (2D or 3D) CNN architectures that learn to classify human actions from video data. These methods usually require a common video modality such as RGB, the RGB-derived optical-flow or both [204, 58, 283] to achieve best performance for this task. For these video modalities many large-scale and publicly available annotated action datasets exists, such as Kinetics-(400, 600 and 700) [113, 24, 25], Sports-1M [111], and ActivityNet [56]. These sets also act as valuable pre-training resource for classifying and detecting new actions from other RGB action datasets, which have smaller amounts of labeled examples.

There is also a large body of works that learn to classify human actions from other video modalities such as depth maps [263, 235, 144, 246], sequences of 3D-skeletons [199, 126, 249, 260], and even radio frequencies [130]. Although action recognition networks for these modalities may perform well, they require a large number of labeled action examples from the target modality for training. In contrast, our method utilizes large-scale labeled datasets of the commonly available RGB modality to boost the performance on non-RGB modalities, especially when only limited amounts of non-RGB labeled examples are available.

2.2.2 Knowledge Transfer

Recently, knowledge distillation has been explored to transfer knowledge across modalities for tasks like emotion recognition [3], pose estimation [285], object detection [89, 78, 27], video captioning [273, 247] and action recognition [66, 65, 38, 210, 67, 222, 192].

Gupta *et al.* [78] transfer knowledge from the RGB to the depth modality for the task of object detection in images using a cross-modal teacher-student network that matches features between the two modalities. Similarly, Sayed *et al.* [192] proposed a self-supervised method to learn feature similarity between RGB and optical flow modalities by maximizing similarity between clip features from paired RGB-flow videos and minimizing similarity across unpaired video clips. We rely on a similar principle, but different from both [78] and [192] we propose to exploit the temporal structure of video data for better information exchange via new feature-supervised granularities, like clip-to-clip, video-to-clip and video-to-video. These granularities not only improve transfer performance for action classification and detection, but are also necessary for challenging modality pairs like RGB and 3D-skeletons.

Thoker and Gall [222] proposed cross-modal transfer where a source (teacher) network is already trained to recognize a set of actions from the RGB modality. Their goal is to train a new (student) network to recognize the same set of actions, but from the skeleton modality. Unlabeled RGB-Skeleton action pairs are used such that the output action class predictions from the RGB teacher are matched by corresponding 3D-skeleton based student via common distillation losses like cross entropy (CE) or Kullback-Leibler divergence (KL). We deal with a more difficult variant of this problem, where the goal is to train a non-RGB student network to recognize a *different* set of actions than those of the RGB teacher. Thus, the action classes seen by the teacher and the student network are disjunct for our case and the CE and KL losses used by Thoker and Gall are no longer applicable. Instead, we propose to rely only on feature-level supervision between teacher and student by minimizing the cosine distance. Instead of transferring class labels, we transfer action-specific feature-representations from the RGB modality to a non-RGB target modality. The features learned in this manner can be then fine-tuned to different downstream tasks.

Closest to our work are [66, 65, 38, 210, 67], who all propose to extract knowledge from one or more source modalities to enhance action classification in a different target modality. Particularly, both Crasto *et al.* [38] and Stroud *et al.* [210] boost the performance of RGB-only action recognition by distilling knowledge from a trained optical-flow teacher. Similarly, Garcia *et al.* in [66] and [67] rely on depth maps with or without optical-flow, as labeled paired source modalities to boost the performance of RGB-only action recognition. These methods also assume the network for the source modality is trained to recognize the same set of action classes as the target modality and they rely on labeled pairs (or triplets) of the respective modalities to transfer class-specific information from the source to the target modality. As mentioned previously, we assume the action classes of the source-based network to be different from the target-based network and we rely on unlabeled pairs to transfer feature-level information from the source to the target modality. As a result, our method can use large-scale labeled RGB action datasets as the pre-training data to recognize or detect new actions from non-RGB datasets with only a limited amount of labeled examples. We further differ from these methods by transferring knowledge to a more difficult target modality like 3D-skeleton data and extending cross-modal transfer to the task of temporal action detection.

2.3 Proposed Method

We consider the tasks of action classification and detection for a non-RGB modality, *e.g.* depth maps or sequences of 3D-skeletons, while requiring a reduced amount of labeled examples. To achieve this, a teacher-student network extracts knowledge from a pre-trained off-the-shelf RGB, or optical-flow, action model using unlabeled modality pairs. We first discuss our general approach for this feature-supervised knowledge transfer and then detail transfer granularities for different modality pairs.

2.3.1 Cross-Modal Teacher-Student

Lets assume that we are given a network that has been trained on a large action-class labeled dataset of trimmed RGB videos. We call this dataset the source dataset and the corresponding network acts as the teacher network. We also assume to have access to another dataset called the target dataset that contains many unlabeled action pairs from two paired modalities –*i.e.*, the RGB and the non-RGB target modality. The target dataset also contains some labeled action examples for the target modality, along with the unlabeled pairs. Note the action classes of the source and target dataset do not overlap. We train the student network to extract knowledge from the teacher network, which has learned from the labeled RGB modality of the source dataset, with the goal to adapt it to the non-RGB target modality.

Formally, given a training pair (V_{RGB}, V_{Target}) from unlabeled target data, the trained teacher network outputs a feature vector F_{RGB} for the RGB video V_{RGB}. The student network uses the target modality V_{Target} as its input and is supervised by the corresponding RGB feature F_{RGB} from the teacher network. The student network is optimized to match its features F_{Target} with that of the teacher network using an appropriate similarity loss. We select for F_{RGB} and F_{Target} the outputs of the layer just before the fully connected layers of the teacher and the student network. By doing so, we teach the student network to learn high-level semantics of the actions learned by the teacher, rather than learning the class-specific information present in the fully connected layers. Note that the student network can have same or different architecture as that of the teacher network, but the dimension of its output features F_{Target} should be matched with F_{RGB}, if different. After the knowledge transfer step, the student network can be fine-tuned for a downstream task using the labeled data from the target dataset. During fine-tuning, a standard cross-entropy loss is used for the action classification and a regression loss for the temporal action detection.

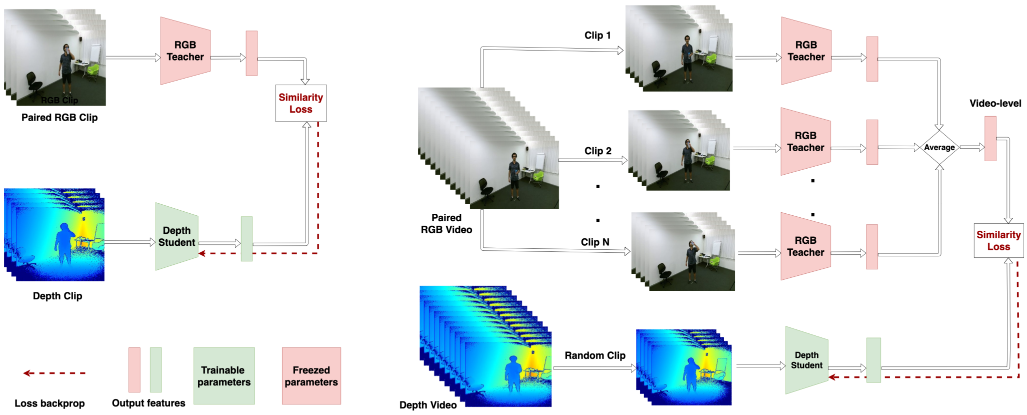


FIGURE 2.1: **Feature-supervised action modality transfer.** A teacher network is trained on a large and labeled RGB, or derived optical-flow, action dataset. *(left) Clip-to-Clip transfer from the RGB teacher to the depth student.* A paired RGB-depth clip is sampled from the whole RGB-depth video pair and the student network is optimized to match its features from the depth clip with that of the teacher features obtained from the corresponding paired RGB clip. (right) *Video-to-Clip transfer from RGB to depth maps.* The whole RGB video is divided into N clips to aggregate clip level features from teacher network into a video-level feature. A clip is then randomly sampled from the paired whole depth video and the student network matches the clip features with the corresponding video level RGB features from the teacher.

2.3.2 Feature-Supervised Granularity

The input granularity to the action recognition models depends on the architecture of the network and the nature of the modality involved. For example, most state of the art works for image based modalities (RGB, optical-flow, depth, *etc*.) use a small video clip as their input for predicting the action classes, while skeleton based models rely on a whole video sequence as the input. Similarly, during inference clip-level models combine prediction of different clips from the whole video to produce more accurate results. Thus, in order to explore the architecture of our cross-modal framework for different pairs of modalities and how to effectively extract knowledge from the teacher network, we consider three different transfer granularities, *i.e*., clip-to-clip, video-to-clip and video-to-video.

2.3.2.1 Clip-to-Clip

We use this granularity for transfer between RGB and depth pairs, since both modalities require a small video clip as input. The transfer strategy for this granularity is shown in Figure 2.1 (left). In particular, a paired small clip is sampled from the whole video of two modalities and a depth-based student network is optimized to match the features of the corresponding RGB clip. Thus, the student network learns to mimic the clip-level features of the RGB teacher by minimizing the following loss function:

$$\mathcal{L}\left(F^{clip}_{Target}, F^{clip}_{RGB}\right) = Cosine\_Distance\left(F^{clip}_{Target}, F^{clip}_{RGB}\right) \tag{2.1}$$

where F^{clip}_{RGB} and F^{clip}_{Target} are the clip level features predicted by the RGB and depth networks respectively.

2.3.2.2 Video-to-Clip

Different from the clip-to-clip transfer, the teacher network extracts for this granularity information from the whole RGB video and then transfers it to the student network. This transfer strategy is shown in Figure 2.1 (right). This strategy again fits for knowledge transfer between RGB and depth pairs. More formally, the RGB video is divided into N clips $C_1, C_2, ..., C_N$ of equal duration. For each clip C_i the teacher network outputs a feature vector $\mathcal{F}_i$ and a global video level feature is obtained by combining these clip level features. Then, the student network randomly samples one of the clips from the corresponding paired depth video and uses the video level RGB feature from the teacher for supervision. Thus, the student network is optimized to match each target clip with the corresponding video-level RGB feature from the teacher network by minimizing the following loss:

$$\mathcal{L}\left(F^{clip}_{Target}, F^{video}_{RGB}\right) = Cosine\_Distance\left(F^{clip}_{Target}, F^{video}_{RGB}\right) \tag{2.2}$$

$$F^{video}_{RGB} = \frac{1}{N}\sum_{1}^{N}\mathcal{F}^{clip}_{i} \tag{2.3}$$

where F^{video}_{RGB} is computed by taking an average over N RGB clips. The number of clips N for each video is calculated as $\frac{Total\ Number\ of\ frames}{Input\ size\ of\ the\ network}$.

Naturally, we can also combine the clip-to-clip and video-to-clip granularity, such that both clip-level and video-level information is transferred to the student network. In this scenario, the student network will be optimized to match features of each target clip with that of the corresponding paired RGB clip and with the global video level feature from the whole paired RGB video. The following loss function is minimized for this optimization:

$$\mathcal{L}\left(F^{clip}_{Target}, F^{clip}_{RGB}, F^{video}_{RGB}\right) = \mathcal{L}\left(F^{clip}_{Target}, F^{clip}_{RGB}\right) + \mathcal{L}\left(F^{clip}_{Target}, F^{video}_{RGB}\right) \tag{2.4}$$

We will assess the knowledge transfer abilities of these strategies in the ablation experiments.

2.3.2.3 Video-to-Video

This granularity is required to transfer knowledge from the RGB to the 3D-skeleton modality, since the input to a skeleton-based network is the whole skeleton sequence. The transfer strategy is the same as for the video-to-clip; the only change being the student network is now replaced by a skeleton based architecture whose input is the whole skeleton video, instead of a small clip. Now, the student network is optimized to match the whole skeleton sequence with the global video-level feature of the paired RGB teacher by minimizing the following loss:

$$\mathcal{L}\left(F_{Target}^{video}, F_{RGB}^{video}\right) = Cosine\_Distance\left(F_{Target}^{video}, F_{RGB}^{video}\right) \tag{2.5}$$

where F_{Target}^{video} is the output feature of the student network which operates over a whole sequence.

2.4 Experimental Setup

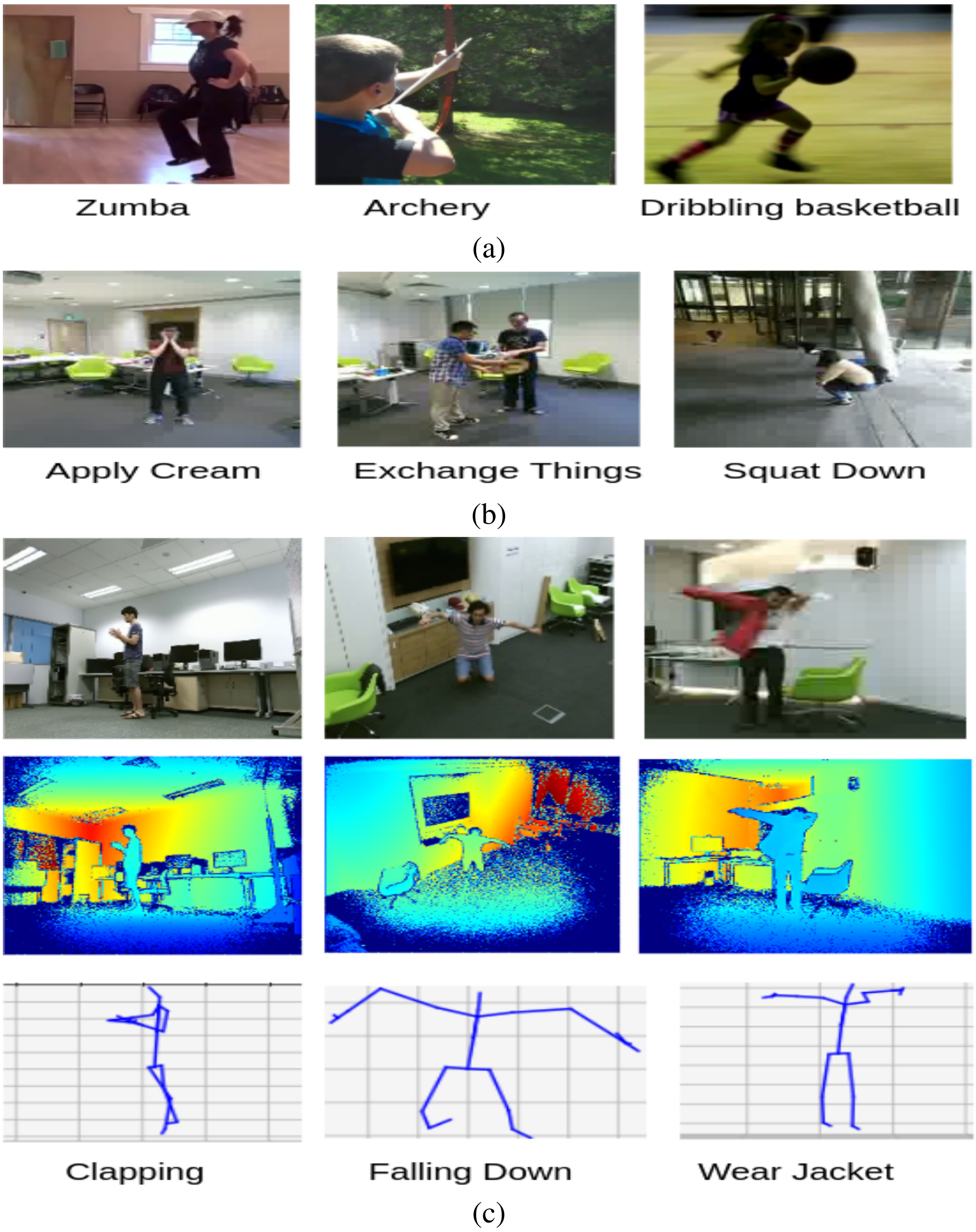
2.4.1 Target Datasets

NTU RGB+D [196] contains 56,880 trimmed videos with paired RGB, depth and 3D-skeleton modalities for 60 action classes. All actions are captured in indoor scenes with multiple cameras, different backgrounds, multiple subjects and camera setups. The dataset is split by a cross-view setup into 37,920 training and 18,960 validation videos [196]. For the knowledge transfer, we use the video modality pairs from the training set without any action class labels. For fine-tuning, we sample the action class examples of the target modality from the training set. The validation set is only used for evaluation purposes and is not seen during any training.

PKU-MMD [34] contains 1,074 long untrimmed videos with paired RGB, depth and 3D-skeleton modalities for 51 action classes and about 20K action instances. All actions are captured in indoor scenes as well, with multiple cameras, different backgrounds, different views and 66 subjects. The annotations for each video contain the start and end locations of multiple activities, along with the action class. The dataset is split by a cross-subject setup into 942 training and 132 validation videos. We sample the labeled videos from the training set for the task of temporal action detection, for fine-tuning only. The validation set is used only for evaluation.

2.4.2 Source Datasets

Kinetics-400 [113] is a large RGB-only dataset containing 400K labeled examples for 400 different action classes. The dataset is collected from YouTube with videos coming from a variety of sources, setups, *etc*. Note the domain difference between this dataset and our target datasets is considerable.



(a)

(b)

(c)

FIGURE 2.2: **Action modality samples from different source and target datasets**. (a) Sample frames from the Kinetics-400 dataset. (b) Sample frames from the NTU-120_minus_60 dataset. (c) Paired multimodal samples from the NTU RGB+D dataset. Note the domain change between the dataset in (a) and the other two datasets.

NTU-120_minus_60 Starting from the NTU RGB+D 120 [141] dataset, we remove all the videos that contain action classes overlapping with NTU RGB+D. Thus, this dataset shares the same domain as our target dataset, but contains different action classes. We only use the RGB modality of this dataset to train the teacher network.

We provide some target and source dataset samples in Figure 2.2.

2.4.3 Implementation Details

Teacher Network: We use 3D-ResNets [83] with RGB or optical-flow as input for our teacher networks. For Kinetics-400, we rely on the pre-trained 3D-ResNext101 models available from [38] and for NTU-120_minus_60 we train a 3D-ResNet18 from scratch. We use 16-frame clips as the input and all other hyperparameters from [83].

Depth Student Network: For the *depth maps*, we use a 3D-ResNet18 architecture as the student network with 16-frame clips as the input. We first apply a multi-scale corner cropping to the 16-frame clip as mentioned in [83], followed by a resizing operation into a 3 x 16 x 112 x 112 sample and a horizontal flipping with 50% probability. The network is trained with an SGD optimizer using a weight decay of 0.001, 0.9 momentum, an initial learning rate of 0.1 and a batch size of 128. During cross-modal transfer we train the whole network for 400 epochs. For fine-tuning, we only train the fully connected layer and the last ResNet block for a total of 100 epochs. For evaluation, the predictions of all clips from the whole video are averaged for final classification.

3D-skeleton Student Network: For the *3D-skeleton data*, we use a Spatio Temporal Graph Convolution Network [260] as the student network with whole 3D-skeleton sequences as the input. Following the setting of [260] no data augmentation is applied. The network is trained with an SGD optimizer using a weight decay of 0.00001, momentum 0.9, an initial learning rate of 0.1 and a batch size of 100. We train the network for 120 epochs during cross-modal transfer and 70 epochs during fine-tuning. During both steps, all layers of the ST-GCN network are trained. For evaluation, a class is predicted for a given 3D-skeleton sequence.

2.5 Results

2.5.1 Ablation

For all ablation experiments we consider the task of classifying actions of NTU RGB+D from depth maps using only a limited amount of labeled depth examples. An action classification network trained with RGB or optical-flow modality from the NTU_120_minus_60 source dataset acts as the teacher network. For the cross-modal transfer step, all RGB-depth or optical-flow-depth pairs from the training set of NTU RGB+D are used without action labels as unlabeled modality pairs for knowledge

	Target-Modality: Depth		
Source-Modality	20 per-class	50 per-class	100 per-class
RGB	62.85±0.5	66.01±0.6	68.64±0.3
Flow	68.43±0.2	71.53±0.1	73.43±0.3
Two-stream	69.16±0.5	72.10±0.5	74.41±0.5

TABLE 2.1: **Which source modality.** RGB and optical-flow teacher networks trained on the NTU_120_minus_60 dataset. The two-stream fuses the individual predictions. The video-to-clip strategy is used for knowledge transfer. The optical-flow teacher provides the best individual features for knowledge transfer to depth maps, fusion improves results further.

	Target-Modality: Depth		
Loss-Function	20 per-class	50 per-class	100 per-class
MSE	65.69±0.3	69.27±0.5	71.65±0.4
Cosine	68.43±0.2	71.53±0.1	73.43±0.3

TABLE 2.2: **Which loss function.** MSE vs. cosine loss for feature-supervised knowledge transfer. For both, the teacher network is trained on the optical-flow modality from NTU_120_minus_60. The video-to-clip strategy is used for knowledge transfer. We obtain better results with a cosine loss.

transfer. For fine-tuning we sample some depth examples with action labels from the training set of NTU RGB+D (20, 50 or 100 examples per action class). For evaluation, we report the video-level accuracy on the validation set from the cross-view split of NTU RGB+D. Each experiment is repeated five times with different random seeds and mean accuracy with variance is reported.

2.5.1.1 Which source modality?

We first consider which source modality to use in the teacher network for knowledge transfer. Table 2.1 shows the results for RGB and optical-flow as teacher modalities. While both modalities provide good features for knowledge transfer, the optical-flow teacher performs better than the RGB teacher. We attribute this to the motion features that contain better cues about the action representations for transfer, as compared to the RGB teacher, which provides features that mainly model spatial action information. Naturally, we can also combine the two source modalities into a two-stream network by fusing the results of the RGB and optical-flow student streams during inference. This further improves the action classification performance, at the expense of double compute and parameters. In summary, the optical-flow teacher provides the best individual features for knowledge transfer and we focus on the optical-flow as the main source modality in the remaining experiments, unless indicated otherwise.

	Target-Modality: Depth		
Granularity	20 per-class	50 per-class	100 per-class
Clip-to-Clip	64.80±1.0	70.30±0.4	72.92±0.5
Video-to-Clip	68.43±0.2	71.53±0.1	73.43±0.3
Combined	69.16±0.2	73.60±0.1	76.24±0.3

TABLE 2.3: **Which granularity.** Combining clip and video level features from the teacher network, as detailed in Equation. 2.4, acts as the best feature-supervision strategy.

2.5.1.2 Which loss function?

Since the goal of the student network is to match the features of the teacher network, any similarity (or distance) function will work for this optimization. We just consider two common loss functions: the mean square loss and the cosine distance loss in Table 2.2. We observe the cosine loss to work better for our task. Thus, for all other experiments we choose the cosine loss to optimize the student networks.

2.5.1.3 Which granularity?

Next we evaluate the granularity of the feature-supervised transfer. From the results in Table 2.3 we observe the video-to-clip transfer works better than the clip-to-clip transfer, especially when only few labeled examples are available for fine-tuning. This is expected, as the student network in the video-to-clip strategy is optimized to match the video-level features that aggregate global information about the action representations from the whole video. Hence, this strategy provides better supervision, as compared to clip-to-clip features that transfer only local information about the action from a small clip. We also observe that combining the two strategies achieves the best knowledge transfer. To summarize, the video-to-clip transfer works better than the clip-to-clip transfer and combining the two gives an additional improvement. For the rest of the experiments we use this granularity for the feature-supervised transfer, unless indicated otherwise.

2.5.1.4 Which layers to transfer?

We also explore which layers from the teacher network are best suited for the transfer. In particular we tried to match the output features of each ResNet block of the teacher network with the corresponding ResNet block of the student network, along with the last layer as before. We found that matching these additional layers does not add anything significant to the transfer and relying on the last-layer only still provides the best results.

	Target-Modality: Depth		
Method & Source Domain	20 per-class	50 per-class	100 per-class
From-scratch	11.00±2.0	33.51±0.4	54.10±0.5
Flow-pretraining (Kinetics)	23.68±2.0	41.95±1.0	54.45±0.5
RGB-pretraining (Kinetics)	24.84±2.0	42.96±1.0	55.05±0.5
Flow-pretraining (NTU)	24.34±1.0	53.72±0.8	64.88±0.4
RGB-pretraining (NTU)	41.55±1.0	57.20±0.8	66.41±0.4
RGB-feature-supervised (Kinetics)	33.31±1.0	47.38±0.5	55.10±0.5
Flow-feature-supervised (Kinetics)	52.17±1.0	59.51±0.6	64.14±0.5
RGB-feature-supervised (NTU)	63.05±1.0	66.71±0.6	68.64±0.3
Flow-feature-supervised (NTU)	68.43±0.2	71.53±0.1	73.43±0.3

TABLE 2.4: **Domain Generalization.** Performance of the depth student under varying source domains. From-scratch indicates training directly on the target modality without any pretraining. Pretraining indicates pretraining on RGB/Flow modality of the source dataset and then directly fine-tuned with labeled depth maps from the target dataset. Feature-supervised indicates the pretraining via our cross-modal transfer with the RGB/Flow trained on a source dataset as the teacher network, followed by fine-tuning with the labeled depth maps from the target dataset. All models are evaluated on the cross-view validation set of NTU RGB+D. Our method outperforms both training from scratch as well as simple pretraining. Also, the teacher network from a more similar domain provides better transfer features.

2.5.2 Domain Generalization

Next, we evaluate the effect of training with different source domains and compare our method with simple pretraining with RGB/Flow modalities. Table 2.4 shows the performance of simple RGB/Flow pretraining and our feature-supervised method on NTU-120_minus_60 and Kinetics-400 source datasets. It is evident from the table that for both datasets our method outperforms simple pretraining with RGB and Flow modalities by a considerable margin especially when using limited amounts of labeled examples during fine-tuning. At the same time, we observe that relying on NTU-120_minus_60 as source dataset performs better than the Kinetics-400, which is expected because it is more similar in domain to our target dataset, see also Figure 2.2. Hence, it provides features which are easier to match during transfer. Our approach is agnostic to the source dataset for knowledge transfer, but the more similar the domain between source and target dataset, the better the action classification results.

2.5.3 Modality Generalization

We now change the student modality to 3D-skeleton data to assess to what extent our method generalizes over modalities. As discussed in Section 2.3.2, the video-to-video level strategy is needed for knowledge transfer to 3D-skeleton data. Table 2.5 shows

	Target-Modality: 3D-skeleton		
Source-Modality & Domain	20 per-class	50 per-class	100 per-class
From-scratch	33.00±3.0	50.11±2.0	67.50±1.5
Kinetics-RGB	52.15±1.0	65.86±1.0	74.98±0.4
Kinetics-Flow	52.39±2.0	66.11±0.5	75.46±0.2
NTU_120_minus_60-RGB	57.53±1.5	70.30±0.5	77.95±0.5
NTU_120_minus_60-Flow	58.57±1.5	71.11±0.6	78.59±0.4

TABLE 2.5: **Modality Generalization.** Performance of the 3D-skeleton student as the target modality. From-scratch indicates training the 3D-skeleton network directly on the target modality. The video-to-video strategy is used for the transfer. All models are evaluated the on cross-view validation set of NTU RGB+D. Our method generalises to a difficult 3D-skeleton target modality.

that our feature-supervised knowledge transfer improves the performance for the 3D-skeleton modality as well, especially for limited amounts of labeled data. Again we observe the optical-flow from a similar domain (NTU-120_minus_60) acts as the best source modality for transfer. We also observe that the performance increase is not as good as the transfer to depth maps (compare with Table 2.4). This is because completely dissimilar modalities (image-based RGB and optical-flow *vs.* joint-based 3D-Skeleton poses) and different network architectures (a 3D-CNN teacher and a graph-CNN student) are involved in this transfer. This makes it harder to match the features as compared to depth maps (image-based and 3D-CNN student). In summary, our method also generalizes to a much more difficult target modality such as 3D-skeleton data.

2.5.4 Task Generalization

In this experiment we evaluate our method for the task of temporal action detection where the goal is to predict the start and end locations of multiple activities in a long untrimmed depth video, along with the action classes. The knowledge transfer step remains the same as before, however, during fine-tuning the student network is now optimized for the task of action detection using (a limited amount of) labeled examples. As before, the unlabeled modality pairs from the training set of NTU RGB+D are used for the feature-supervised knowledge transfer. For fine-tuning we now sample from the labeled training set of PKU-MMD.

Action Detection Student Network. We rely on the R-C3D network [257] for this task with depth maps as the input. The architecture contains a backbone network which is connected to a region proposal network and a classification network. The student network is first trained for the knowledge transfer separately as before, and, then acts as the backbone network in the R-C3D framework during fine-tuning. We follow the same training procedure for the knowledge transfer as described in Section 2.4.3. For fine-tuning, the whole R-C3D framework is trained with all the hyperparameters

	Target-Modality: Depth		
Method	1/4 train-set	1/2 train-set	entire train-set
From-Scratch	52.85	66.45	73.39
RGB-pretraining	70.57	79.61	81.67
Flow-pretraining	73.68	81.21	81.72
RGB-feature-supervised	78.89	82.73	85.95
Flow-feature-supervised	79.87	84.85	86.81

TABLE 2.6: **Task Generalization.** Temporal action detection results (mAP@IoU=0.5) on PKU-MMD from depth maps using an R-C3D network. From-scratch indicates training the action detection network directly on the target modality without any pretraining. Pretraining indicates the action detection backbone is pretrained with RGB/Flow modality on NTU-120_minus_60 dataset and then directly finetuned with the labeled untrimmed depth maps. Feature-supervised indicates the backbone network is pretrained via our cross-modal transfer with RGB/Flow model trained on NTU-120_minus_60 as the teacher network, followed by fine-tuning with the labeled untrimmed depth maps. We use a quarter (235), half (470) or the entire (942) video train set for fine-tuning. All methods are evaluated on the validation set of the cross-subject split of PKU-MMD. Feature-supervised transfer outperforms both training from scratch and simple pretraining for the task of temporal action detection too.

from [257]. We train the network for a total of 8 epochs with a batch size of 4. The learning rate is initialized to 0.0001 and decreased to 0.00001 for the last 2 epochs. For inference and evaluation we follow the setting suggested in [257].

Action Detection Results. Table 2.6 shows the mean average precision (mAP at an IoU threshold of 0.5) on PKU-MMD for varying amounts of labeled training data (a quarter, half and all available training examples). We observe for all three splits there is a considerable gap in performance as compared to training from scratch as well as with simple RGB/Flow based pretraining, especially for the smallest split making cross-modal feature-supervised transfer beneficial when only limited amounts of start, end and class labels for the target modality are available. Thus, we can also use our pre-training strategy to reduce the number of labeled examples without any drastic drop in performance. Finally, we again observe the optical-flow proves to be the best modality for transfer.

2.5.5 Comparison with the state-of-the-art

We now compare our method with the state of the art methods for depth-based action recognition that rely on additional modalities for transferring knowledge during training, along with using the labeled depth examples. Figure 2.3 shows the results for action classification from depth maps on the NTU RGB+D dataset. The method by Garcia *et al*. [66] (MDMS) uses a four-step process that relies on labeled RGB-depth pairs from NTU RGB+D to distill knowledge from the RGB stream to the depth

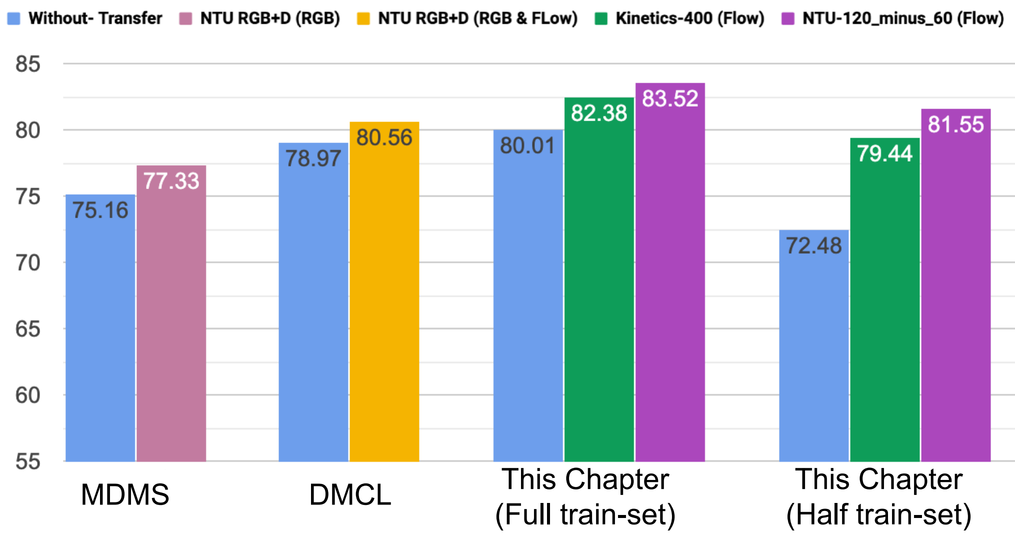


FIGURE 2.3: **Comparison with the state-of-the-art** for depth based action recognition on NTU RGB+D for the cross-view split. Without-Transfer indicates using only depth maps from the training set. For the rest, each column represents performance of respective method by distilling knowledge from one or more RGB action models trained on a source dataset (as shown in legend). Each method utilizes full-training set of labeled depth maps from the training set of NTU RGB+D, except the last one that uses only half of the same train set. Our method boosts the performance for depth modality while distilling from a different source dataset. Also, when we reduce the number of labeled examples for the depth modality by half, our method has no drastic decrease in performance.

stream. They achieve a 2% improvement over their baseline (*i.e.*, training a depth stream without any knowledge transfer). Similarly, another method by Garcia *et al.* [67] (DMCL) relies on labeled RGB-optical-flow-depth triplets from NTU RGB+D to train three networks together, such that, the RGB and the optical-flow streams are used to transfer knowledge to the depth stream during training. They achieve a boost of 1.5% over their baseline. Both methods use the full training set of labeled depth maps and other source modalities (also with labels) from NTU RGB+D to achieve this performance.

Our method relies on unlabeled RGB-depth or optical-flow-depth pairs from NTU RGB+D and pre-trained action models from other RGB datasets (like Kinetics-400 and NTU-120_minus_60) to distill knowledge to a depth stream. Followed by fine-tuning with labeled depth examples from the NTU RGB+D training set. For the full train set, we achieve a boost of around 2% and 3% over our baseline by transferring from the optical-flow teachers trained on Kinetics-400 and NTU-120_minus_60, respectively. We further show by reducing the number of labeled depth examples from NTU RGB+D for fine-tuning by half, the performance drop is small. Thus, in this way our method can act as pre-training source for reducing the number of labeled examples for non-RGB modalities such as depth maps. In other words, our method is particularly useful for the scenario where labeled examples for the target dataset are scarce and available in the target action modality only.

2.6 Conclusion

In this chapter, we presented a method to train action models for a non-RGB target modality, such as depth maps or 3D-skeletons, by extracting knowledge from a large-scale action labeled RGB dataset. Unlabeled pairs of the RGB and target modality are leveraged for cross-modal knowledge transfer by feature-supervision. Our extensive evaluation showed that we can use pre-trained RGB action models (particularly optical-flow from a more similar domain) to transfer knowledge to recognize and detect new actions from other action modalities like depth maps and 3D-skeleton sequences. In conclusion, we showed how large RGB action datasets can be used as valuable pre-training source for other non-RGB action datasets with limited labeled examples.

Chapter 3

Skeleton-Contrastive 3D Action Representation Learning

3.1 Introduction

The goal of this chapter is to learn a latent feature space suitable for 3D human action understanding. Different from traditional RGB frames [121, 113], skeleton data consists of 3D coordinates representing the major joints of each person in a video [196, 141, 34]. It offers a light-weight representation that can be processed faster and in a privacy-preserving manner providing application potential in video surveillance, assisted living, gaming and human-computer interaction. Moreover, when compared to RGB, such a representation is robust to changes in background and appearance [143, 278]. However, learning a good feature space for 3D actions requires large amounts of labeled skeleton data [51, 86, 278, 205, 260, 269, 207], which is much harder to obtain than large amounts of labeled RGB video. To address this major shortcoming, we propose a new self-supervised contrastive learning method for 3D skeleton data.

Several previous works also considered self-supervised learning for 3D skeleton data [286, 211, 136, 165]. These works design pretext tasks, such as learning to reconstruct masked input [286] and motion prediction [136], which still require the features to represent variations such as the viewpoint and skeleton scale, rather than focusing on higher-level semantic features relevant to downstream tasks. Instead, we take inspiration from recent self-supervised literature for RGB images, which aims to learn the high-level similarity between augmented forms of the same image and the dissimilarity between these and other images [167, 85, 29]. At the core of such contrastive learning is the nature of the RGB data, where each sample contains abundant pixel information, allowing for augmentations like spatial-cropping and color-jittering to easily generate subtly different versions of an image without changing its semantic content. However, skeleton sequences are much more sparse than RGB data and the augmentations commonly applied to images would not change the estimated skeleton of a person. Thus, for contrastive learning with skeleton sequences, we need skeleton-specific augmentations to encourage the learned features to encode information relating to spatio-temporal dynamics of the joints. We also want to enrich the input space which can be sampled from, to increase the variety of samples with

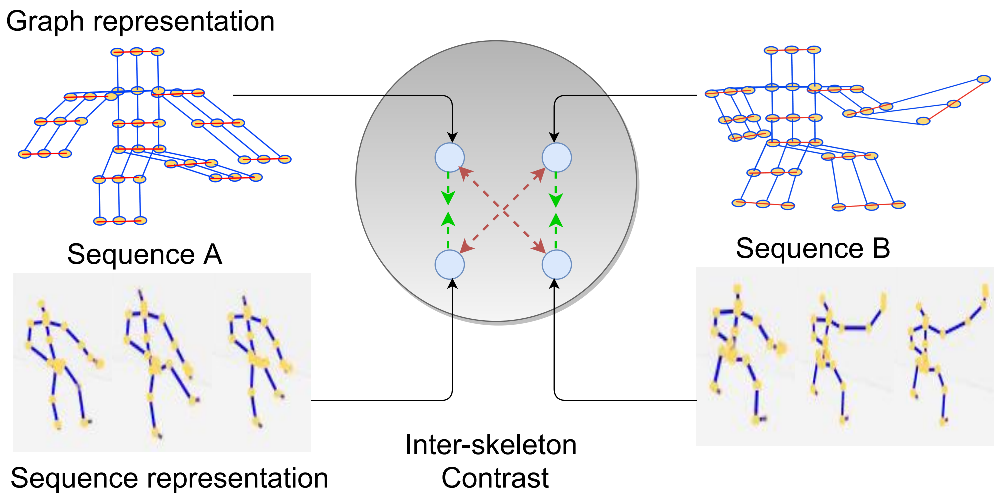


FIGURE 3.1: **Inter-skeleton contrast** learns high-level semantics of skeleton data in a self-supervised fashion. While contrastive methods normally learn invariance to augmentations we additionally learn invariance to the input representation. Different representations of the same sequence are encouraged to be close together in the feature space, while being far away from other sequences.

the same semantic content, and thus increase the difficulty of the contrastive learning task.

We make three contributions. Our first contribution is to leverage multiple input-representations of the 3D-skeleton sequences. In particular, we propose inter-skeleton contrast to learn from a pair of skeleton-representations in a cross-contrastive fashion, see Figure 3.1. This allows us to enrich the sparse input space and focus on the high-level semantics of the skeleton data rather than the nuances of one specific input representation. Second, we introduce several skeleton-specific spatial and temporal augmentations for generating positive pairs which encourage the model to focus on the spatio-temporal dynamics of skeleton-based action sequences, ignoring confounding factors such as viewpoint and the exact joint positions. Finally, we provide a comprehensive evaluation of our learned feature space on various challenging downstream tasks, showing considerable improvement over prior methods in all tasks.

3.2 Related Work

Self-Supervised Learning. Self-supervised learning aims to learn feature representations without human annotation, typically by solving *pretext tasks* which exploit the structure of unlabeled data. Previous works have proposed a variety of such tasks for learning image representations, *e.g.* solving spatial jigsaw puzzles [166], rotation prediction [72], spatial context-prediction [46], image inpainting [170] and colorization [279, 280]. Similarly, pre-text tasks have been designed for learning video representations, such as spatio-temporal puzzles [114], prediction of frame-order [64], clip-order [255], speed [15], future [79] and temporal coherence [124]. Such pretext tasks rely on the rich structured nature of RGB data with the hope that by learning to solve these tasks the encoded features will rely on the high-level semantics of the image or video and are thus applicable to the downstream task(s). Unfortunately, these existing RGB-based pretext tasks are not suited for 3D-skeleton sequences which have a simple structure and are less rich in information.

Instead of designing specific pretext tasks, recent self-supervised methods rely on instance discrimination and learn the similarity between sample pairs [167, 29, 85, 226, 156]. A noise contrastive loss learns invariances to certain image or video transformation functions, resulting in good feature representations. For example, Chen *et al.* [29] show that learning invariance to simple image augmentations, such as color jitter, results in highly discriminative features. He *et al.* [85] propose a momentum contrast which is able to utilize a large number of negatives for the noise contrast by storing image features from previous batches in a dynamic queue. In this paper, we rely on contrastive estimation for 3D action representation learning. As existing works use augmentations specific to RGB images, we introduce three skeleton-specific augmentations to generate positive pairs for learning the spatio-temporal dynamics of 3D-skeleton sequences. Furthermore, we propose inter-skeleton contrastive learning which additionally aims to learn invariance to the particular input representation of the 3D-skeleton sequences.

Supervised 3D Action Recognition. Numerous methods for supervised 3D action recognition exist. While earlier methods design handcrafted features [148, 234, 233] to model geometric relationships between skeleton joints, recent approaches rely on data-driven deep neural networks. Three skeleton-representations have become popular for deep learning. Sequence-based treats the 3D-skeleton data as a multi-dimensional time-series and models it with a recurrent architecture [142, 205, 278, 140, 196] to learn the temporal dynamics of the joints. Image-based create a pseudo-image representation of the 3D-skeleton data [51, 208, 143, 86, 128] which is encoded by CNN architectures to model the co-occurrence of multiple joints and their motion. Finally, graph-based [260, 206, 129, 32, 145, 199, 95, 174] represents the 3D-skeleton data with a graph consisting of spatial and temporal edges. Graph-convolutional architectures then encode the spatio-temporal motion from the human skeleton graph. Although these methods achieve excellent performance, they are all fully supervised and require time-consuming action class annotations. We propose a self-supervised method for 3D-skeleton data that leverages the diversity of the skeleton-representations to learn highly discriminative features from unlabeled data.

Self-Supervised 3D Action Recognition. Overcoming the need for large amounts of annotations has only recently received attention in the 3D action recognition community. Zheng *et al.* [286] propose a seq2seq model that learns to reconstruct masked input 3D-skeleton sequences. In particular, a GAN is trained such that the decoder attempts to regenerate the input sequences, while a discriminator measures the quality of the regenerated sequences. Similarly, Nie *et al.* [165] propose a cross-view reconstruction task that relies on a siamese denoising autoencoder to reconstruct the correct version of corrupted and rotated input skeletons. Su *et al*. [211] also propose a seq2seq model that regenerates input skeleton sequences. To encourage the encoder to learn better latent features, the decoder is weakened by fixing its weights.

Lin *et al*. [136] take a different approach and propose multi-task self-supervised learning for the sequence-based skeleton representation. Their framework solves multiple pretext tasks simultaneously, such as motion prediction and skeleton-jigsaw. Si *et al*. [201] propose an adversarial self-supervised learning approach that couples the self-supervised learning and the semi-supervised scheme via neighbor relation exploration and adversarial learning.

Different from all these works, we do not rely exclusively on a sequence-based skeleton-representation and pretext tasks such as input-reconstruction and motion prediction. Instead, we propose to exploit the diversity of skeleton-representations in an inter-contrastive learning regime and design skeleton-specific spatial and temporal augmentations for use in this contrastive method.

3.3 Skeleton-Contrastive Learning

In this section we present our inter-skeleton contrast approach for self-supervised learning of 3D action features. Contrastive methods aim to learn a good feature space by learning the similarity between augmented views of the same data. Since augmentations in existing contrastive learning works are primarily designed for RGB images [29] they are not suitable for the skeleton data that is considered in this work. Therefore, we first propose several skeleton-specific augmentation functions in Section 3.3.1. These augmentations enable us to apply existing contrastive learning methods, such as MoCo [85], to skeleton data. We describe this is Section 3.3.2.

However, contrastive learning can be vulnerable to shortcuts, where simple features, irrelevant to the downstream task, may be enough to identify the different augmented views of the same data. For instance, Chen *et al*. [29] show that color distributions can be a shortcut to identify different crops from the same image. To avoid such shortcuts and make the contrastive learning task more difficult, we additionally contrast pairs of different input skeleton representations with each other. We call this *inter-skeleton contrastive learning* and detail our approach in Section 3.3.3.

3.3.1 Skeleton Augmentations

The goal of contrastive learning is to learn the semantic similarity between items in a dataset without labels. This is usually done by learning the similarity of two augmented views (positive pairs) of a sample X. A data augmentation function D, composed of a single or multiple transformations, creates the augmented views. Hence, the network learns features for X, which are invariant to the transformations in D. The nature of the data X and the downstream task determines the appropriate invariances that the learned features should possess. In our case, X is a 3D-skeleton sequence, where each sequence represents a particular spatial configuration of human joints and its motion

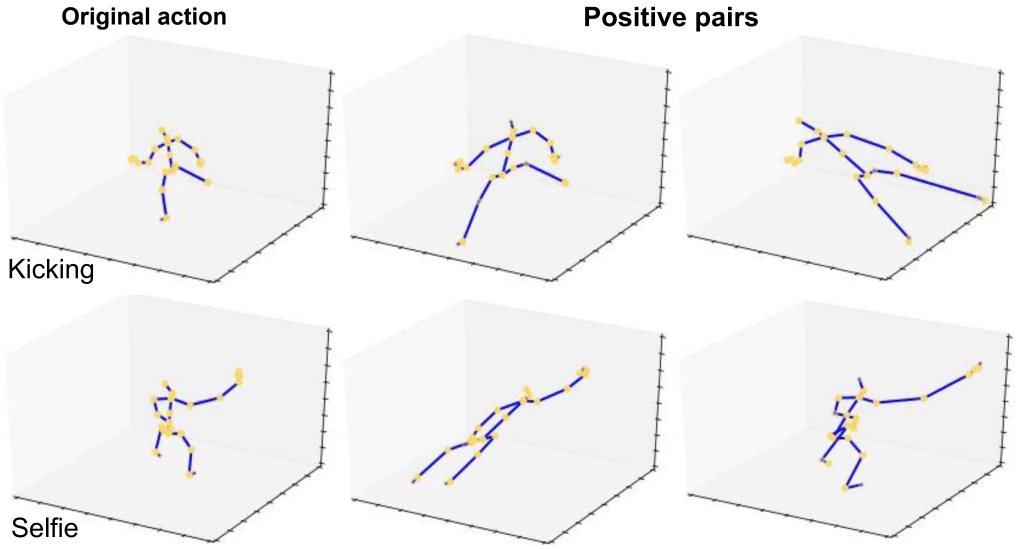


FIGURE 3.2: **Spatial pose augmentation** examples. A shear operation is applied to the original action so that the augmented pairs differ in viewpoint and camera distance.

over a short period of time. Thus, to learn useful representations for 3D-skeleton data, the commonly used RGB augmentations, such as color-distortion and Gaussian blurring [29], are not suitable. Instead, we need to learn invariances to transformations that encode the spatial and temporal dynamics of 3D skeleton action sequences. We introduce multiple spatial and temporal skeleton augmentation techniques to generate positive pairs for 3D-skeleton action sequences: *Pose Augmentation*, *Joint Jittering* and *Temporal Crop-Resize*. We then combine these to create our final spatio-temporal skeleton augmentation. Let us assume each raw action sequence $X \in R^{T \times J \times 3}$ consists of 3D coordinates of J body joints in T consecutive video frames. We define our individual augmentations D based on X.

3.3.1.1 Spatial Skeleton Augmentations

To apply our learned feature space to downstream tasks such as 3D action recognition, we require the feature encodings to rely on more discriminatory spatial semantics like joint configurations, while being invariant to factors such as viewpoint, camera distance, skeleton scale and joint perturbations. Existing augmentations for RGB images would not achieve this, thus we propose two new skeleton-specific spatial augmentations: pose augmentation and joint jittering. These can be applied to each of the T skeletons in the sequence X so a contrastive learning framework can learn invariance to these augmentations.

Pose Augmentation. With this transformation, we aim to create positive pairs which differ in viewpoint and distance to the camera, while retaining the same pose from the

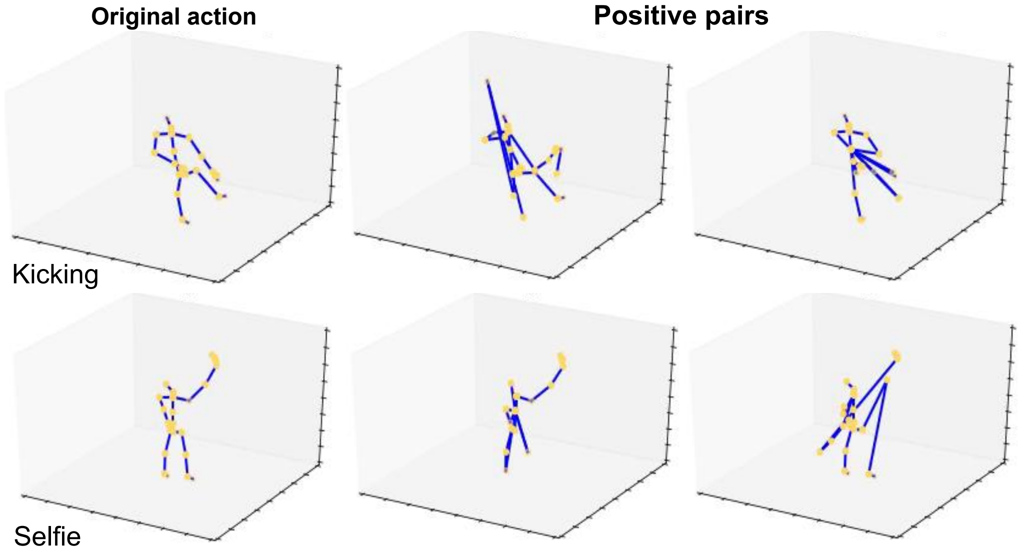


FIGURE 3.3: **Spatial joint jittering** examples. The augmented pairs contain a subset of common joint connections while other joint connections are randomly moved to an irregular position.

original sequence. To achieve this, we apply a 3D shear on the action sequence X:

$$D_{Spatial_1}(X) = X \cdot \begin{bmatrix} 1 & r_{01} & r_{02} \\ r_{10} & 1 & r_{12} \\ r_{20} & r_{21} & 1 \end{bmatrix}, \tag{3.1}$$

where the elements of the augmentation matrix are randomly drawn from a uniform distribution $[-1, 1]$. Figure 3.2 shows several examples. By applying the same shearing operation to each joint of the skeleton at each time-step in the sequence we are able to simulate changes in camera viewpoint and distance between the subject and camera. Therefore, a contrastive network which learns invariance to this transformation is forced to learn more discriminatory pose semantics of the positive pairs and ignore redundant information such as the viewpoint and proximity to the camera.

Joint Jittering. We also want a contrastive method to be invariant to noise in the estimated skeleton. Therefore we propose joint jittering to create positive pairs where some of the joint connections in X are randomly perturbed. We select j of the J joints at random and move these joints to irregular positions, while keeping other joints in their original position. The transformation is defined as:

$$D_{Spatial_2}(X) = X[:, j] \cdot \begin{bmatrix} r_{00} & r_{01} & r_{02} \\ r_{10} & r_{11} & r_{12} \\ r_{20} & r_{21} & r_{22} \end{bmatrix}, \tag{3.2}$$

where j is a subset of the joints such that $|j| < J$, and the elements of the jitter matrix are randomly drawn from a uniform distribution $[0, 1]$. The same jitter matrix

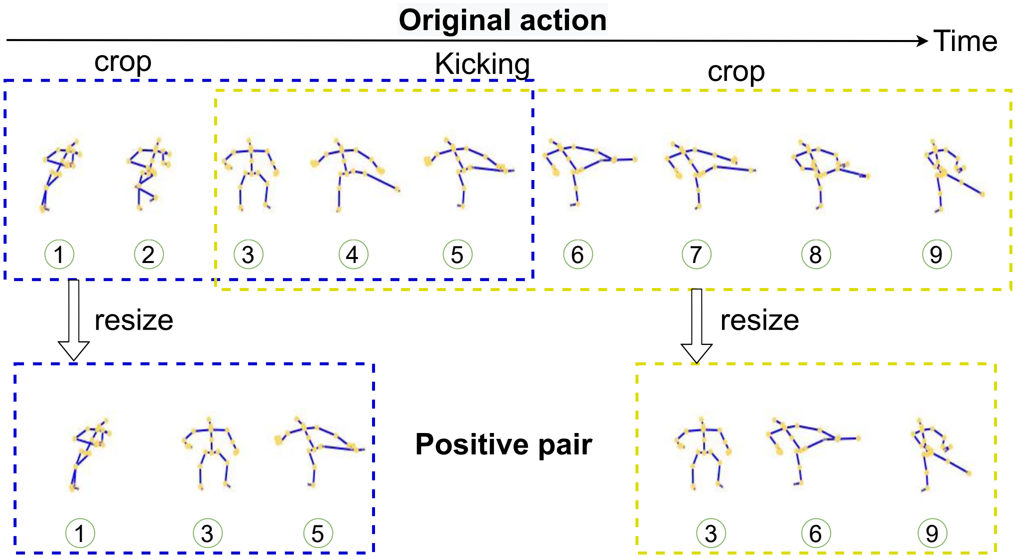


FIGURE 3.4: **Temporal crop-resize**. The augmented views start at different time steps and sample different temporal periods (blue and yellow boxes). Each crop is re-sampled to a fixed size, effectively altering its speed depending on the length of the temporal crop.

is applied to each joint in j at each time-step T. Examples are shown in Figure 3.3. To learn invariance to such transformations, the contrastive task is encouraged to rely on the spatio-temporal semantics of the common joint connections and ignore the noise from the irregular joint connections.

3.3.1.2 Temporal Skeleton Augmentation

Besides the spatial perturbations, a good 3D skeleton feature space should also be robust to temporal modifications of the skeleton sequences, such as the speed of an action and changes to the temporal bounds of the sequence. To this end, we propose temporal crop-resize.

Temporal Crop-Resize. In this transformation, we create positive pairs with varying speed and varying starting and ending points. We sample different parts of the action sequence X via a random crop and resize this crop over the temporal dimension T:

$$D_{Temporal}(X) = \text{Interpolate}(X[L_{start} : L_{start} + TL_{ratio}]). \tag{3.3}$$

The length ratio L_{ratio} is first randomly sampled from distribution $[l_{min}, 1.0]$, followed by randomly selecting a starting frame L_{start} between $(0, T - TL_{ratio})$. The sub-sequence $X[L_{start} : L_{start} + TL_{ratio}]$ is then re-sampled to a fixed length. This re-sampling causes the temporal crop-resize to also alter the speed of a sequence as well as its start and end times; a shorter sub-sequence will effectively have a slower speed once re-sampled. Figure 3.4 shows examples of this transformation. By including this augmentation the contrastive task is forced to focus on the commonalities of the joint motion dynamics over the sampled temporal periods and be robust to changes in the exact start, end and speed of an action.

3.3.1.3 Spatio-Temporal Skeleton Augmentations.

To learn the spatio-temporal dynamics of the skeleton sequences, we propose to combine the above spatial and temporal transformations into a single augmentation function. Such composition results in strong positive pairs which vary in both spatial and temporal dynamics locally, while retaining the high-level semantics of the original action sequence. In particular, we first apply the temporal crop-resize augmentation $D_{Temporal}$ on the original action sequence X followed by a spatial augmentation $D_{Spatial_i}$ to the resulting sequence:

$$D_{Spatio\text{-}Temporal}(X) = D_{Spatial_i}(D_{Temporal}(X)) \tag{3.4}$$

Here, i can either be fixed to the pose augmentation or the joint jitter or randomized to select either of the spatial augmentations. As we will show in the experiments, learning invariance to spatio-temporal transformations produces a better 3D action feature space and randomizing the composition further improves the result.

3.3.2 Intra-Skeleton Contrast

Before describing our proposed inter-skeleton method, we first describe how the above augmentations can be incorporated into an existing contrastive method, such as MoCo [85], with a single input skeleton-representation. We call this intra-skeleton contrastive learning. Each raw action sequence $X \in R^{T\times J\times 3}$ is first augmented into two different views X_q and X_k (called query and key) via a data augmentation function D. Both views of the skeleton data are then instantiated into the same skeleton-representation, be it image-based or sequence-based or graph-based. A contrastive method such as MoCo uses two encoders, one for the query and one for the key. We refer to the query encoder as f_q and the key encoder as f_k. Let $(Z_q, Z_k) = (f_q(X_q), f_k(X_k))$ be output embeddings of the encoders for the input query-key pair. We then train the contrastive network using the noise contrastive estimation loss InfoNCE [167]:

$$\mathcal{L}(X) = -\log \frac{\exp(Z_q \cdot Z_k/\tau)}{\exp(Z_q \cdot Z_k/\tau) + \sum_{Z_n \sim \mathcal{N}} \exp(Z_q \cdot Z_n/\tau)}, \tag{3.5}$$

where τ is a temperature softening hyper-parameter and $\mathcal{N}$ is the current set of negatives that are stored in a dynamic queue via previous states of the key encoder f_k as in [85]. Only the query encoder is actively trained using Equation 3.5 and the key encoder is updated as a moving average of the query encoder. This trains the framework to learn 3D action features which are invariant to the transformations in D for the chosen skeleton-representation.

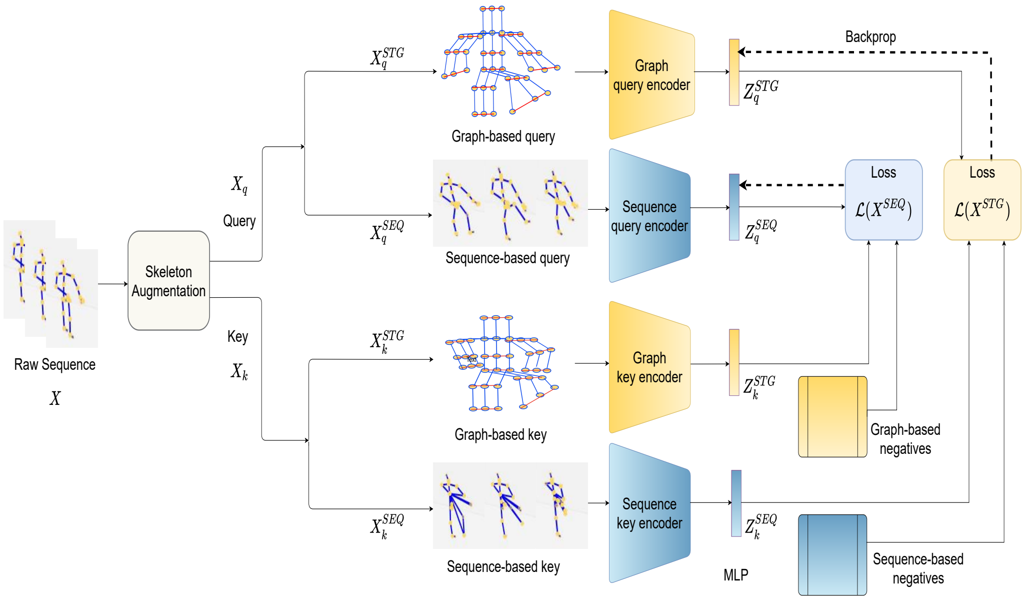


FIGURE 3.5: **Inter-skeleton contrast**. We learn invariances to input skeleton representations, as well data augmentations, in a cross-contrastive manner. We first augment the input sequence into two different views called the query and key using our proposed spatio-temporal augmentations. Each of these views is then represented with two different input skeleton-representations, here graph-based and sequence-based. We encourage the embedding for the graph-based query to be similar to the embedding of the sequence-based key while being dissimilar to the current set of sequence-based negatives. The same applies for the sequence-based query and graph-based key and negatives.

3.3.3 Inter-Skeleton Contrast

Up to this point, our method, like previous contrastive learning approaches [29, 85, 156], learns the similarity between different augmented forms of the same input. We now extend contrastive learning for 3D skeleton data beyond these augmentations and propose inter-skeleton contrast which aims to learn invariance to the *input representation* of the skeleton sequence. Three 3D-skeleton representations are common: *image-based* as a $T \times J$ pseudo-image where the 3D coordinates of each joint are the image channels, *sequence-based* as a multi-dimensional time series, or *graph-based* as a spatio-temporal graph. Each requires a different network architecture and encodes different characteristics of the sequence. For example, RNNs treat skeleton sequences as a time series and explicitly model the temporal evolution of joints, while GCNs treat sequences as a graph with both spatial and temporal edges and thus explicitly encode human pose as well as each joint's temporal motion. While the action depicted by the skeleton sequence is the same, the way the input sequence is represented and encoded is different. To learn invariance to the input representation the contrastive framework has to learn the similarities between the characteristics of these different representations as well as our data augmentations which will result in more discriminative features.

The overall network is depicted in Figure 3.5. The raw skeleton sequence is first augmented into two views as in Section 3.3.2. Each view is then represented in two ways, in this case with a graph-based representation and a sequence-based representation. We refer to the different representations of the raw action sequence X as X^{IMG} for image-based, X^{SEQ} for seq-based and X^{STG} for graph-based. For the rest of this section we will take the example of the pair X^{SEQ} and X^{STG} as displayed in Figure 3.5. We adapt our model to contrast the different input representations by using a pair of momentum contrastive models together, one for each input-representation X^{SEQ} and X^{STG}. In particular, the model now consists of two query encoders f_q^{SEQ} and f_q^{STG} and two key encoders f_k^{SEQ} and f_k^{STG}. A query-key pair (X_q, X_k) is obtained by augmenting a raw action sequence X with D as before. We instantiate two different skeleton-representation pairs (X_q^{SEQ}, X_k^{SEQ}) and (X_q^{STG}, X_k^{STG}). Then, for the query in each input representation, we generate the positives and negatives from the key encoder of the *other* input representation and vice versa. The encoders (f_q^{SEQ}, f_q^{STG}) are trained jointly using a cross-contrastive loss function:

$$\mathcal{L}(X^{SEQ}, X^{STG}) = \mathcal{L}(X^{SEQ}) + \mathcal{L}(X^{STG}), \tag{3.6}$$

$$\mathcal{L}(X^{SEQ}) = -\log \frac{\exp(Z_q^{SEQ} \cdot Z_k^{STG} / \tau)}{\exp(Z_q^{SEQ} \cdot Z_k^{STG} / \tau) + \sum\limits_{Z_n \sim \mathcal{N}^{STG}} \exp(Z_q^{SEQ} \cdot Z_n^{STG} / \tau)}, \tag{3.7}$$

$$\mathcal{L}(X^{STG}) = -\log \frac{\exp(Z_q^{STG} \cdot Z_k^{SEQ} / \tau)}{\exp(Z_q^{STG} \cdot Z_k^{SEQ} / \tau) + \sum\limits_{Z_n \sim \mathcal{N}^{SEQ}} \exp(Z_q^{STG} \cdot Z_n^{SEQ} / \tau)}, \tag{3.8}$$

where $Z_q^{SEQ} {=} f_q^{SEQ}(X_q^{SEQ})$ is the embedding of the sequence-based query and $\mathcal{N}^{SEQ}$ is the current set of negative sequence-based embeddings. These are defined similarly for the other representations and augmentations of X. This formulation serves two purposes. First the input space of the contrastive task is enriched to learn from multiple representations of the same sequence, in addition to the multiple 'views' the data augmentation D provides. Second, different from Equation 3.5, the cross-contrastive loss *i.e.*, Equation 3.6 forces the framework to rely on mutual information between the embeddings of the two skeleton representations. Thus the contrastive framework is encouraged to focus on higher-level semantics and avoid resorting to shortcut solutions to identify the similarity between query-key pairs.

3.4 Experiments

We first describe the datasets and implementation details. We then demonstrate the effectiveness of our contrastive learning approach on several 3D action understanding downstream tasks. Finally, we ablate the effects of our proposed skeleton augmentations and inter-skeleton contrast.

3.4.1 Datasets and Evaluation

NTU RGB+D 60 [196]. This is the most commonly used dataset for 3D action recognition. All actions are captured in indoor scenes with three cameras concurrently. The dataset contains 40 different subjects and 60 action classes. Each action sequence is performed by an individual or pair of actors with each actor represented by the 3D coordinates of 25 skeleton joints. The dataset consists of 56,880 video samples and is evaluated under the two standard protocols as suggested by [196]. The first is *cross-view*, where samples from two angles ($0°, 45°$) are used for training (37,920 samples) and a third angle ($-45°$) is used for testing (18,960 samples). The second is *cross-subject*, where the actors in the training and testing sets are different, with 40,320 training and 16,560 testing samples.
NTU RGB+D 120 [141]. This is an extension to NTU RGB-D 60 and is currently the largest benchmark for 3D action recognition with 114,480 samples over 120 action classes. Actions are captured with 106 subjects in a multi-view setting using 32 different setups (varying camera distances and background). Each action sample has 1 or 2 subjects, and each is represented by 25 3D-skeleton joints. The dataset is challenging due to the variation in subject, background, viewpoint and fine-grained actions captured. For evaluation, two recommended protocols [141] are used: *cross-setup*, where even-numbered setups are used for training (54,471 samples) and odd-numbered setups are used for testing (59,477 samples), and again *cross-subject*, with 63,026 training and 50,922 testing samples.
PKU-MMD [34]. This dataset was originally proposed for action detection but has also been used for action recognition [136]. It contains 52 human action classes. Each action is represented by the 3D coordinates of the 25 joints of each actor involved in the action. The dataset consists of two parts: ***PKU-MMD I*** and ***PKU-MMD II***, with almost 20,000 and 7,000 action instances. Both parts are challenging for action recognition, as the number of action classes is large while the training sets are relatively small, however PKU-MMD II is more challenging due to the large view variation causing more skeleton noise. We split both sets into a training and a testing set using the recommended *cross-subject* protocol [34]. The training sets of PKU-MMD I & II contain 18,841 and 5,332 samples, while the testing sets contain 2,704 and 1,613 samples.
Evaluation Criteria. For all datasets, protocols and downstream tasks we report the top-1 accuracy.

3.4.2 Implementation Details

Network Architectures. We instantiate the encoder pairs (f_q, f_k) based on the skeleton-representations used. For the sequence-based encoder f^{SEQ} we rely on a 3-Layer BI-GRU with H=1024 units per layer [211]. For the image representation encoder f^{IMG}, we adopt the CNN based Hierarchical Co-occurrence Network (HCN) [128]. For the graph representation encoder f^{STG}, a joint based graph-convolutional network A-GCN [199] is used. We represent each skeleton sequence X as two people, with the second actor being all zeros for single actor actions. The augmented forms of the raw skeleton sequence X (X_q and X_k) have temporal length 64. Unless mentioned otherwise we use $|j| = 15$ for the joint jitter augmentation and $l_{min} = 0.1$ for the temporal crop-resize augmentation.
Self-Supervised Pretraining. Our inter-skeleton contrastive network is based on MOCO [85] and is trained on the training data without any labels. A projection head (an MLP) is appended to each encoder to produce embeddings of a fixed size of 128. The embeddings are L2-normalized before computing the contrastive loss. We train the whole network with a temperature value of τ=0.07, an SGD optimizer, a learning rate of 0.01 and a weight decay of 0.0001. For NTU RGB+D 60 & 120, the size of the set negatives $\mathcal{N}$ is 16,384 and the model is pre-trained for a total of 450 epochs. For PKU-MMD I & II, the size of $\mathcal{N}$ is set to 8,192 and 2,048, and we pre-train for 600 epochs. The training and evaluation details of the downstream tasks are discussed in the Appendix A. Code is available at https://github.com/fmthoker/skeleton-contrast.

3.4.3 Downstream Tasks

In this section, we evaluate the 3D action features learned by our inter-skeleton contrast for various downstream tasks in comparison with the respective state-of-the-art in self-supervised learning. For a fair comparison we follow the setups of prior works and only train and evaluate downstream tasks with the sequence-based input representation X^{SEQ}. In particular, we pre-train our inter-skeleton contrast network with X^{SEQ} and X^{STG} skeleton representations as this gives the best result (see Section 3.4.4) and evaluate only the sequence-based query encoder f_q^{SEQ}. We also show some qualitative results in the Appendix A.

3.4.3.1 3D Action Recognition.

We compare our method to prior works in self-supervised learning for skeleton data by training a linear classifier on top of the frozen features from our inter-skeleton contrast. We compare with the proposed methods of Zheng *et al.* [286], Su *et al.* [211] and Nie *et al.* [165], all of which use reconstruction of the skeleton sequence as a pretext task. We also compare to the multi-task self-supervised method by Lin *et al.* [136], which uses skeleton-jigsaw and motion prediction as auxiliary tasks.

We present results on the NTU RGB+D 60, NTU-120 and PKU-MMD (I and II) datasets in Table 3.1. It is evident our inter-skeleton contrast outperforms all methods

	NTU RGB+D 60		NTU RGB+D 120		PKU-MMD I	PKU-MMD II
	x-view	x-sub	x-setup	x-sub	x-sub	x-sub
Zheng *et al.* [286]	56.4	52.1	39.7	35.6	68.7	26.5
Lin *et al.* [136]	–	52.5	–	–	64.8	27.6
Su *et al.* [211]	59.3	56.1	44.1	41.1	59.9	25.5
Nie *et al.* [165]	79.7	–	–	–	–	–
This chapter	**85.2**	**76.3**	**67.9**	**67.1**	**80.9**	**36.0**

TABLE 3.1: **3D action recognition.** Our method learns better 3D-action features from unlabeled data than alternatives, no matter the dataset or evaluation protocol. All results of Zheng *et al.* and Su *et al.* obtained with code provided by Su *et al.*

	NTU RGB+D 60		NTU RGB+D 120	
	x-view	x-sub	x-setup	x-sub
Zheng *et al.* [286]	48.1	39.1	35.5	31.5
Su *et al.* [211]	76.3	50.7	41.8	39.5
This chapter	**82.6**	**62.5**	**52.3**	**50.6**

TABLE 3.2: **3D action retrieval.** Results for Zheng *et al.* and Su *et al.* in [211] obtained with code provided by Su *et al.* Our method learns best features for retrieval than prior self-supervised methods.

by a considerable margin on each benchmark. We conclude the self-supervised feature space learned by our method is state-of-the-art for 3D action recognition.

3.4.3.2 3D Action Retrieval.

We follow the setup introduced by Su *et al.* [211]. We apply the kNN classifier (k=1) to the pre-trained features of the training set to assign classes. We match each test sample to the most similar training class using cosine similarity. Besides comparison with Su *et al.* [211], we also compare with Zheng *et al.* [286], using numbers and code provided by Su *et al.* We present results for NTU RGB+D 60 and NTU RGB+D 120 in Table 3.2. For both datasets, our method outperforms the alternatives, especially for the more challenging cross-subject and cross-setup protocols. Both [286, 211] rely on an input reconstruction pretext-task for learning their feature space, which easily captures varying viewpoints. However, with a simple reconstruction, it is difficult to capture variation with respect to subjects and setups as our inter-skeleton contrast can.

3.4.3.3 Semi-Supervised 3D Action Recognition.

In semi-supervised setting, a network utilizes both labeled and unlabeled data during the training process. Following prior work for semi-supervised learning in 3D action recognition, we first train our encoder on our unsupervised inter-skeleton contrastive

	NTU RGB+D 60								**PKU-MMD I**	
	x-view				**x-sub**				**x-sub**	
	(1%)	(5%)	(10%)	(20%)	(1%)	(5%)	(10%)	(20%)	(1%)	(10%)
Zheng *et al.* [286]	-	-	-	-	35.2	-	62.0	-	34.4	69.5
Lin *et al.* [136]	-	-	-	-	33.1	-	65.1	-	36.4	70.3
Si *et al.* [201]	-	63.6	69.8	74.7	-	57.3	64.3	68.0	-	-
This chapter†	21.7 ±1.0	47.6 ±1.0	59.8 ±0.5	69.1 ±0.5	17.6 ±0.5	42.8 ±0.5	51.6 ±1.0	59.5 ±1.0	22.5 ±1.0	55.4 ±1.0
This chapter	**38.1** ±1.0	**65.7** ±0.5	**72.5** ±0.4	**78.2** ±0.3	**35.7** ±0.5	**59.6** ±0.5	**65.9** ±1.0	**70.8** ±1.0	**37.7** ±1.0	**72.1** ±1.0

TABLE 3.3: **Semi-supervised 3D action recognition.** We report average accuracy of five runs with random subsets of labeled samples. Pre-training with our inter-skeleton shows improvement over prior semi-supervised works as well as training only with the labeled subset (†).

learning task. Then, we fine-tune the final classification layer and the pre-trained encoder together using a portion of the data labeled with the action class. Again, we compare with Zheng *et al.* [286] and Lin *et al.* [136] as well as the method of Si *et al.* [201] on NTU RGB+D 60 and the PKU-MMD I datasets. To compare with prior works, we report results when using 1%, 5%, 10% and 20% of the training data with labels for NTU RGB+60 and when using 1% and 10% of the labels for PKU-MMD I. The rest of the training set is used as the unlabeled data.

The results in Table 3.3 reveal that our method outperforms all previous methods on each benchmark. We also demonstrate a large improvement over supervised only training (†), *i.e.* training with only the available labeled data from randomly initialized weights. From these results we can see that our inter-skeleton contrastive learning is especially suited to learn from both unlabeled and labeled skeleton data in order to boost the performance of 3D action recognition.

3.4.3.4 Transfer Learning for 3D Action Recognition.

To evaluate if knowledge gained from a source dataset generalizes to a different target dataset, we also consider transfer learning. In this setting, an encoder network is first trained on the source dataset for our inter-skeleton contrastive task, followed by jointly finetuning the pretrained encoder and a classifier on a target dataset for action recognition. As in Lin *et al.* [136], we use NTU RGB+D 60 and PKU-MMD I as the source datasets and PKU-MMD II as the target dataset. Table 3.4 shows our features are just as or more transferable than those of Zheng *et al.* [286] and Lin *et al.* [136], especially for transfer from PKU-MMD I to PKU-MMD II which are from same domain. Thus, the knowledge gained by our method from a source dataset can improve action classification accuracy on a different target set, especially one with a similar domain.

	Transfer to PKU-MMD II	
	PKU-MMD I	NTU RGB+D 60
Zheng *et al.* [286]	43.6	44.8
Lin *et al.* [136]	44.1	**45.8**
This chapter	**45.1**	**45.9**

TABLE 3.4: **Transfer learning for 3D action recognition.** All results by Zheng *et al.* provided by Lin *et al.* in [136]. Knowledge gained via inter-skeleton contrastive pretraining transfers well, especially when source and target datasets are more similar.

3.4.4 Ablation Studies

We now ablate the effect of each of our skeleton augmentations and demonstrate the effectiveness of our inter-skeleton contrastive learning. These ablations are performed on the cross-view protocol of NTU RGB+D 60 for the downstream task of 3D action recognition. As before, after pre-training the models with our contrastive self-supervision methods, we train a linear classifier with action labels on top of the frozen features of the query encoder f_q.

3.4.4.1 Benefit of Skeleton Augmentation.

First, we show the benefit of each of the proposed skeleton augmentations when learning from a single input skeleton representation. We choose as skeleton augmentation function D, either pose augmentation, joint jitter, temporal crop-resize or combinations thereof, and train an intra-skeleton contrastive model as described in Section 3.3.2.

Table 3.5 shows the accuracy of our augmentations with each input representation. We find that all of the proposed spatial and temporal skeleton augmentations individually perform better than using no augmentation. Thereby, reinforcing our claim that learning invariances to spatial changes like viewpoints, scale and joint perturbations, or, temporal changes such as delay and speed result in learning good action features. The composition of augmentations further improves the accuracy by a considerable margin for all input representations, with the best combination being the inclusion of all three augmentation functions. For example, the final accuracy with the X^{IMG} representation is a $\sim$10% increase over using only pose augmentation and $\sim$28% over using no augmentation.

The benefit of our proposed skeleton augmentations are also reflected in the contrastive pre-training plots in Figure 3.6, which demonstrate that without augmentation the contrastive task is too easy, resulting in early saturation of the loss and poor features. With our spatial and temporal augmentations the contrastive task becomes more difficult as the network is encouraged to focus more on the pose and spatio-temporal movements of the joints, thereby improving downstream accuracy. Thus the combination of all our augmentations result in learning our best 3D action features.

Augmentations			**Downstream Representation**		
Temporal Crop-resize	Pose	Joint Jitter	X^{IMG}	X^{STG}	X^{SEQ}
-	-	-	51.0	51.4	50.0
✓	-	-	62.5	53.5	64.1
-	✓	-	69.8	63.8	71.7
-	-	✓	74.6	66.1	75.2
✓	✓	-	73.2	69.3	73.8
✓	-	✓	77.0	68.3	80.0
✓	✓	✓	**79.6**	**72.5**	**82.5**

TABLE 3.5: **Benefit of skeleton augmentation**. We ablate the effect of our augmentations with 3D action recognition on NTU RGB+D 60. Combining all three augmentations generates strong positive pairs for increased accuracy, no matter the 3D action representation.

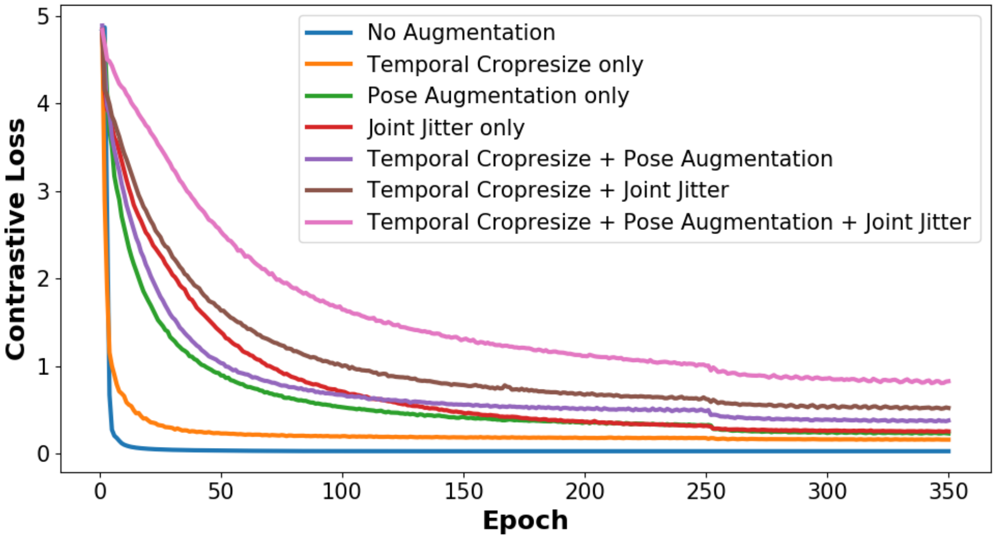


FIGURE 3.6: **Skeleton augmentation loss curves**. Our proposed spatial and temporal skeleton augmentations make the contrastive task more difficult which prevents early saturation of the loss. The network is forced to focus more on commonalities in pose and joint motion dynamics to learn the similarities.

3.4.4.2 Intra-Skeleton *vs.* Inter-Skeleton.

Next, we examine the effectiveness of learning two skeleton representations together in our inter-skeleton framework over learning from each input representation separately (intra-skeleton). While our inter-skeleton network pre-trains two input skeleton representations alongside one another, to allow for fair comparison to the intra-skeleton network we train and test the downstream action recognition model with each input representation separately. The results of combining multiple representations in

	Downstream Representation		
Pretraining	X^{IMG}	X^{STG}	X^{SEQ}
Intra (X^{IMG} only)	79.6	-	-
Intra (X^{STG} only)	-	72.5	-
Intra (X^{SEQ} only)	-	-	82.5
Inter (X^{IMG}, X^{STG})	80.0	78.0	-
Inter (X^{IMG}, X^{SEQ})	**81.7**	-	83.0
Inter (X^{SEQ}, X^{STG})	-	78.9	**85.2**
Inter (X^{IMG}, X^{SEQ}, X^{STG})	**81.2**	**81.6**	**85.4**

TABLE 3.6: **Intra-skeleton *vs.* Inter-skeleton**. Training alongside a second input representation in our inter-skeleton contrast results in better features for all input representations, regardless of the pair used. Note that a representation can only be used in the downstream task when it is present in pre-training. Ablation performed on 3D action recognition with NTU RGB+D 60.

downstream tasks are presented in Appendix A.

Table 3.6 shows the accuracy of our inter-skeleton contrast compared to the intra-skeleton baseline for each skeleton representation. We first observe that pretraining with any two skeleton representations side by side in our inter-skeleton contrast is considerably better than only learning with a single representation as in the intra-skeleton contrast. For example, the accuracy with X^{STG} increases by 6% when pre-trained together with X^{SEQ} in our inter-skeleton contrast model. A similar increase of 5% occurs when pre-training alongside X^{IMG}. We find this to be the case with each skeleton representation; regardless of the second representation it is trained alongside in the inter-skeleton contrast, there is an increase in performance. We also tried training all three skeleton representations together. While this does give the best result, the improvement is outweighed by the computational cost of training all three representations simultaneously. Overall, these results reinforce our claim that learning invariance to skeleton augmentations alone leads to sub-optimal features and learning additional invariance to skeleton-representations results in a better feature space.

3.5 Conclusion

In this chapter, we presented a method for self-supervised learning of 3D skeleton data. We design a contrastive learning framework that relies on novel skeleton augmentations and multiple skeleton-representations to learn spatio-temporal dynamics of the skeleton sequences. Our comprehensive evaluation with different skeleton augmentations and skeleton-representation pairs reveal that learning invariance to our spatio-temporal augmentations and contrasting sequence-based and graph-based representations with each other results in best action features. The final model achieves considerable performance gains and outperforms prior state-of-the-art in self-supervised learning for multiple downstream tasks on NTU RGB+D 60 & 120 and PKU-MMD.

Chapter 4

How Severe is Benchmark-Sensitivity in Video Self-Supervised Learning?

4.1 Introduction

Video self-supervised learning has progressed at a tremendous pace in recent years, *e.g.* [237, 1, 182, 177, 173, 171], as it offers a crucial starting point from which to learn. This is especially important for video understanding applications, where annotating large amounts of data is extremely expensive, error-prone and sensitive to annotator bias. Hence, learning video representations through self-supervision is crucial, especially for use cases where the downstream video data is limited because of the domain, task or actions the video contains. However, the majority of current works in video self-supervised learning, *e.g.* [255, 157, 158, 8, 169], do not test beyond standard benchmarks. The standard protocol is to use unlabeled Kinetics-400 [113] for pre-training and then measure performance by finetuning on two action recognition datasets: UCF-101 [209] and HMDB-51 [122]. While these benchmarks have facilitated the impressive progress of video self-supervised learning in recent years, they cannot indicate the generalizability of such methods as these pre-training and downstream datasets are all similar in appearance and the type of actions they contain. Some methods have started to report finetuning performance on additional datasets like Something-Something-v2 [74] in [237, 173, 60], Diving-48 [132] in [41, 243], AVA [77] in [253, 262, 60] and EPIC-Kitchens-100 [39] in [262]. However, such evaluations are insufficient to understand the generalization of video self-supervised methods alone since they only add a single additional dataset, often without comparison to prior methods.

In this chapter, we address the essential need to gauge the sensitivity of existing video self-supervised methods to the current benchmark by thoroughly evaluating their performance for generalization across diverse downstream settings. Similar benchmarking studies have been performed for self-supervised pre-training in images [36, 99, 120, 53, 73, 264, 117, 272, 7, 160, 191, 232, 54], which investigate model transferability [99, 53, 160, 236] or the importance of factors like pre-training dataset [36, 120, 73] and backbone architecture [117]. Unfortunately, lessons from these works do not directly transfer to video self-supervised learning. First, video self-supervised tasks are distinct from those of images as they are designed to understand the temporal dimension of video [173, 41, 237, 262] in addition to the spatial understanding needed in images [29]. Second, video is multi-modal and several methods [171, 8, 158] are designed to exploit cross or multi-modal understanding, which is again absent in image-based methods. For videos, [60] extends four image-based self-supervised methods to videos and investigate their performance focusing on different pre-training setups. We take inspiration from this and benchmarking works in image self-supervised learning and perform a much-needed study for understanding the generalizability of self-supervised methods for video in relation to different downstream factors.

As our first contribution, we identify the problem of benchmark-sensitivity in video self-supervised learning and examine this sensitivity along the factors of domain, samples, actions and task. As our second contribution, we perform an extensive evaluation which spans a total of over 500 experiments with 9 video self-supervised learning methods across 7 video datasets and 6 video understanding tasks. We find that standard benchmarks in video self-supervised learning do not indicate generalization along the said sensitivity factors and vanilla supervised pre-training outperforms self-supervised pre-training, particularly when domain change is large and there are only a few downstream finetuning samples available. Third, we propose a subset of our experiments as the SEVERE-benchmark for future self-supervised learning methods to benchmark generalization capability. We also discuss the implication of this benchmark for evaluating the generalizability of representations obtained by existing methods as well as the nature of video self-supervised objectives that currently generalize well.

4.2 Identifying Benchmark Sensitivity

The vast majority of current works in video self-supervised learning evaluate their approach by pre-training on Kinetics-400 [113] and finetuning the learned representation for action recognition on UCF-101[209] and HMDB-51[122]. Some works [171, 41, 217, 237, 173, 8, 68, 137, 94] also report performance on video retrieval for UCF-101 and HMDB-51 and several recent works [182, 262, 189] compare linear evaluation performance on Kinetics-400. However, these downstream datasets are very similar to each other and also share many similarities with the pre-training dataset of Kinetics-400. Videos in all three datasets are collected from YouTube and are mostly recorded with a single camera containing a single well-positioned human actor. In terms of class labels, all datasets focus on similar, coarse-grained and mutually exclusive actions with many actions common between pre-training and downstream datasets. Besides all these data similarities, the existing evaluations also ignore a major benefit of self-supervised representation learning for videos, *i.e.* finetuning the representation with only a small amount of data samples and transferring to other video understanding tasks beyond action recognition. Hence, we believe the current benchmark standard is insufficiently equipped to gain a true understanding of where

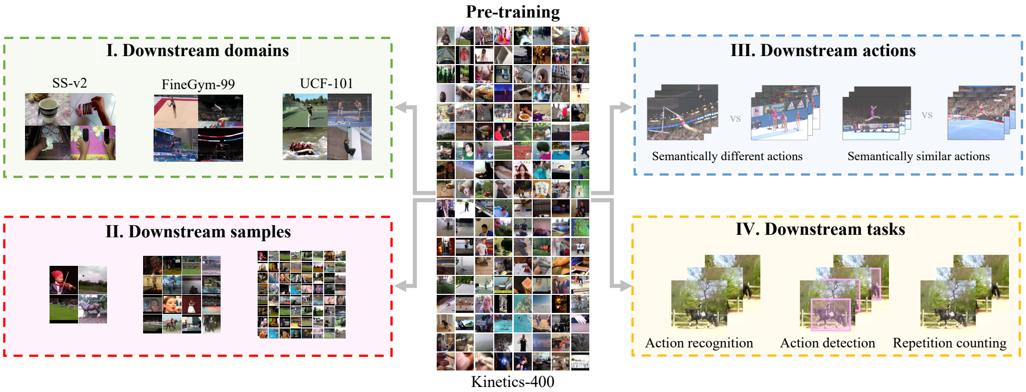


FIGURE 4.1: **Benchmark-sensitivity.** We evaluate the sensitivity of 9 video self-supervised learning methods along 4 downstream factors which vary from the pre-training source: the domain, the samples, the actions and the task.

video self-supervised models are successful, as it cannot show the generalizability or the sensitivity of methods to factors such as domain shift, amount of finetuning data samples, action similarity or task shift. In this study, we identify the sensitivity of existing evaluations and thoroughly benchmark self-supervised video learning methods along four sensitivity factors as depicted in Figure 4.1.

I. **Downstream domain.** First, we analyze whether features learned by self-supervised models transfer to datasets that vary in domain with respect to the pre-training dataset.

II. **Downstream samples.** Second, we evaluate the sensitivity of self-supervised methods to the number of downstream samples available for finetuning.

III. **Downstream actions.** Third, we investigate if self-supervised methods learn fine-grained features required to recognize semantically similar actions.

IV. **Downstream task.** Finally, we study the sensitivity of video self-supervised methods to the downstream task and question whether self-supervised features can be used beyond action recognition.

4.2.1 Downstream Video Datasets

We evaluate various self-supervised models along our four sensitivity factors on 7 video datasets: **UCF-101** [209], **NTU-60** [195], **SomethingSomething-v2** (SS-v2) [74], **FineGym** (Gym-99) [197], **EPIC-Kitchens-100** (EK-100) [39], **Charades** [202] and **AVA** [77]. They include a considerable variety in video domain, the actions they contain and cover a range of video understanding tasks. To get a sense of the differences between these downstream datasets and the Kinetics-400 source dataset, we summarize their similarity to Kinetics-400 by radar plots in Figure 4.2

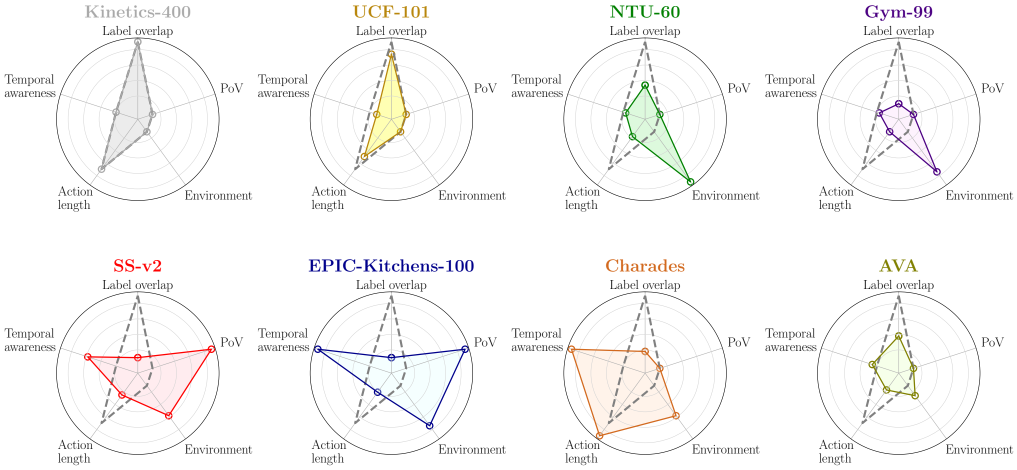


FIGURE 4.2: **Video dataset characteristics.** Characterizing domain shift in datasets via difference in label overlap, point-of-view (PoV), environment, action length and temporal awareness with Kinetics-400 (shown by dotted line). Kinetics-400 and UCF-101 are highly similar to each other, while datasets like Something-Something-v2, EPIC-Kitchens-100 and Charades have different attributes compared to Kinetics-400.

based on several attributes. *Environment* refers to the variety of settings contained in the dataset. *Point-of-view* is whether a video is recorded from a first-person or third-person viewpoint. *Temporal awareness* defines the extent to which temporal context is required to recognize or detect actions. We quantify this as the point at which performance saturates with increasing temporal context in the input. *Label overlap* is the fraction of actions in a target dataset that are also present in Kinetics-400. *Action length* is the temporal length of the actions in seconds. Details are provided in the Appendix B.

4.2.2 Evaluated Self-Supervised Video Learning Methods

Self-supervised learning methods in video can be grouped into two categories based on the objective they use: pretext task methods and contrastive learning methods. Pretext task methods use predictive tasks such as solving spatio-temporal jigsaw puzzles [2, 97, 115], rotation prediction [106], frame and clip order [157, 64, 216, 255, 266], video speed [15, 33, 267, 239], video completion [147], predicting motion statistics [241], tracking random patches in video frames [237] or audio-visual clustering [26, 92, 8, 5]. Contrastive learning methods discriminate between 'positive' and 'negative' pairs to learn invariances to certain data augmentations and instances either from visual-only input [169, 41, 81, 262, 182, 137, 44, 213] or multi-modal data [171, 158, 80, 219, 149, 119, 221].

Some methods also combine pretext and contrastive approaches [217, 173, 281, 11, 44, 94]. A detailed survey of video self-supervised learning methods can be found in [193]. We consider 9 video-based self-supervised methods which achieve good performance on current benchmarks and cover a range of self-supervised paradigms in the video domain, including contrastive learning, pretext-tasks, their combination and cross-modal audio-video learning.

Due to the high computational cost of training self-supervised methods, we focus on works with publicly available weights for a common R(2+1)D-18 network [229] pre-trained on Kinetics-400 [113]: **MoCo** [30], **SeLaVi** [8], **VideoMoCo** [169], **Pretext-Contrast** [217], **RSPNet** [173], **AVID-CMA** [158], **CtP** [237], **TCLR** [41] and **GDT** [171]. We compare these to no pre-training, *i.e.* training from scratch, and fully supervised pre-training for action recognition. It is worth noting that since we use publicly available models we cannot control the exact pre-training setup. There are subtle differences in the training regime for each method, such as the number of epochs, the data augmentations used and the batch size. Details of these differences are provided in the Appendix B. However, all models use the same backbone and pre-training dataset thus we can evaluate their downstream abilities in exactly the same way. To finetune for downstream tasks we simply attach a task-dependent head at the last layer of the pre-trained R(2+1)D-18 backbone to produce label predictions for the corresponding task. For a fair comparison, we use the same set of hyper-parameters, optimization and pre-processing during the downstream training of each model.

4.3 Sensitivity Factor I: Downstream Domain

We first investigate to what extent self-supervised methods learn features that are applicable to action recognition in any domain. We evaluate the suite of pre-trained models on UCF-101, NTU-60, Gym-99, SS-v2 and EK-100 for the task of action recognition. It is worth noting that as well as variety in domain, these datasets include variety in the amount of training data (9.5k - 168k examples) and cardinality of classification (60 - 300 classes). We attach a single classification layer to the pre-trained backbone and evaluate the models' performance on the downstream task in two settings. First, **full finetuning** where we train the whole network from the initialization of the pre-trained weights. Second, **linear evaluation** where we train the classification layer only using the frozen features of pre-trained backbones. We follow the standard splits proposed in the original datasets and report video-level top-1 accuracy on the test sets. The details about splits, pre-processing, training for each dataset are provided in the Appendix B.

Full finetuning. The left part of Table 4.1 shows the results of full finetuning. From the results, it is clear that all self-supervised methods are very effective on UCF-101 as there is a significant gap between training from scratch and using self-supervised pre-training. This gap is reduced as the difference between Kinetics-400 and the downstream domain increases. SeLaVi, MoCo and AVID-CMA in particular are evidence of this as these methods suffer when datasets have higher temporal awareness and less label overlap with Kinetics-400. When moving from UCF-101 to NTU-60 and

Pre-training	**Finetuning**					**Linear Evaluation**					
	UCF101	NTU60	Gym99	SSv2	EK 100	K 400	UCF101	NTU60	Gym99	SSv2	EK 100
None	77.3	92.9	89.8	57.1	25.7	-	-	-	-	-	-
MoCo	83.3	93.4	90.7	57.1	26.4	34.5	65.4	16.0	21.2	7.4	21.4
VideoMoCo	84.9	94.1	90.3	59.0	43.6	31.0	66.3	51.6	41.6	19.5	25.7
SeLaVi	85.2	92.8	88.9	56.2	33.8	24.1	51.2	15.7	20.2	4.5	22.4
Pretext-Contrast	87.7	93.9	90.5	56.9	34.3	22.4	57.2	17.6	30.0	10.9	20.0
RSPNet	88.7	93.9	91.1	59.0	42.7	46.0	76.6	33.5	32.2	12.5	24.9
AVID-CMA	88.8	94.0	90.4	52.0	29.9	43.5	78.1	53.9	45.1	16.1	22.5
CtP	90.1	94.3	92.0	59.6	42.8	7.6	37.9	22.6	30.6	12.2	20.0
TCLR	90.8	94.1	91.6	59.8	36.2	19.9	63.3	33.5	33.0	10.8	21.8
GDT	91.3	93.9	90.5	58.0	37.3	38.6	75.7	38.2	34.2	11.9	25.3
Supervised	93.9	93.9	92.1	60.8	47.7	65.9	91.7	45.5	42.7	16.6	26.6

TABLE 4.1: **Sensitivity Factor I: Downstream Domain.** Video self-supervised methods evaluated across datasets with increasing domain shift with respect to the source dataset (see Figure 4.2). Colors denote relative rankings across methods for each dataset, ranging from low ▬ high. The ranking of methods is domain-sensitive for both finetuning and linear classification and becomes less and less correlated with the current UCF-101 benchmark as the domain shift increases.

Gym-99 there is a change in the ordering of self-supervised methods. This demonstrates a high performance on UCF-101 does not guarantee a self-supervised model is generalizable to other domains. The change in ranking is even more prominent for SS-v2 and EK-100, which require the most temporal awareness and also shift to a first-person viewpoint. This is particularly noticeable for AVID-CMA. On these datasets, MoCo has similar results to no pre-training, which is evidence that video-specific self-supervised learning methods are needed and that image-based methods are insufficient. Overall, supervised pre-training achieves good performance across the board, outperforming self-supervised methods on the most similar domain (UCF-101) as well as the most dissimilar domains (SS-v2 and EK-100). Amidst the models tested, CtP, RSPNet, VideoMoCo and TCLR stand out as the self-supervised pre-training methods most generalizable to different domains.

Linear classification. The right part of Table 4.1 shows the results for linear classification. As with finetuning, the ranking among the self-supervised methods changes as the domain difference between the pre-training and the downstream dataset increases. For example, VideoMoCo ranks lower than GDT and RSPNet for UCF-101 and Kinetics-400 but ranks higher than both for all other datasets. This again demonstrates that performance on UCF-101 does not give a complete picture of a self-supervised model's success. We also observe that linear evaluation on Kinetics-400, as some papers report [182, 189, 262], has the same issue since it is highly correlated to UCF-101 performance. For UCF-101 and Kinetics-400, self-supervised models with contrastive objectives learn highly discriminative features compared to the non-contrastive models. This can be seen by comparing contrastive models AVID-CMA, GDT and RSPNet to non-contrastive SeLaVi and CtP. From the NTU-60 and Gym-99 results we observe that as the label overlap between the pre-training and the downstream dataset decreases, the performance gap between finetuning and linear evaluation increases considerably. This is true for both supervised and self-supervised pre-training. The most generalizable methods in the linear classification setting are contrastive methods VideoMoCo and AVID-CMA as well as supervised pre-training. Interestingly, there are cases where VideoMoCo and AVID-CMA even outperform supervised pre-training, namely for NTU-60, Gym-99 and SS-v2.

> **Conclusion.** We observe from Table 4.1 that performance for both UCF-101 finetuning and Kinetics-400 linear evaluation is not indicative of how well a self-supervised video model generalizes to different downstream domains, with the ranking of methods changing substantially across datasets and whether full finetuning or linear classification is used.

4.4 Sensitivity Factor II: Downstream Samples

The previous section analyzed sensitivity to the downstream domain by evaluating performance on several different datasets. However, finetuning on each of these datasets uses a large number of labeled examples, which means training from scratch already obtains good performance. Not all domains and use cases have ample labeled video examples available, thus we investigate what the impact of the number of finetuning samples is and whether self-supervised methods can be beneficial in scenarios where we have little data to finetune with. We vary the amount of finetuning data, beginning from 1000 videos, sampled uniformly from the classes, and double the amount until we reach the full training set size. We report on four of the downstream datasets from the previous section: UCF-101, NTU-60, Gym-99 and SS-v2. The results are summarized in Figure 4.3.

We first observe that the trends in the low data regime are different from those with the full data. The gap between supervised and self-supervised pre-training is much larger in low data settings, particularly for UCF-101 and Gym-99. NTU is an exception, where, with 1000-4000 samples CtP, GDT, AVID-CMA and TCLR outperform supervised pre-training. As with changes in the downstream domain, change in the amount of downstream examples also causes a change in the ranking of self-supervised models. For example, on UCF-101, RSPNet is much more successful than CtP and TCLR when using only 1000 samples. This is because some self-supervised models benefit more than others from an increased amount of downstream samples. For example, CtP is one of the most generalizable pre-training strategies when finetuning with the full data on UCF-101, Gym-99 and SS-v2, but this is not the case with fewer training samples. Interestingly, GDT is consistently high in the ranking with low amounts of finetuning samples. This is likely due to the large number of temporal augmentations it uses, which help the generalization ability when the training data is limited.

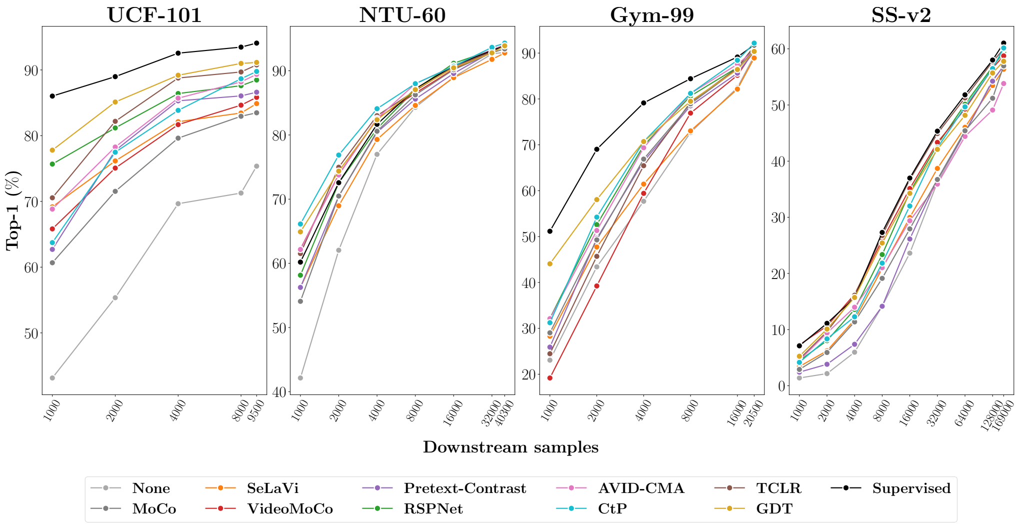


FIGURE 4.3: **Sensitivity Factor II: Downstream Samples.** Comparison of video self-supervised learning methods using varying number of finetuning samples for four downstream datasets. Both the gap and rank among pre-training methods are sensitive to the number of samples available for finetuning.

> **Conclusion.** We observe from Figure 4.3 that video self-supervised models are highly sensitive to the amount of samples available for finetuning, with both the gap and rank between methods changing considerably across sample sizes on each dataset.

4.5 Sensitivity Factor III: Downstream Actions

As indicated earlier, existing evaluations of self-supervised video learning methods have been limited to coarse-grained action recognition. In this section, we investigate whether current self-supervised tasks are only effective for these types of benchmarks or whether they are able to learn features that are useful for differentiating more challenging and semantically similar actions.

FineGym [197] provides us with an experimental setup to study sensitivity to this factor. The dataset contains different evaluations with varying levels of semantic similarity, namely action recognition *across all events*, *within an event* or *within a set*. Recognition *across all events* uses the whole of Gym-99 containing actions from four gymnastic events. For recognition *within an event* there are two subsets: Vault and Floor containing only actions from these two events. Recognition *within a set* has two subsets namely FX-S1, containing different *leaps-jumps-hops* in Floor, and UB-S1, which consists of types of *circles* in Uneven Bars. We also experiment with the long-tailed version of FineGym, Gym-288, which adds 189 more tail classes. Details of these subsets are in the Appendix B. As before, we attach a classification head to the pre-trained models and finetune the whole network with the training set of

	Gym99					Gym288
Pre-training	Across Events	Within Event		Within Set		Across Events
	All	Vault	Floor	FX-S1	UB-S1	All
None	84.8	24.7	75.9	46.6	82.3	50.0
SeLaVi	84.5	25.4	76.0	51.3	80.9	52.8
AVID-CMA	85.7	30.4	82.7	68.0	87.3	52.5
VideoMoCo	85.9	28.4	79.5	57.3	83.9	54.1
Pretext-contrast	86.0	28.5	81.4	66.1	86.1	52.7
MoCo	86.5	33.2	83.3	65.0	84.5	55.1
GDT	86.6	36.9	83.6	66.0	83.4	55.4
RSPNet	86.9	33.4	82.7	65.4	83.6	55.2
TCLR	87.7	29.8	84.3	60.7	84.7	55.4
CtP	88.1	26.8	86.2	79.1	88.8	56.5
Supervised	88.6	37.7	86.1	79.0	87.1	58.4

TABLE 4.2: **Sensitivity Factor III: Downstream Actions.** Video self-supervised models evaluated on different semantic similarities of action in FineGym: across events, within an event and within a set. Colors denote relative rankings across methods for each dataset, ranging from low high. Many methods struggle on the within a set benchmark where actions are most semantically similar.

each subset. In Table 4.2 we report Top-1 accuracy (mean per-class) on the testing sets following [197].

Performance of self-supervised methods varies considerably across downstream actions. The methods that perform best on Gym-99 often do not generalize well to the subsets with higher semantic similarity among actions. This is particularly noticeable for RSPNet and TCLR which drop in the ranking for the within-set subsets. All self-supervised methods, except GDT, struggle on Vault, likely due to the intense motions. Surprisingly, MoCo performs reasonably well when actions are more semantically similar, and is comparable to GDT and RSPNet. The best self-supervised method for subsets with high semantic similarity is CtP. This is especially evident from FX-S1 where it outperforms the second-best self-supervised method, AVID-CMA, by 12%. As with downstream domain and samples, supervised pre-training generalizes better than self-supervised methods across downstream actions with only CtP achieving comparable performance.

Table 4.2 also compares balanced Gym-99 with long-tailed Gym-288. We observe that self-supervised methods are not robust to this change in distribution, with the gap in performance with respect to supervised pre-training increasing. However, the ranking remains consistent, meaning the performance on the balanced set is generally indicative of the performance on the long-tailed set.

> **Conclusion.** Most self-supervised methods in Table 4.2 are sensitive to the actions present in the downstream dataset and do not generalize well to more semantically similar actions. This further emphasizes the need for proper

> evaluation of self-supervised methods beyond current coarse-grained action classification.

4.6 Sensitivity Factor IV: Downstream Tasks

The fourth factor we investigate is whether self-supervised video models are sensitive to the downstream task or whether features learned by self-supervised models are useful to video understanding tasks beyond action recognition. We evaluate this in two ways. First, we keep the domain fixed and evaluate different tasks in a domain similar to the pre-training dataset. We also explore further tasks by changing the domain and seeing how these two factors interplay.

4.6.1 Task-shift within domain.

We consider three different tasks which are all defined for UCF-101: spatio-temporal action detection [118], repetition counting [277] and arrow-of-time prediction [71]. Using UCF-101 allows us to keep the domain fixed across tasks and eliminates the impact of domain shift. Note that each task uses a different subset of the full UCF-101 dataset, however, the domain remains consistent. For each task, we use the R(2+1)D-18 networks as the pre-trained backbones as before and attach task-dependent heads. We report mean Average Precision for spatio-temporal localization [153], mean absolute counting error for repetition counting [277] and classification accuracy for arrow-of-time prediction [71, 248]. Further details are in the Appendix B.

From the results in Table 4.3, we observe that self-supervised learning is beneficial to tasks beyond action recognition, with almost all methods outperforming training from scratch on spatio-temporal action detection, repetition counting and arrow-of-time prediction. Action detection results are well correlated with action recognition. Repetition counting and arrow-of-time have less correlation with action recognition, suggesting that the current benchmark on UCF-101 action recognition by itself is not a good indication of how well self-supervised methods generalize to other tasks. For repetition counting and arrow-of-time prediction, some methods perform comparably to or outperform supervised pre-training. Notably, RSPNet and TCLR generalize the best across these tasks, with GDT also performing well on repetition counting. CtP ranks high on action recognition and detection but performs modestly for repetition counting. This shows that different methods have different task sensitivity, so a thorough evaluation along downstream tasks is needed.

4.6.2 Task-shift out of domain.

We also evaluate how well the self-supervised models generalize when both the domain and the task change. We do so with two popular video understanding benchmarks: long-term multi-label classification on Charades [202] and short-term spatio-temporal

	Task-shift within domain				**Task-shift out of domain**	
Pre-training	Action Recognition	Action Detection	Repetition Counting	Arrow of Time	Multi-label Recognition	Action Detection
None	77.3	0.327	0.217	56.1	7.9	7.4
MoCo	83.3	0.416	0.208	80.3	8.3	11.7
VideoMoCo	84.9	0.440	0.185	72.9	10.5	13.1
SeLaVi	85.2	0.419	0.162	77.4	8.4	10.2
Pretext-contrast	87.7	0.462	0.164	77.2	8.9	12.7
RSPNet	88.7	0.467	0.145	87.0	9.0	14.1
AVID-CMA	88.8	0.435	0.148	83.3	8.2	10.0
CtP	90.1	0.465	0.178	77.1	9.6	10.0
TCLR	90.8	0.476	0.142	85.6	12.2	10.8
GDT	91.3	0.463	0.123	76.4	8.5	12.6
Supervised	93.9	0.482	0.132	77.0	23.5	17.9

TABLE 4.3: **Sensitivity Factor IV: Downstream Tasks.** Transferability of self-supervised video learning methods across video understanding tasks. Colors denote relative rankings across methods for each task, ranging from low high. Note that for repetition counting lower (error) is better. Self-supervised features are transferable to different downstream tasks when the domain shift is low, but struggle when there is also a domain shift. Action recognition on UCF-101 is not a good proxy for self-supervised video learning use cases where a downstream domain- and task-shift can be expected.

action detection on AVA [77]. For both, we follow the setup and training procedure from [62] with R(2+1)D-18 models as the pre-trained backbone and we measure performance in mean Average Precision. Details are in the Appendix B.

From the results in Table 4.3, we observe that supervised pre-training is far more generalizable than all self supervised methods, which all struggle considerably when both the domain and task change. For long-term action classification on Charades, TCLR is slightly better than other methods. On AVA, RSPNet is the best performing self-supervised method with VideoMoCo second. In Section 4.3, we earlier observed that these were two of the methods more robust to domain shift suggesting that this factor is key to success on AVA.

> **Conclusion.** The results in Table 4.3 reveal that action classification performance on UCF-101 is mildly indicative for transferability of self-supervised features to other tasks on UCF-101. However, when methods pre-trained on Kinetics-400 are confronted with a domain change in addition to the task change, UCF-101 results are no longer a good proxy and the gap between supervised and self-supervised pre-training is large.

4.7 SEVERE-benchmark

As evident from the results in previous sections, current video self-supervised methods are benchmark-sensitive to the four factors we have studied. Based on our findings, we propose the SEVERE-benchmark (SEnsitivity of VidEo REpresentations) for use in future works to more thoroughly evaluate new video self-supervised methods for generalization along the four sensitivity factors we have examined. Since we do not expect future works to run all the experiments from our study, we create a subset of experiments that are indicative benchmarks for each sensitivity factor and realistic to run. We summarize the benchmark composition in Table 4.4 and detail its motivation per factor. Standard deviations for the results we obtain on this benchmark can be found in the Appendix B.

Downstream domain. To measure a self-supervised model's domain sensitivity we recommend using Something-Something-v2 and FineGym-99. These two datasets come from domains distinct to Kinetics-400 and UCF-101 and also each other. FineGym-99 evaluates a model's ability to generalize to datasets with less distinctive backgrounds where there are few actions in common with Kinetics-400. SS-v2 evaluates the generalizability to actions that require high temporal awareness as well as the shift to a first-person viewpoint. It is evident from Table 4.4 that there are significant rank changes between UCF-101, Gym-99 and SS-v2 thus these three datasets provide a challenging subset for future methods.

Downstream samples. For the sample sensitivity, we recommend using 1000 samples on UCF-101 and Gym-99. Using 1000 samples showed the most dramatic difference from the full dataset size particularly for these datasets where there is a considerable gap between self-supervised and supervised pre-training as well as considerable rank change among the methods.

Downstream actions. To test generalizability to recognizing semantically similar actions, we recommend evaluating the two within-set granularities of Gym-99 *i.e.* FX-S1 and UB-S1. Both of these subsets have high semantic similarity between actions with methods currently struggling to generalize to both of these subsets as can be seen in Table 4.4. There is also a significant gap between supervised and most self-supervised pre-training methods for FX-S1, highlighting the potential for future works in this area.

Downstream task. To evaluate the task sensitivity, we recommend using repetition counting on UCF-101 and multi-label classification on Charades. Repetition counting on UCF-101 highlights different strengths to action recognition as it allows investigation of a model's ability to generalize to a task that requires more temporal understanding without measuring the impact of the domain. We recommend multi-label classification on Charades as it is currently a very challenging task for self-supervised models and allows the combination of domain and task shift to be investigated. Code to compare on the SEVERE-benchmark is available at https://github.com/fmthoker/SEVERE-BENCHMARK.

4.8 Observations, Limitations and Recommendations

Observations. We hope that our study and resulting benchmark provides a helpful insight for future research to design novel self-supervised methods for generalizable

Pre-training	**Existing**	**SEVERE-benchmark**							
		Domains		Samples		Actions		Tasks	
	UCF101	SS-v2	Gym-99	UCF (10^3)	Gym-99 (10^3)	FX-S1	UB-S1	UCF-RC	Charades-MLC
None	77.3	57.1	89.8	38.3	22.7	46.6	82.3	0.217	7.9
MoCo	83.3	57.1	90.7	60.4	30.9	65.0	84.5	0.208	8.3
VideoMoCo	84.9	59.0	90.3	65.4	20.6	57.3	83.9	0.185	10.5
SeLaVi	85.2	56.2	88.9	69.0	30.2	51.3	80.9	0.162	8.4
Pretext-Contrast	87.7	56.9	90.5	64.6	27.5	66.1	86.1	0.164	8.9
RSPNet	88.7	59.0	91.1	74.7	32.2	65.4	83.6	0.145	9.0
AVID-CMA	88.8	52.0	90.4	68.2	33.4	68.0	87.3	0.148	8.2
CtP	90.1	59.6	92.0	61.0	32.9	79.1	88.8	0.178	9.6
TCLR	90.8	59.8	91.6	72.6	26.3	60.7	84.7	0.142	12.2
GDT	91.3	58.0	90.5	78.4	45.6	66.0	83.4	0.123	8.5
Supervised	93.9	60.8	92.1	86.6	51.3	79.0	87.1	0.132	23.5

TABLE 4.4: **Proposed SEVERE-benchmark** for evaluating video self-supervised methods for generalization along downstream domains, samples, actions and tasks.

video representation learning. From the benchmark results in Table 4.4, we observe that:

(i) There is no clear winner as different methods stand out in different downstream settings.

(ii) Supervised pre-training is dominant across all sensitivity factors, especially when the number of available downstream samples are limited and when there is a change in both the downstream domain and the downstream task.

(iii) Self-supervised contrastive methods that explicitly encourage features to be distinct across the temporal dimension transfer well. This is visible from the consistent performance of GDT, TCLR and RSPNet across different sensitivity factors.

(iv) Learning certain temporal invariances may prevent generalizability to temporal or fine-grained benchmarks. This is evident from GDT's performance on SS-v2 and UB-S1. These benchmarks require distinction between actions such as *moving something left* vs. *moving something right* in SS-v2 and *giant circle forwards* vs. *giant circle backwards* in UB-S1. The invariance to temporal reversal learned by GDT impacts its ability to recognize such actions. Similarly, MoCo outperforming VideoMoCo on the FX-S1 and UB-S1 Gym-99 subsets suggests that invariance to frame dropout in VideMoCo can harm the performance on highly similar actions.

(v) Pretext-tasks specific to videos can be effective to learn more fine-grained features. CtP generalizes well both to different domains where the background is less indicative of the action and to more semantically similar actions. The pretext task is to track and estimate the position and size of image patches moving in a sequence of video frames. Such a formulation requires the network to learn to follow moving targets and ignore the static background information. CtP's generalization success demonstrates that contrastive learning is not the only way forward for self-supervised video representation learning.

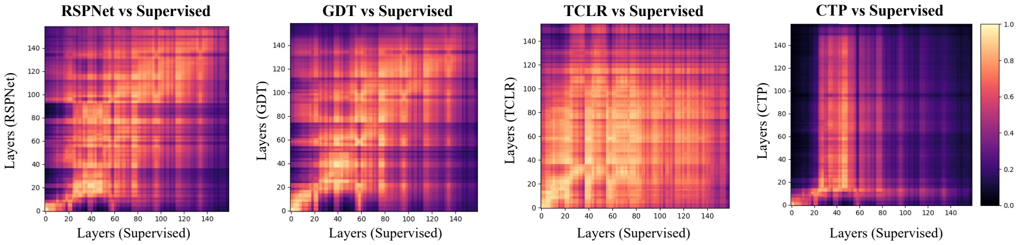


FIGURE 4.4: **Representation similarity** between features of top self-supervised methods and supervised pre-training on Kinetics-400 validation set (using centered kernel alignment [162]). Contrastive methods have a high correlation with supervised pretraining, while CtP's features are far away. Thus, showing potential for both imitating supervised learning as well as learning features distinct to it.

(vi) Figure 4.4 shows the feature similarity on Kinetics using centered kernel alignment [162] between supervised pre-training and the best self-supervised methods *i.e*. GDT, RSPNet, TCLR, CtP. This figure illustrates that contrastive methods seem to imitate supervised pre-training as the correlation between supervised pre-training and the three contrastive methods (RSPNet, GDT and TCLR) is high. This explains the good performance of these methods on UCF-101 with 1000 examples. By contrast, CtP's features are far away from supervised pre-training. This is interesting because CtP generalizes well to new domains and actions, it shows that good generalization capability can be obtained without imitating supervised pre-training.

Limitations. While our study has highlighted the benchmark sensitivity of video self-supervised learning across four factors, there are many more factors that we do not consider in this chapter. Due to computational limits, we keep the source dataset fixed as Kinetics-400 and use publicly available pre-trained models. This means there is variability in the exact pre-training setup such as the spatial data augmentations that are used by each model. We hope that future works will explore impact of such pretraining factors as well as the impact of pre-training on other large-scale datasets such as Ego4D [75] for the generalization of video self-supervised models. Another limitation of our study is that we only consider a fixed R(2+1)D-18 backbone, which is currently one of the most commonly used in video self-supervised learning. This allows our comparison between methods to be fair, however, it does limit the ability of methods to perform well on datasets such as EPIC-Kitchens-100. Another factor that could be explored further is the task. We have considered a selection of various video understanding tasks centered around human actions. However, there are many more video understanding tasks that could be explored such as human centric tasks like action anticipation [39] and temporal action detection[39], as well as non-human centric tasks like animal behavior analysis [55, 161, 214], multi-object tracking [172] and visual grounding [214].

Recommendations. Based on the results and our observations, we have several recommendations for future works in video self-supervised learning. (i) Our study has highlighted the need for more focus on generalizability of self-supervised learning methods, particularly along the domain and dataset size factors. (ii) Distinguishing across the temporal dimension is effective and is a useful direction to pursue further for generalizability. (iii) Pretext-tasks like the one used in CtP are good for the generalizability to domain and action, thus designing new video specific pretext tasks is a promising direction. This could also be combined with contrastive learning tasks to gain the benefits of both types of learning.

Chapter 5

Tubelet-Contrastive Self-Supervision for Video-Efficient Generalization

5.1 Introduction

This chapter aims to learn self-supervised video representations, useful for distinguishing actions. In a community effort to reduce the manual, expensive, and hard-to-scale annotations needed for many downstream deployment settings, the topic has witnessed tremendous progress in recent years [64, 255, 102, 193], particularly through contrastive learning [181, 169, 173, 60]. Contrastive approaches learn representations through instance discrimination [167], by increasing feature similarity between spatially and temporally augmented clips from the same video. Despite temporal differences, such positive video pairs often maintain high spatial similarity (see Figure 5.1), allowing the contrastive task to be solved by coarse-grained features without explicitly capturing local motion dynamics. This limits the generalizability of the learned video representations, as shown in our prior work [224]. Furthermore, prior approaches are constrained by the amount and types of motions present in the pretraining data. This makes them data-hungry, as video data has high redundancy with periods of little to no motion. In this chapter, we address the need for data-efficient and generalizable self-supervised video representations by proposing a contrastive method to learn local motion dynamics.

We take inspiration from action detection, where tubelets are used to represent the motions of people and objects in videos through bounding box sequences *e.g.*,[100, 107, 134]. Typically, many tubelet proposals are generated for a video, which are processed to find the best prediction. Rather than finding tubelets in video data, we simulate them. In particular, we sample an image patch and 'paste' it with a randomized motion onto two different video clips as a shared tubelet (see Figure 5.1). These two clips form a positive pair for contrastive learning where the model has to rely on the spatiotemporal dynamics of the tubelet to learn the similarity. With such a formulation, we can simulate a large variety of motion patterns that are not present in the original videos. This allows our model to be data-efficient while improving generalization to new domains and fine-grained actions.

We make four contributions. First, we explicitly learn from local motion dynamics in the form of synthetic tubelets and design a simple but effective tubelet-contrastive

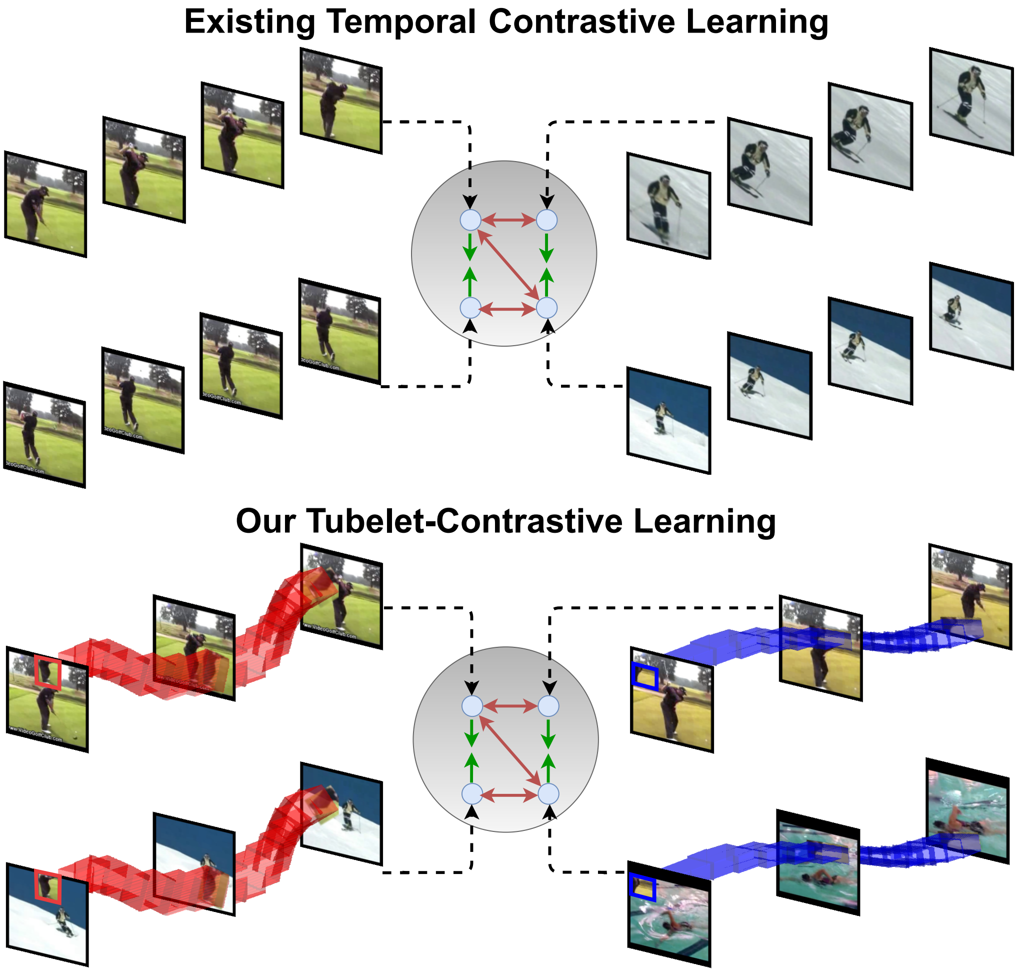

Existing Temporal Contrastive Learning
Our Tubelet-Contrastive Learning

FIGURE 5.1: **Tubelet-Contrastive Positive Pairs** (bottom) only share the spatiotemporal motion dynamics inside the simulated tubelets, while temporal contrastive pairs (top) suffer from a high spatial bias. Contrasting tubelets results in a data-efficient and generalizable video representation.

framework. Second, we propose different ways of simulating tubelet motion and transformations to generate a variety of motion patterns for learning. Third, we reveal the remarkable data efficiency of our proposal: on five action recognition datasets our approach maintains performance when using only 25% of the pretraining videos. What is more, with only 5-10% of the videos we still outperform the vanilla contrastive baseline with 100% pretraining data for several datasets. Fourth, our comparative experiments on 10 downstream settings, including UCF101 [209], HMDB51 [122], Something Something [74], and FineGym [197], further demonstrate our competitive performance, generalizability to new domains, and suitability of our learned representation for fine-grained actions.

5.2 Related Work

Self-Supervised Video Representation Learning. The success of contrastive learning in images [85, 29, 76, 159] inspired many video contrastive works [181, 169, 173, 94, 139, 218]. Alongside spatial invariances, these works learn invariances to temporal crops [169, 181, 188] and video speed [173, 94, 139]. Some diverge from temporal invariances and encourage equivariance [171, 41] to learn finer temporal representations. For instance, TCLR [41] enforces within-instance temporal feature variation, while TE [101] learns equivariance to temporal crops and speed with contrastive learning. Alternatively, many works learn to predict temporal transformations such as clip order [64, 255, 125], speed [15, 33, 268] and their combinations [147, 102]. These self-supervised temporal representations are effective for classifying and retrieving coarse-grained actions but are challenged by downstream settings with subtle motions [224, 193]. Other works utilize the multimodal nature of videos [4, 8, 171, 158, 80, 69, 150] and learn similarity with audio [8, 4, 158] and optical flow [80, 69, 164, 252]. We contrast motions of synthetic tubelets to learn a video representation from only RGB data that can generalize to tasks requiring fine-grained motion understanding.

Other self-supervised works learn from the spatiotemporal dynamics of video. Both BE [244] and FAME [45] remove background bias by adding static frames [244] or replacing the background [45] in positive pairs. Several works instead use masked autoencoding to learn video representations [227, 61, 215]. However, these works are all limited to the motions present in the pretraining dataset. We prefer to be less dataset-dependent and generate synthetic motion tubelets for contrastive learning, which also offers a considerable data-efficiency benefit. CtP [237] and MoSI [96] both aim to predict motions in pretraining. CtP [237] learns to track image patches in video clips while MoSI [96] learns to predict the speed and direction of added pseudo-motions. We take inspiration from these works and contrast synthetic motions from tubelets which allows us to learn generalizable and data-efficient representations.

Supervised Fine-Grained Motion Learning. While self-supervised works have mainly focused on learning representations to distinguish coarse-grained actions, much progress has been made in supervised learning of motions. Approaches distinguish actions by motion-focused neural network blocks [123, 150, 135, 116], decoupling motion from appearance [131, 212], aggregating multiple temporal scales [261, 163, 63], and sparse coding to obtain a mid-level motion representation [152, 178, 198]. Other works exploit skeleton data [52, 90] or optical flow [203, 58]. Alternatively, several works identify motion differences within an action class, by repetition counting [93, 276, 282], recognizing adverbs [50, 49] or querying for action attributes [275]. Different from all these works, we learn a motion-sensitive video representation with self-supervision. We do so by relying on just coarse-grained video data in pretraining and demonstrate downstream generalization to fine-grained actions.

Tubelets. Jain *et al.* defined tubelets as class-agnostic sequences of bounding boxes over time [100]. Tubelets can represent the movement of people and objects and are commonly used for object detection in videos[108, 109, 59], spatiotemporal action

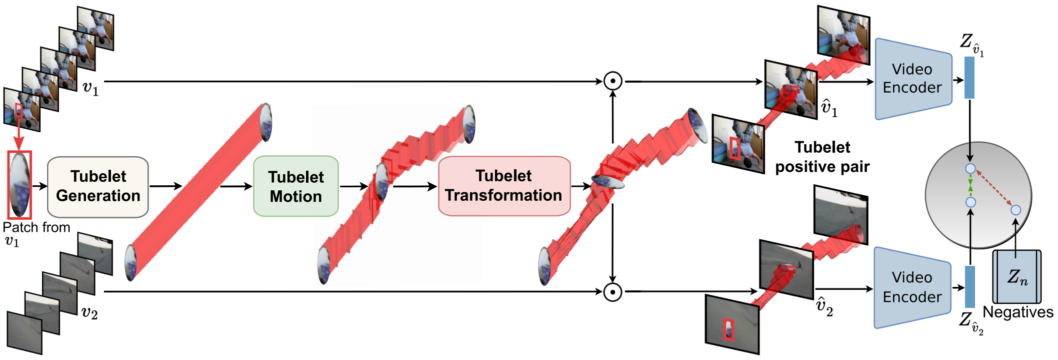


FIGURE 5.2: **Tubelet-Contrastive Learning.** We sample two clips (v_1, v_2) from different videos and randomly crop an image patch from v_1. We generate a tubelet by replicating the patch in time and add motion through a sequence of target locations for the patch. We then add complexity to these motions by applying transformations, such as rotation, to the tubelet. The tubelet is overlaid $\odot$ onto both clips to form a positive tubelet pair $(\hat{v}_1, \hat{v}_2)$. We learn similarities between clips with the same tubelets (positive pairs) and dissimilarities between clips with different tubelets (negatives) using a contrastive loss.

localization [107, 265, 134, 100, 91, 284] and video relation detection [28]. Initially, tubelets were obtained by supervoxel groupings and dense trajectories [100, 70] and later from 2D CNNs [107, 134], 3D CNNs [91, 265] and transformers [284]. We introduce (synthetic) tubelets of pseudo-objects for contrastive video self-supervised learning.

5.3 Tubelet Contrast

We aim to learn motion-focused video representations from RGB video data with self-supervision. After revisiting temporal contrastive learning, we propose tubelet-contrastive learning to reduce the spatial focus of video representations and instead learn similarities between spatiotemporal tubelet dynamics (Section 5.3.2). We encourage our representation to be motion-focused by simulating a variety of tubelet motions (Section 5.3.3). To further improve data efficiency and generalizability, we add complexity and variety to the motions through tubelet transformations (Section 5.3.4). Figure 5.2 shows an overview of our approach.

5.3.1 Temporal Contrastive Learning.

Temporal contrastive learning learns feature representations via instance discrimination [167]. This is achieved by maximizing the similarity between augmented clips from the same video (positive pairs) and minimizing the similarity between clips from different videos (negatives). Concretely given a set of videos V, the positive pairs (v, v') are obtained by sampling different temporal crops of the same video [169, 173] and applying spatial augmentations such as cropping and color jittering. Clips sampled from other videos in the training set act as negatives. The extracted clips are passed through a video encoder and projected on a representation space by a non-linear projection head to obtain clip embeddings $(Z_v, Z_{v'})$. The noise contrastive estimation loss InfoNCE [167] is used for the optimization:

$$\mathcal{L}_{contrast}(v, v') = -\log \frac{h(Z_v, Z_{v'})}{h(Z_v, Z_{v'}) + \sum\limits_{Z_n \sim \mathcal{N}} h(Z_v, Z_n)} \tag{5.1}$$

where $h(Z_v, Z_{v'})$=$\exp(Z_v \cdot Z_{v'}/\tau)$, τ is the temperature parameter and $\mathcal{N}$ is a set of negative clip embeddings.

5.3.2 Tubelet-Contrastive Learning

Different from existing video contrastive self-supervised methods, we explicitly aim to learn motion-focused video representations while relying only on RGB data. To achieve this we propose to learn similarities between simulated tubelets. Concretely, we first generate tubelets in the form of moving patches which are then overlaid onto two different video clips to generate positive pairs that have a high motion similarity and a low spatial similarity. Such positive pairs are then employed to learn video representations via instance discrimination, allowing us to learn more generalizable and motion-sensitive video representations.

Tubelet Generation. We define a tubelet as a sequence of object locations in each frame of a video clip. Let's assume an object p of size $H' \times W'$ moving in a video clip v of length T. Then the tubelet is defined as follows:

$$\text{Tubelet}_p = [(x^1, y^1), .., (x^T, y^T)], \tag{5.2}$$

where (x^i, y^i) is the center coordinate of the object p in frame i of clip v. For this work, a random image patch of size $H' \times W'$ acts as a pseudo-object overlaid on a video clip to form a tubelet. To generate the tubelet we first make the object appear static, *i.e.*, x^1=x^2=...=x^T and y^1=y^2 = ...=y^T, and explain how we add motion in Section 5.3.3.

Tubelet-Contrastive Pairs. To create contrastive tubelet pairs, we first randomly sample clips v_1 and v_2 of size $H \times W$ and length T from two different videos in V. From v_1 we randomly crop an image patch p of size $H' \times W'$. such that $H' \ll H$ and $W' \ll W$. From the patch p, we construct a tubelet Tubelet_p as in Equation 5.2. Then, we overlay the generated tubelet Tubelet_p onto both v_1 and v_2 to create two modified video clips $\hat{v}_1$ and $\hat{v}_2$:

$$\hat{v}_1 = v_1 \odot \text{Tubelet}_p \qquad\qquad \hat{v}_2 = v_2 \odot \text{Tubelet}_p, \tag{5.3}$$

where $\odot$ refers to pasting patch p in each video frame at locations determined by Tubelet_p. Equation 5.3 can be extended for a set of M tubelets $\{\text{Tubelet}_{p_1}, ..., \text{Tubelet}_{p_M}\}$

from M patches randomly cropped from v_1 as:

$$\begin{aligned}\hat{v}_1 &= v_1 \odot \{\text{Tubelet}_{p_1}, ..., \text{Tubelet}_{p_M}\}\\ \hat{v}_2 &= v_2 \odot \{\text{Tubelet}_{p_1}, ..., \text{Tubelet}_{p_M}\}.\end{aligned} \tag{5.4}$$

As a result, $\hat{v}_1$ and $\hat{v}_2$ share the spatiotemporal dynamics of the moving patches in the form of tubelets and have low spatial bias since the two clips come from different videos. Finally, we adapt the contrastive loss from Equation 5.1 and apply $\mathcal{L}_{contrast}(\hat{v}_1, \hat{v}_2)$. Here the set of negatives $\mathcal{N}$ contains videos with different tubelets. Since the only similarity in positive pairs is the tubelets, the network must rely on temporal cues causing a motion-focused video representation.

5.3.3 Tubelet Motion

To learn motion-focused video representations, we need to give our tubelets motion variety. Here, we discuss how to simulate motions by generating different patch movements in the tubelets. Recall, Equation 5.2 defines a tubelet by image patch p and its center coordinate in each video frame. We consider two types of tubelet motion: linear and non-linear.

Linear Motion. We randomly sample the center locations for the patch in K keyframes: the first frame (i=1), the last frame (i=T), and K−2 randomly selected frames. These patch locations are sampled from uniform distributions $x \in [0, W]$ and $y \in [0, H]$, where W and H are the video width and height. Patch locations for the remaining frames $i \notin K$ are then linearly interpolated between keyframes so we obtain the following linear motion definition:

$$\text{Tubelet}^{\text{Lin}} = [(x^1, y^1), (x^2, y^2), ..., (x^T, y^T)], \text{ s.t.} \tag{5.5}$$

$$(x^i, y^i) = \begin{cases} (\mathcal{U}(0, W), \mathcal{U}(0, H)), & \text{if } i \in K \\ \text{Interp}((x^k, y^k), (x^{k+1}, y^{k+1})), & \text{otherwise} \end{cases}$$

where $\mathcal{U}$ is a function for uniform sampling, k and k+1 are the neighboring keyframes to frame i and Interp gives a linear interpolation between keyframes. To ensure smoothness, we constrain the difference between the center locations in neighboring keyframes to be less than Δ pixels. This formulation results in tubelet motions where patches follow linear paths across the video frames. The left of Figure 5.3 shows examples of such linear tubelet motions.

Non-Linear Motion. Linear motions are simple and limit the variety of motion patterns that can be generated. Next, we simulate motions where patches move along more complex non-linear paths, to better emulate motions in real videos. We create non-linear motions by first sampling N 2D coordinates ($N \gg T$) uniformly from $x \in [0, W]$ and $y \in [0, H]$. Then, we apply a 1D Gaussian filter along x and y axes to generate a random smooth nonlinear path as:

$$\begin{aligned}\text{Tubelet}^{\text{NonLin}} =& [(g(x^1), g(y^1)), ..., (g(x^N), g(y^N))]\\ \text{s.t.} \quad g(z) =& \frac{1}{\sqrt{2\pi\sigma}} e^{-z^2/2\sigma^2}\end{aligned} \tag{5.6}$$

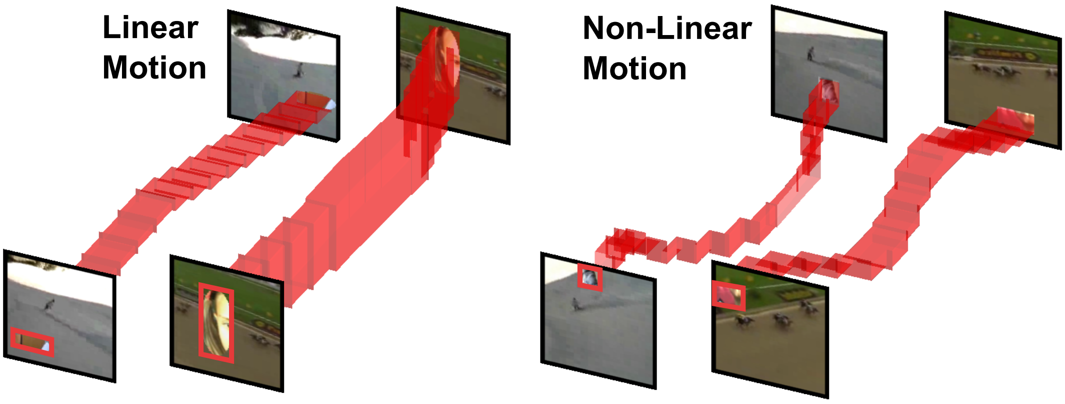


FIGURE 5.3: **Tubelet Motion.** Examples for *Linear* (left) and *Non-Linear* (right). Non-linear motions enable the simulation of a larger variety of motion patterns to learn from.

where σ is the smoothing factor for the gaussian kernels. Note the importance of sampling $N \gg T$ points to ensure a non-linear path. If N is too small then the path becomes linear after gaussian smoothing. We downsample the resulting non-linear tubelet in Equation 5.6 from N to T coordinates resulting in the locations for patch p in the T frames. The right of Figure 5.3 shows examples of non-linear tubelet motions.

5.3.4 Tubelet Transformation

The tubelet motions are simulated by changing the position of the patch across the frames in a video clip, *i.e.* with translation. In reality, the motion of objects in space may appear as other transformations in videos, for instance, scale decreasing as the object moves away from the camera or motions due to planer rotations. Motivated by this, we propose to add more complexity and variety to the simulated motions by transforming the tubelets. In particular, we propose scale, rotation, and shear transformations. As before, we sample keyframes K with the first (i=0) and last frames (i=T) always included. Transformations for remaining frames are linearly interpolated. Formally, we define a tubelet transformation as a sequence of spatial transformations applied to the patch p in each frame i as:

$$\text{Trans}_F = [p, F(p,\theta^2), ..., .., F(p,\theta^T)], \text{ s.t.}$$
$$\theta^i = \begin{cases} \mathcal{U}(\text{Min}, \text{Max}), & \text{if } i \in K \\ \text{Interp}(\theta^k, \theta^{k+1}), & \text{otherwise} \end{cases} \tag{5.7}$$

where $F(p,\theta^i)$ applies the transformation to patch p according to parameters θ^i, $\mathcal{U}$ samples from a uniform distribution and θ^k and θ^{k+1} are the parameters for the keyframes neighboring frame i. For the first keyframe, no transformation is applied thus representing the initial state of the patch p. We instantiate three types of such tubelet transformations: scale, rotation, and shear. Examples are shown in Figure 5.4.

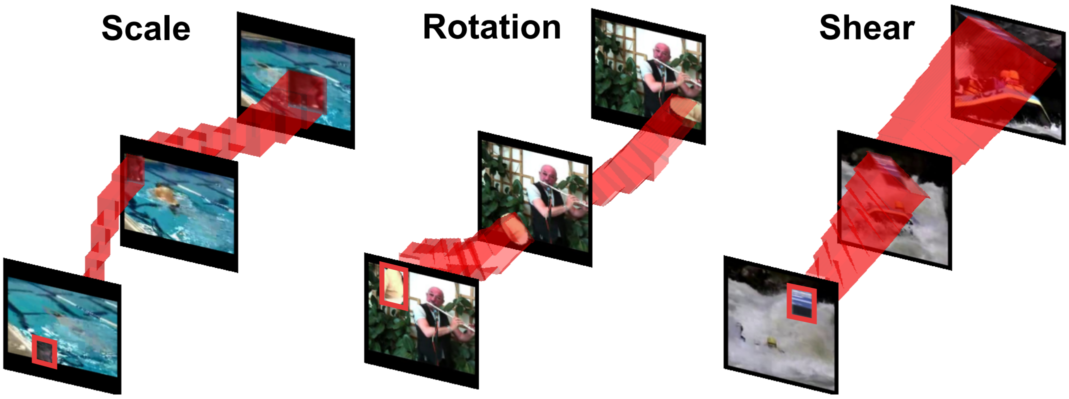


FIGURE 5.4: **Tubelet Transformation.** Examples for *Scale* (left), *Rotation* (middle), and *Shear* (right). The patch is transformed as it moves through the tubelet.

Scale. We scale the patch across time with $F(p, \theta^i)$ and horizontal and vertical scaling factors $\theta^i{=}(w^i, h^i)$. To sample w^i and h^i, we use Min=0.5 and Max=1.5.
Rotation. In this transformation $F(p, \theta^i)$ applies in-plane rotations to tubelet patches. Thus, θ^i is a rotation angle sampled from Min$=-90°$ and Max$=+90°$.
Shear. We shear the patch as the tubelet progresses with $F(p, \theta^i)$. The shearing parameters are $\theta^i{=}(r^i, s^i)$ which are sampled using Min$=-1.5$ and Max=1.5.

With these tubelet transformations and the motions created in Section 5.3.3 we are able to simulate a variety of subtle motions in videos, making the model data-efficient. By learning the similarity between the same tubelet overlaid onto different videos, our model pays less attention to spatial features, instead learning to represent these subtle motions. This makes the learned representation generalizable to different domains and action granularities.

5.4 Experiments

5.4.1 Datasets, Evaluation & Implementation

Pretraining Datasets. Following prior work [169, 171, 41, 237, 173, 94] we use **Kinetics-400** [113] for self-supervised pretraining. Kinetics-400 is a large-scale action recognition dataset containing 250K videos of 400 action classes. To show data efficiency, we also pretrain with **Mini-Kinetics** [254], a subset containing 85K videos of 200 action classes.
Downstream Evaluation. To evaluate the video representations learned by our tubelet contrast, we finetune and evaluate our model on various downstream datasets summarized in Table 5.1. Following previous self-supervised work, we evaluate on standard benchmarks: **UCF101** [209] and **HMDB51** [122]. These action recognition datasets contain coarse-grained actions with domains similar to Kinetics-400. For both, we report top-1 accuracy on split 1 from the original papers. We examine the generalizability of our model with the **SEVERE** benchmark [224] proposed in

Evaluation Factor	Experiment	Dataset	Task	#Classes	#Finetuning	#Testing	Eval Metric
Standard	UCF101	UCF 101 [209]	Action Recognition	101	9,537	3,783	Top-1 Accuracy
	HMDB51	HMDB 51 [122]	Action Recognition	51	3,570	1,530	Top-1 Accuracy
Domain Shift	SSv2	Something-Something [74]	Action Recognition	174	168,913	24,777	Top-1 Accuracy
	Gym99	FineGym [197]	Action Recognition	99	20,484	8,521	Top-1 Accuracy
Sample Efficiency	UCF (10^3)	UCF 101 [122]	Action Recognition	101	1,000	3,783	Top-1 Accuracy
	Gym (10^3)	FineGym [197]	Action Recognition	99	1,000	8,521	Top-1 Accuracy
Action Granularity	FX-S1	FineGym [197]	Action Recognition	11	1,882	777	Mean Class Acc
	UB-S1	FineGym [197]	Action Recognition	15	3,511	1,471	Mean Class Acc
Task Shift	UCF-RC	UCFRep [276]	Repetition Counting	-	421	105	Mean Error
	Charades	Charades [202]	Multi-label Recognition	157	7,985	1,863	mAP

TABLE 5.1: **Benchmark Details** for the downstream evaluation factors, experiments, and datasets we cover. For non-standard evaluations, we follow the SEVERE benchmark [224]. For self-supervised pretraining, we use Kinetics-400 or Mini-Kinetics.

chapter 4. This consists of eight experiments over four downstream generalization factors: *domain shift*, *sample efficiency*, *action granularity*, and *task shift*. *Domain shift* is evaluated on Something-Something v2 [74] (SSv2) and FineGym [197] (Gym99) which vary in domain relative to Kinetics-400. *Sample efficiency* evaluates low-shot action recognition on UCF101 [209] and FineGym [197] with 1,000 training samples, referred to as UCF (10^3) and Gym (10^3). *Action granularity* evaluates semantically similar actions using FX-S1 and UB-S1 subsets from FineGym [197]. In both subsets, action classes belong to the same element of a gymnastic routine, *e.g.*, FX-S1 is types of jump. *Task shift* evaluates tasks beyond single-label action recognition. Specifically, it uses temporal repetition counting on UCFRep [276], a subset of UCF-101 [276], and multi-label action recognition on Charades [202]. The experimental setups are detailed in Table 5.1 and all follow SEVERE [224].

Tubelet Generation and Transformation. Our clips are 16 112×112 frames with standard spatial augmentations: random crops, horizontal flip, and color jitter. We randomly crop 2 patches to generate $M{=}2$ tubelets (Equation 5.4). The patch size $H^{'} \times W^{'}$ is uniformly sampled from $[16\times16, 64\times64]$. We also randomly sample a patch shape from a set of predefined shapes. For linear motions, we use $\Delta{=}[40{-}80]$ displacement difference. For non-linear motion, we use $N{=}48$ and a smoothing factor of $\sigma{=}8$ (Equation 5.6). For linear motion and all tubelet transformations, we use $K{=}3$ keyframes.

Networks, Pretraining and Finetuning. We use R(2+1)D-18 [229] as the video encoder, following previous self-supervision works [240, 171, 173, 45, 41, 169]. The projection head is a 2-layer MLP with 128D output. We use momentum contrast [85] to increase the number of negatives $|\mathcal{N}|$ (Equation 5.1) to 16,384 for Mini-Kinetics and 65,536 for Kinetics. We use temperature $\tau{=}0.2$ (Equation 5.1). The model is optimized using SGD with momentum 0.9, learning rate 0.01, and weight decay 0.0001. We use a batch size of 32 for Mini-Kinetics and 128 for Kinetics, a cosine scheduler [146], and pretrain for 100 epochs. After pretraining, we replace the projection head with a task-dependent head following SEVERE [224]. The whole network is finetuned for the downstream task with labels. We provide finetuning and evaluation details in Appendix C. Code is available at https://github.com/fmthoker/tubelet-contrast.

	UCF (10^3)	Gym (10^3)	SSv2-Sub	UB-S1
Temporal Contrast				
Baseline	57.5	29.5	44.2	84.8
Tubelet Contrast				
Tubelet Generation	48.2	28.2	40.1	84.1
Tubelet Motion	63.0	45.6	47.5	90.3
Tubelet Transformation	65.5	48.0	47.9	90.9

TABLE 5.2: **Tubelet-Contrastive Learning** considerably outperforms temporal contrast on multiple downstream settings. Tubelet motion and transformations are key.

5.4.2 Ablation Studies & Analysis

To ablate the effectiveness of individual components we pretrain on Mini-Kinetics and evaluate on UCF (10^3), Gym (10^3), Something-Something v2 and UB-S1. To decrease the finetuning time we use a subset of Something Something (SSv2-Sub) with 25% of the training data (details in Appendix C). Unless specified otherwise, we use non-linear motion and rotation to generate tubelets.

Tubelet-Contrastive Learning. Table 5.2 shows the benefits brought by our tubelet-contrastive learning. We first observe that our full tubelet-contrastive model improves considerably over the temporal contrastive baseline, which uses MoCo [85] with a temporal crop augmentation. This improvement applies to all downstream datasets but is especially observable with Gym (10^3) (+18.5%) and UB-S1 (+6.1%) where temporal cues are crucial. Our model is also effective on UCF (10^3) (+8.0%) where spatial cues are often as important as temporal ones. These results demonstrate that learning similarities between synthetic tubelets produces generalizable, but motion-focused, video representations required for finer temporal understanding.

It is clear that the motion within tubelets is critical to our model's success as contrasting static tubelets obtained from our tubelet generation (Section 5.3.2) actually decreases the performance from the temporal contrast baseline. When tubelet motion is added (Section 5.3.3), performance improves considerably, *e.g.*, Gym (10^3) +17.4% and SSv2-Sub +7.4%. Finally, adding more motion types via tubelet transformations (Section 5.3.4) further improves the video representation quality, *e.g.*, UCF (10^3) +2.5% and Gym (10^3) +2.4%. This highlights the importance of including a variety of motions beyond what is present in the pretraining data to learn generalizable video representations.

Tubelet Motions. Next, we ablate the impact of the tubelet motion type (Section 5.3.3) without transformations. We compare the performance of static tubelets with no motion, linear motion, and non-linear motion in Table 5.3. Tubelets with simple linear motion already improve performance for all four datasets, *e.g.*, +6.4% on Gym (10^3). Using non-linear motion further improves results, for instance with an additional +11.0% improvement on Gym (10^3). We conclude that learning from non-linear motions provides more generalizable video representations.

Tubelet Motion	UCF (10^3)	Gym (10^3)	SSv2-Sub	UB-S1
No motion	48.2	28.2	40.1	84.1
Linear	55.5	34.6	45.3	88.5
Non-Linear	63.0	45.6	47.5	90.3

TABLE 5.3: **Tubelet Motions.** Learning from tubelets with non-linear motion benefits multiple downstream settings.

Transformation	UCF (10^3)	Gym (10^3)	SSv2-Sub	UB-S1
None	63.0	45.6	47.5	90.5
Scale	65.1	46.5	47.0	90.5
Shear	65.2	47.5	47.3	90.9
Rotation	65.5	48.0	47.9	90.9

TABLE 5.4: **Tubelet Transformation.** Adding motion patterns to tubelet-contrastive learning through transformations improves downstream performance. Best results for rotation.

Tubelet Transformation. Table 5.4 compares the proposed tubelet transformations (Section 5.3.4). All four datasets benefit from transformations, with rotation being the most effective. The differences in improvement for each transformation are likely due to the types of motion present in the downstream datasets. For instance, Gym (10^3) and UB-S1 contain gymnastic videos where actors are often spinning and turning but do not change in scale due to the fixed camera, therefore rotation is more helpful than scaling. We also experiment with combinations of transformations in Appendix C but observe no further improvement.

Number of Tubelets. We investigate the number of tubelets used in each video in Table 5.5. One tubelet is already more effective than temporal contrastive learning, *e.g.*, 29.5% vs. 39.5% for Gym (10^3). Adding two tubelets improves accuracy on all datasets, *e.g.*, +8.5% for Gym (10^3).

Analysis of Motion-Focus. To further understand what our model learns, Figure 5.5 visualizes the class agnostic activation maps [10] without finetuning for the baseline and our approach. We observe that even without previously seeing any FineGym data,

#Tubelets	UCF (10^3)	Gym (10^3)	SSv2-Sub	UB-S1
1	62.0	39.5	47.1	89.5
2	65.5	48.0	47.9	90.9
3	66.5	46.0	47.5	90.9

TABLE 5.5: **Number of Tubelets.** Overlaying two tubelets in positive pairs improves downstream performance.

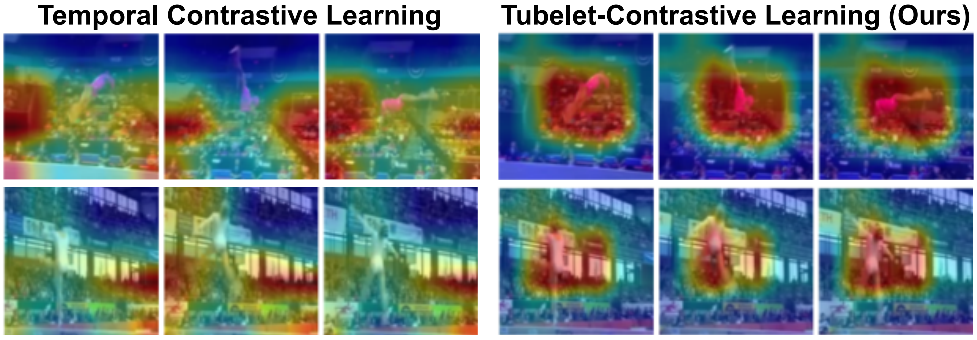

Temporal Contrastive Learning
Tubelet-Contrastive Learning (Ours)

FIGURE 5.5: **Class-Agnostic Activation Maps without Finetuning** for the temporal contrastive baseline and our tubelet-contrast. Our model better attends to regions with motion.

	Linear Classification		Finetuning	
	UCF101	Gym99	UCF101	Gym99
Temporal Contrast	**58.9**	19.7	87.1	90.8
Tubelet Contrast	30.0	**34.1**	**91.0**	**92.8**

TABLE 5.6: **Appearance vs Motion**. Our method learns to capture motion dynamics with pretraining and can easily learn appearance features with finetuning.

our approach attends better to the motions than the temporal contrastive baseline, which attends to the background regions. This observation is supported by the linear classification and finetuning results on UCF101 (appearance-focused) and Gym99 (motion-focused) in Table 5.6. When directly predicting from the learned features with linear classification, our model is less effective than temporal contrast for appearance-based actions in UCF101, but positively affects actions requiring fine-grained motion understanding in Gym99. With finetuning, our tubelet-contrastive representation is able to add spatial appearance understanding and maintain its ability to capture temporal motion dynamics, thus it benefits both UCF101 and Gym99.

5.4.3 Video-Data Efficiency

To demonstrate our method's data efficiency, we pretrain using subsets of the Kinetics-400. In particular, we sample $5\%, 10\%, 25\%, 33\%$ and 50% of the Kinetics-400 training set with three random seeds and pretrain our model and the temporal contrastive baseline. We compare the effectiveness of these representations after finetuning on UCF (10^3), Gym(10^3), SSv2-Sub, UB-S1, and HMDB51 in Figure 5.6. On all downstream setups, our method maintains similar performance when reducing the pretraining data to just 25%, while the temporal contrastive baseline performance decreases significantly. Our method is less effective when using only 5% or 10% of the data, but remarkably still outperforms the baseline trained 100% for Gym (10^3),

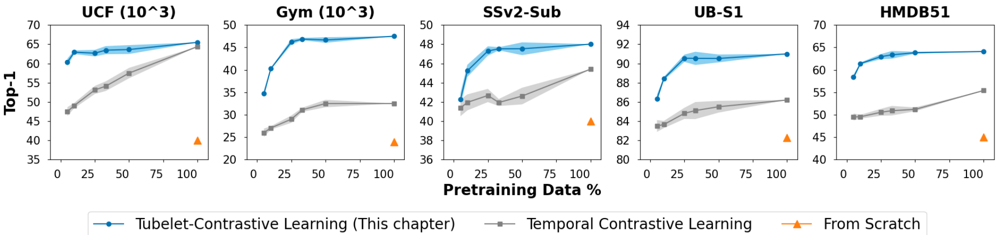


FIGURE 5.6: **Video-Data Efficiency of Tubelet-Contrastive Learning.** Our approach maintains performance when using only 25% of the pretraining data. When using 5% of the pretraining data, our approach is still more effective than using 100% with the baseline for Gym (10^3), UB-S1, and HMDB51. Results are averaged over three pretraining runs with different seeds.

UB-S1, and HMDB. We attribute our model's data efficiency to the tubelets we add to the pretraining data. In particular, our non-linear motion and transformations generate a variety of synthetic tubelets that simulate a greater variety of fine-grained motions than are present in the original data.

5.4.4 Standard Evaluation: UCF101 and HMDB51

We first show the effectiveness of our proposed method on standard coarse-grained action recognition benchmarks UCF101 and HMBD51, where we compare with prior video self-supervised works. For a fair comparison, we only report methods in Table 5.7 that use the R(2+1)D-18 backbone and Kinetics-400 as the pretraining dataset.

First, we observe that our method obtains the best results for UCF101 and HMDB51. The Appendix C shows we also achieve similar improvement with the R3D and I3D backbones. In particular, with R(2+1)D our method beats CtP [237] by 2.6% and 2.4%, TCLR [41] by 2.8% and 4.1%, and TE [101] by 2.8% and 1.9% all of which aim to learn finer temporal representations. This confirms that explicitly contrasting tubelet-based motion patterns results in a better video representation than learning temporal distinctiveness or prediction. We also outperform FAME [45] by 6.2% and 9.6% on UCF101 and HMDB51. FAME aims to learn a motion-focus representation by pasting the foreground region of one video onto the background of another to construct positive pairs for contrastive learning. We however are not limited by the motions present in the set of pretraining videos as we simulate new motion patterns for learning. We also outperform prior multi-modal works which incorporate audio or explicitly learn motion from optical flow. Since our model is data-efficient, we can pretrain on Mini-Kinetics and still outperform all baselines which are trained on the 3x larger Kinetics-400.

Method	Modality	UCF101	HMDB51
VideoMoCo [169]	RGB	78.7	49.2
RSPNet [173]	RGB	81.1	44.6
SRTC [139]	RGB	82.0	51.2
FAME [45]	RGB	84.8	53.5
MCN [138]	RGB	84.8	54.5
AVID-CMA [158]	RGB+Audio	87.5	60.8
TCLR [41]	RGB	88.2	60.0
TE [101]	RGB	88.2	62.2
CtP [237]	RGB	88.4	61.7
MotionFit [69]	RGB+Flow	88.9	61.4
GDT [171]	RGB+Audio	89.3	60.0
This chapter †	RGB	**90.7**	**65.0**
This chapter	RGB	**91.0**	**64.1**

TABLE 5.7: **Standard Evaluation: UCF101 and HMDB51** using R(2+1)D. Gray lines indicate use of additional modalities during self-supervised pretraining. Note that our method pretrained on Mini-Kinetics (†) outperforms all methods which pretrain on the 3× larger Kinetics-400.

5.4.5 SEVERE Generalization Benchmark

Next, we compare to prior works on the challenging SEVERE benchmark [224], which evaluates video representations for generalizability in *domain shift*, *sample efficiency*, *action granularity*, and *task shift*. We follow the same setup as in the original SEVERE benchmark and use an R(2+1)D-18 backbone pretrained on Kinetics-400 with our tubelet-contrast before finetuning on the different downstream settings. Results are shown in Table 5.8.

Domain Shift. Among the evaluated methods our proposal achieves the best results on SSv2 and Gym99. These datasets differ considerably from Kinetics-400, particularly in regard to the actions, environment and viewpoint. Our improvement demonstrates that the representation learned by our tubelet-contrast is robust to various domain shifts.

Sample Efficiency. For sample efficiency, we achieve a good gain over all prior works on Gym (10^3), *e.g.*, +20.7% over TCLR [41] and +14.1% over CtP [237]. Notably, the gap between the second best method GDT [171] and all others is large, demonstrating the challenge. For UCF (10^3), our method is on par with VideoMoCo[169] and CtP but is outperformed by GDT and RSPNet [173]. This is likely due to most actions in UCF101 requiring more spatial than temporal understanding, thus it benefits from the augmentations used by GDT and RSPNet. Our motion-focused representation requires more finetuning samples on such datasets.

Action Granularity. For fine-grained actions in FX-S1 and UB-S1, our method

		Domains		Samples		Actions		Tasks			
	Backbone	SSv2	Gym99	UCF (10^3)	Gym (10^3)	FX-S1	UB-S1	UCF-RC↓	Charades	**Mean**	**Rank↓**
SVT [188]	ViT-B	59.2	62.3	83.9	18.5	35.4	55.1	0.421	35.5	51.0	8.9
VideoMAE [227]	ViT-B	69.7	85.1	77.2	27.5	37.0	78.5	0.172	12.6	58.1	8.3
Supervised [229]	R(2+1)D-18	60.8	92.1	86.6	51.3	79.0	87.1	0.132	23.5	70.9	3.9
None	R(2+1)D-18	57.1	89.8	38.3	22.7	46.6	82.3	0.217	7.9	52.9	11.6
SeLaVi [8]	R(2+1)D-18	56.2	88.9	69.0	30.2	51.3	80.9	0.162	8.4	58.6	11.0
MoCo [85]	R(2+1)D-18	57.1	90.7	60.4	30.9	65.0	84.5	0.208	8.3	59.5	9.1
VideoMoCo [169]	R(2+1)D-18	59.0	90.3	65.4	20.6	57.3	83.9	0.185	10.5	58.6	9.1
Pre-Contrast [218]	R(2+1)D-18	56.9	90.5	64.6	27.5	66.1	86.1	0.164	8.9	60.5	9.0
AVID-CMA [158]	R(2+1)D-18	52.0	90.4	68.2	33.4	68.0	87.3	0.148	8.2	61.6	9.0
GDT [171]	R(2+1)D-18	58.0	90.5	**78.4**	45.6	66.0	83.4	**0.123**	8.5	64.8	8.6
RSPNet [173]	R(2+1)D-18	59.0	91.1	74.7	32.2	65.4	83.6	0.145	9.0	62.6	8.0
TCLR [41]	R(2+1)D-18	59.8	91.6	72.6	26.3	60.7	84.7	0.142	**12.2**	61.7	7.6
CtP [237]	R(2+1)D-18	59.6	92.0	61.0	32.9	79.1	88.8	0.178	9.6	63.2	5.6
This Chapter†	R(2+1)D-18	59.4	92.2	65.5	**48.0**	78.3	90.9	0.150	9.0	66.0	5.4
This Chapter	R(2+1)D-18	**60.2**	**92.8**	65.7	47.0	**80.1**	**91.0**	0.150	10.3	**66.5**	**4.1**

TABLE 5.8: **SEVERE Generalization Benchmark.** Comparison with prior self-supervised methods for generalization to downstream domains, fewer samples, action granularity, and tasks. ↓ indicates lower is better. Results for baselines are taken from SEVERE [224]. Our method generalizes best, even when using the 3x smaller Mini-Kinetics dataset (†) for pretraining.

achieves the best performance, even outperforming supervised Kinetics-400 pretraining. We achieve a considerable improvement over other RGB-only models, *e.g.*, +19.6% and +6.3% over TCLR, as well as audio-visual models, *e.g.*, +14.1% and +7.6% over GDT. These results demonstrate that the video representation learned by our method are better suited to fine-grained actions than existing self-supervised methods. We additionally report results on Diving48 [133] in the Appendix C.

Overall SEVERE Performance. Finally, we compare the mean and the average rank across all generalizability factors. Our method has the best mean performance (66.5) and achieves the best average rank (4.1). When pretraining with the 3x smaller Mini-Kinetics our approach still achieves impressive results. We conclude our method improves the generalizability of video self-supervised representations across these four downstream factors while being data-efficient.

Task Shift. For the task shift to repetition counting, our method is on par with AVID-CMA [158] and RSPNet, but worse than GDT. For multi-label action recognition on Charades, our approach is 3rd, comparable to VideoMoCo but worse than TCLR. This suggests the representations learned by our method are somewhat transferable to tasks beyond single-label action recognition. However, the remaining gap between supervised and self-supervised highlights the need for future work to explore task generalizability further.

Comparison with Transformers. Table 5.8 also contains recent transformer-based self-supervised works SVT [188] and VideoMAE [227]. We observe that both SVT and VideoMAE have good performance with large amounts of finetuning data (SSv2), in-domain fine-tuning (UCF(10^3)), and multi-label action recognition (Charades). However, they considerably lag in performance for motion-focused setups Gym99, FX-S1, UB-S1, and repetition counting compared to our tubelet contrast with a small CNN backbone.

5.5 Conclusion

This chapter presents a contrastive learning method to learn motion-focused video representations in a self-supervised manner. Our model adds synthetic tubelets to videos so that the only similarities between positive pairs are the spatiotemporal dynamics of the tubelets. By altering the motions of these tubelets and applying transformations we can simulate motions not present in the pretraining data. Experiments show that our proposed method is data-efficient and more generalizable to new domains and fine-grained actions than prior self-supervised methods.

Appendix A

Supplementary Materials for Skeleton-Contrastive Learning

In this Appendix, we provide details on the training procedure for each downstream task in Section A.1 and a comparison of our method to supervised-approaches for skeleton-based action recognition in Section A.2. We examine the effect of the hyperparameters of our proposed augmentations in Section A.3. Finally, we show the performance of combining multiple-skeleton representations for the downstream task of action recognition in Section A.4 and provide some qualitative results of our method in Section A.5.

A.1 Downstream Training Details

For the downstream tasks we follow Chen *et al.* [29] and remove the projection head of the pre-trained query encoder, as the projection head tends to focus mostly on information specific to the pretext task. For the 3D action recognition tasks, we then append a classifier to the pre-trained query encoder, while for 3D action retrieval we directly use the feature space without adding a classification head. The dimensionality of the feature space is dependent on the input skeleton-representation used in the downstream task. It is either 4096 (for X^{IMG}), 2048 (for X^{SEQ}) or 256 (for X^{STG}). For downstream tasks we use a temporal crop of length 64. During training this is sampled randomly, while for evaluation we sample a center crop.

3D Action Recognition. For this task, the weights of the pre-trained encoder are frozen and only the linear classifier is trained as in [136, 165]. An SGD optimizer is used with a momentum of 0.9 and learning rate of 0.1. The linear classifier is trained for a total of 80 epochs and learning rate is reduced by a factor of 10 after the 50th and 70th epoch.

3D Action Retrieval For this task we follow [211] to extract the encoder features of the training set. Then, we apply a kNN classifier with k=1 using these features and their corresponding action labels to assign action classes. Finally, during testing we assign to the unseen sample the action class of the closest neighbour in the training set.

Semi-Supervised 3D Action Recognition. For this task, we finetune both the classifier and the pre-trained encoder weights jointly as in [136]. An Adam optimizer is used to train the network for a total of 50 epochs with a learning rate of 0.0001, which is reduced by a factor of 10 after both the 30th and 40th epoch.
Transfer Learning for 3D Action Recognition. For this task we again follow [136] and finetune the classifier and the pre-trained encoder together. An Adam optimizer is used to train the network for a total of 50 epochs with a learning rate of 0.0001 which is reduced by a factor of 10 after 30th and 40th epoch.

A.2 Supervised Approaches

While our method outperforms prior self-supervised learning works for 3D action recognition, it is also useful to know how this compares to state-of-the-art supervised approaches. Table A.1 shows the performance of various supervised approaches on the NTU 60 & 120 datasets. We compare these results to the performance of our sequence-based query encoder f^{SEQ} (a simple 3-layer Bi-GRU) trained end-to-end from randomly initialized weights (supervised-only) and finetuned end-to-end from the weights learnt from our inter-skeleton contrastive learning approach (with pre-training). Note that this setting is different to the experiment performed in the main chapter, which only finetunes the final layer in order to demonstrate the raw performance of the features, rather than the boost they can provide to supervised training. It is evident from the table our method is competitive with many supervised approaches, even though the encoder we use is not state-of-the-art. It is also clear that our contrastive pre-training can boost the performance over supervised-only training. It is likely our inter-skeleton contrastive pre-training can also be used to boost the performance of more complex state-of-the-art encoders too.

A.3 Augmentation Hyperparameter Ablations

In this section we study the impact of hyperparameters $|j|$ and L_{ratio} of the spatial joint jittering and temporal crop-resize augmentations on the downstream performance. We use X^{IMG} skeleton representation and evaluate on the cross-view protocol of NTU RGB+D 60 for the downstream task of 3D action classification. We first pre-train an intra-contrastive framework using X^{IMG} representation with only the relevant augmentation and then train a linear classifier with action labels on top of the frozen features of the query encoder f_q.

A.3.0.1 Effect of number joints to jitter $|j|$

Here, we ablate over the number of joints to jitter $|j|$ in our joint jittering augmentation. This parameter controls the number of joints to be jittered for the augmented view. Table A.2 shows the downstream 3D action classification performance of different

	NTU RGB+D 60		NTU RGB+D 120	
Method	x-view	x-sub	x-setup	x-sub
PA-LSTM [196]	52.8	50.1	26.3	25.5
ST-LSTM [142]	77.7	69.2	57.9	55.7
GCA-LSTM [140]	84.0	76.1	59.2	58.3
VA-LSTM [278]	87.7	79.4	-	-
ST-GCN [260]	88.3	81.5	73.2	70.7
Shift-GCN [32]	96.5	90.7	85.9	87.6
MS-G3D Net [145]	96.2	91.5	86.9	88.4
This chapter (supervised-only)	87.8	72.9	68.2	66.3
This chapter (with-pretraining)	90.4	79.3	75.4	73.1

TABLE A.1: **Comparison with supervised only training** for 3D action recognition. Pre-training with our inter-skeleton contrast improves the performance over supervised only training, especially for the more challenging cross-subject and cross-setup protocols.

values of $|j|$. We found that jittering around half the joints ($|j|$=10, 15) performed best. Using very small or a large values for $|j|$ *e.g.* 2 or 20 is sub-optimal as with too few jittered joints the augmented views become highly similar, while with many jittered joints there remains little commonality between the augmented sequences. For all our experiments we use $|j|$=15 in our joint jittering augmentation as it achieve best downstream performance.

	Number of jittered joints $|j|$				
Augmentation	2	5	10	15	20
Spatial-Jittering	65.6	67.5	69.4	**74.6**	70.6

TABLE A.2: **Effect of number of joints to jitter** on the downstream task of 3D action classification on cross-view protocol of NTU RGB+D 60. Increasing the number of joints to jitter improves the downstream performance.

A.3.0.2 Effect of temporal length ratio L_{ratio}

We next ablate over the distribution from which temporal length ratio $L_{ratio} \in [l_{min}, 1.0]$ is sampled in our temporal crop-resize augmentation, see Equation equation 3.3. The parameter l_{min} controls the minimum length of the temporal crop, which can be sampled for the augmented view. Table A.3 shows the 3D action classification performance with different minimum samples lengths l_{min}. A smaller l_{min}, and thus a larger temporal range improves the downstream performance. We therefore use l_{min}=0.1, *i.e.* $L_{ratio} \in [0.1, 1.0]$, for all our experiments.

	l_{min}		
Augmentation	0.1	0.3	0.5
Temporal Crop-Resize	**62.5**	62.0	60.8

TABLE A.3: **Effect of temporal length ratio** on the downstream task of 3D action classification on cross-view protocol of NTU RGB+D 60. The bigger the range, the better the downstream performance.

A.4 Multi-representation Downstream

In this section, we examine the effect of combining skeleton representations when finetuning for the downstream task of 3D action recognition. All of our previous results use only one representation in the downstream task for efficiency, even when representations are trained together in our inter-skeleton contrast. Here we report the results of combining representations in the downstream task for both intra and inter-skeleton contrast. For intra-skeleton, each skeleton representation is first pre-trained separately (see Section 3.3.2) and then their query encoders are combined for the downstream task. For inter-skeleton two skeleton representations are pretrained together (see Section 3.3.3) with their query encoders also combined for the downstream task. Table A.4 shows the results of these experiments alongside the results when using only one representation during the downstream task from Table 3.6. We again evaluate on the cross-view protocol of NTU RGB+D 60 by training a linear classifier on frozen features. In Table A.4 we also highlight the number of parameters needed for each representation in this downstream task. The downstream encoders (i.e query encoders) for the skeleton representations are as a 3-Layer BI-GRU with H=1024 units) for X^{SEQ}, an HCN [128] model for X^{IMG} and a joint-based A-GCN [199] network for X^{STG} (see Implementation details Section 3.4.2).

From Table A.4, we first observe when combining representations in the downstream task pretraining with inter-skeleton contrast outperforms the intra-skeleton pretraining for all combinations. In this setting both the computational costs required for pretraining and inference of the intra and inter-skeleton contrast are same as training two representations separately requires the same computation as training them together (inter), thereby showing the superiority of our inter-skeleton contrast.

As we saw in the main chapter, inter-skeleton contrast shows considerable improvement in performance over the intra-skeleton contrast for the each single representation downstream evaluation, with X^{SEQ} obtaining the best results. However, it is worth noting that while the number of parameters required for inference are the same, the inter-skeleton does require additional computational resources for pretraining since each representation is required to be pre-trained with one of the other skeleton representations while in intra-skeleton each representation is pretrained alone.

We also observe that combining representations in the downstream task improves over using a single representation in the majority of cases, with the combination

X^{SEQ} and X^{STG} showing the best results. Note that this improvement comes with an additional cost of model size during the inference time. With these results we can conclude that our model can be used for all skeleton representations individually or in combination based on the trade off between the pretraining computational cost, the inference model size and the performance.

Downstream Reps.	Intra	Inter	# Inference Params.
X^{IMG}	79.6	81.7	1.0M
X^{STG}	72.5	78.9	3.0M
X^{SEQ}	82.5	**85.2**	10.0M
$X^{IMG} + X^{STG}$	80.3	81.8	4.0M
$X^{IMG} + X^{SEQ}$	80.3	82.6	11.0M
$X^{SEQ} + X^{STG}$	84.5	**86.0**	14.0M

TABLE A.4: **Combining representations for 3D action recognition.** We show the trade-off between accuracy and number of parameters involved in the downstream task when using two representations to fine-tune the features learnt from both intra and inter-skeleton pretraining. Pretraining with our inter-skeleton contrast learns better features for each representation whether used individually or combined.

A.5 Qualitative Results

A.5.0.1 Visualization of learned features

First we visualize the features by our inter-skeleton contrastive learning in comparison to those learned by Su *et al.* [211], one of the best performing methods on both the 3D action recognition and retrieval tasks. We randomly select 10 of the 60 action classes, so as not to overcrowd the figure, and plot their features using t-SNE. This is repeated three times for three different subsets of action classes. We observe from the Figure A.1 that the features learned by our method form better clusters and are therefore more discriminatory and more suitable for the downstream tasks of action recognition and retrieval.

A.5.0.2 3D Action retrieval results

In Figure A.2, we visualize the results of 3D action retrieval. For a given query video we retrieve the top four nearest neighbours in the feature space learned by Su *et al.* [211] and by our inter-skeleton contrastive learning. We observe from Figure A.2 that the nearest neighbours in the feature space are generally more relevant to the query when using our method. The videos retrieved by Su *et al.* tend to be from different actions, which contain similar body poses. For instance ‘kicking’, ‘staggering’ and ‘hop on one leg’ all contain poses with one leg off the floor. Instead, our method is

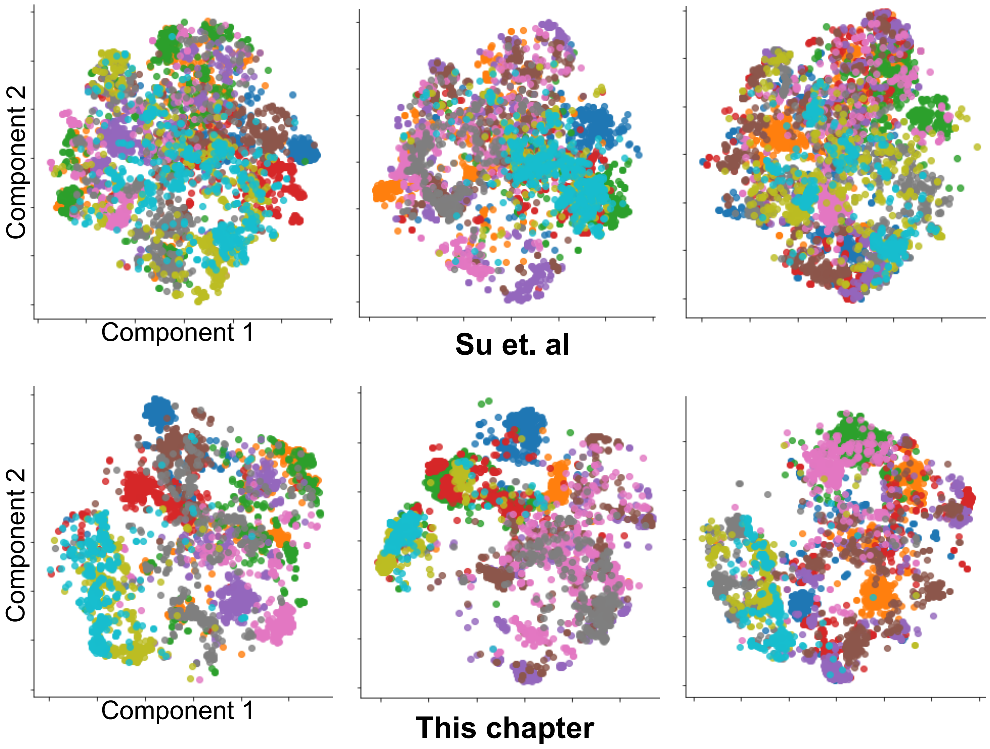


FIGURE A.1: **t-SNE visualization** of learned features on NTU RGB+D 60 dataset. Each plot shows the features of 10 randomly selected action classes. Top row shows the features learned by Su *et al*. [211] and bottom row shows the corresponding features learned by our inter-skeleton contrastive learning. Our methods learns a more discriminatory feature space forming better clusters which are more dense with most samples from same action class and distant from other clusters as compared to [211].

able to better focus on the motion of the query action and retrieve other instances of the same action.

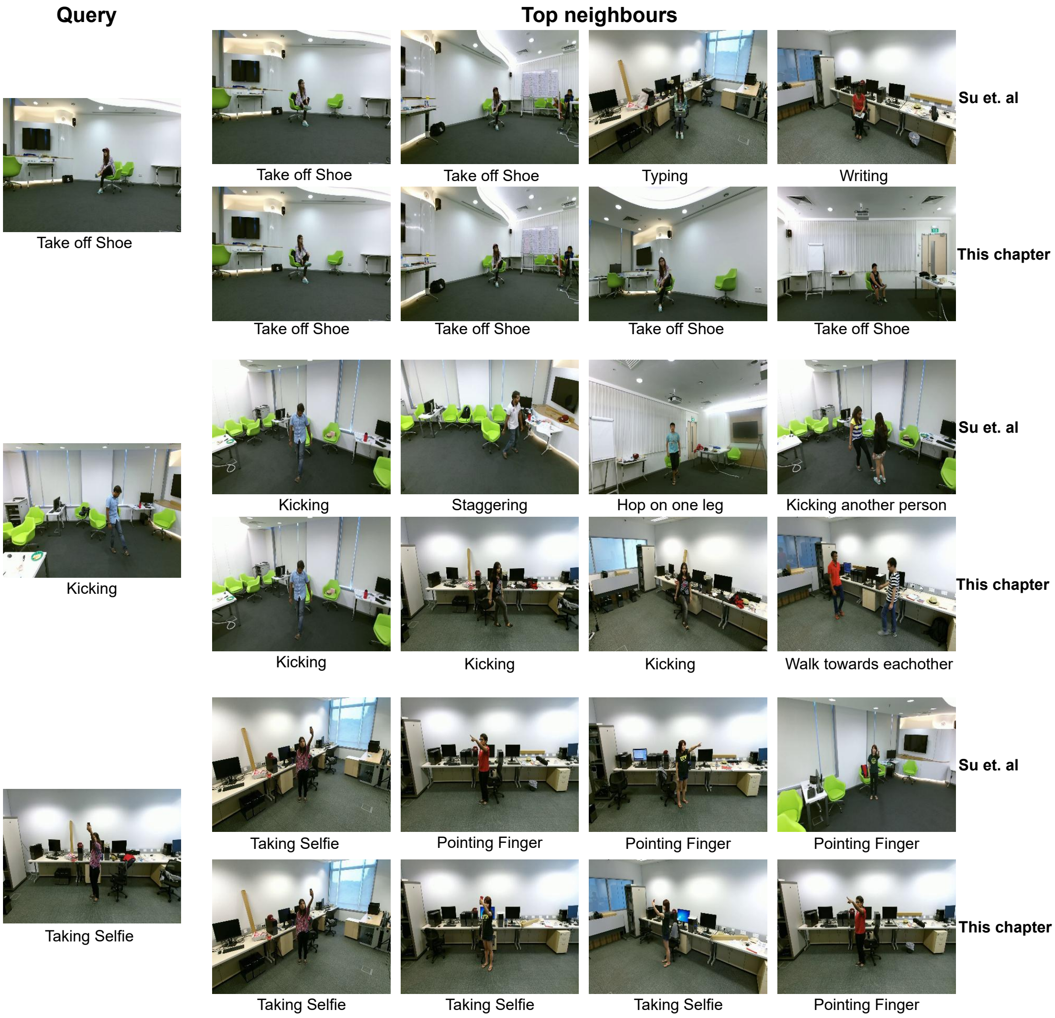


FIGURE A.2: **3D Action retrieval results** on NTU RGB+D 60 dataset. For each query, the first row shows nearest neighbours learned by Su *et al*. [211] and the second row shows the nearest neighbours in the feature space learnt by our inter-skeleton contrastive learning. For our method most neighbours belong to the same action classes. All results were obtained using 3D skeleton data, however, for the ease of visualization/interpretation we show the corresponding RGB videos instead of the skeleton sequences.

Appendix B

Supplementary Materials for Benchmark Sensitivity

In Appendix B.1, we provide details of the video self-supervised models we use in our evaluation study. Appendix B.2 provides details on the experimental setup for each of our downstream sensitivity factors. We also show correlation plots between current benchmarks and the experimental results for each sensitivity factor in Appendix B.3. Feature similarities between supervised pre-training and each self-supervised pre-training method are shown in Appendix B.4. In Appendix B.5, we describe domain difference between the downstream video datasets we use and the attributes we use to characterize this difference. We show the standard deviations of the experiments on the SEVERE benchmark Appendix B.6 and also compare the SEVERE benchmark to results on HMDB51 action recognition in Appendix B.7. Finally, we report results of some additional experiments in Appendix B.8 and Appendix B.9 that we did not have room for in the main paper.

B.1 Details of the Evaluated Self-Supervised Models

We use a variety of different self-supervised methods in our paper, here we describe each method:
MoCo [30] is a contrastive learning method proposed for representation learning in images. Positives are created by performing different spatial augmentations on a video. Negatives are other videos. To obtain negatives beyond the current batch, MoCo proposes a momentum encoder which maintains a queue of momentum-updated data samples from previous batches.
SeLaVi [8] views the audio and visual modalities as different augmentations of a video and learns with a cross-modal clustering pretext task.
VideoMoCo [169] extends MoCo to the temporal domain. It does this with an adversarial dropout augmentation which removes the frames the model considers most important. With the contrastive learning loss, the model learns invariance to this adversarial frame dropout alongside the spatial augmentations used in MoCo.
Pretext-Contrast [217] combines the pretext task approach with contrastive learning. As its pretext task it uses video cloze procedure [147] where the goal is to predict which

augmentations have been applied to a video clip. For the contrastive learning objective different temporal shifts, *i.e*. distinct clips from the same video, are considered.

RSPNet [173] also combines pretext and contrastive tasks, with a focus on video speed. The pretext task is to predict the relative difference in speed between two versions of the same video, while the contrastive task creates extra positives and negatives by augmenting videos with different speeds along with the spatial augmentations.

AVID-CMA [158] is a multi-modal contrastive learning method which uses audio in addition to the visual modality. It first uses cross-modal contrastive learning where the one modality is used as the positives and the other as the negatives. Then it uses within modality contrastive learning where additional positives which have high audio and visual similarity are sampled.

CtP [237] performs self-supervised learning through a “catch the patch” pretext task. The goal in this task is to predict the trajectory of an image patch which is resized and moved through a sequence of video frames.

TCLR [41] is a contrastive method which encourages features to be distinct across the temporal dimension. It does this by using clips from the same video as negatives. Therefore, instead of encouraging invariance to temporal shift as other methods to, it encourages the model to be able to distinguish between different shifts. It also uses an extensive set of spatial augmentations.

GDT [171] is a multi-modal contrastive method which composes a series of different augmentations and encourages model to learn invariance to some and learns to distinguish between others. We use the best performing version of GDT which encourages invariance to spatial augmentations, the audio and visual modalities and temporal reversal, while encouraging the model to distinguish between different temporal shifts.

While all models are pre-trained on Kinetics-400 and use an R(2+1)D-18 backbone with 112x112 spatial input size, there are some smaller differences in how the models are trained. Due to the computational cost of training these models we download publicly available models or obtain them from the authors, therefore we cannot control for these smaller differences in the pre-training set up. These differences include number of pre-training epochs, batch size, number of video frames used and spatial and temporal augmentations. We list these differences in Table B.1.

B.2 Downstream Experimental Details

B.2.1 Downstream Domain

In Section 4.3 we investigate to what extent self-supervised methods learn features applicable to action recognition in any domain. Here we explain the datasets, splits and training details we use to do this.

Datasets We report our experiments on the following datasets:

UCF-101 [209] is currently one of the most widely used datasets for evaluating video self-supervised learning models. It consists of YouTube videos from a set of 101 coarse-grained classes with a high overlap with actions in Kinetics-400. We use the

Method				**Spatial Augmentations**						**Temporal Augmentations**		
	Epochs	Batch Size	Num Frames	Random Crop	Horiz. Flip	Grayscale	Color Jitter	Gaussian Blur	Scaling	Shift	Reversal	Speed
MoCo	200	128	16	✓	✓	✓	✓			✓		
SeLaVi †	200	1024	30	✓	✓							
VideoMoCo	200	128	32	✓	✓	✓	✓					
Pretext-Contrast	200	16	16	✓	✓	✓	✓	✓		✓		
RSPNet	200	64	16	✓			✓	✓		✓		✓
AVID-CMA †	400	256	16	✓	✓		✓		✓			
CtP	90	32	16									
TCLR	100	40	16	✓	✓	✓	✓		✓			
GDT †	100	512	30	✓	✓		✓				✓	
Supervised	45	32	16	✓	✓					✓		

TABLE B.1: **Pre-training differences of our evaluated self-supervised methods.** While all models are pre-trained with the same backbone and dataset, there are differences in how many epochs they were trained for, the batch size and number of frames they use and the spatial and temporal augmentations they are encouraged to be invariant to. (†) represents methods that use Audio as the extra modality during pretraining.

first standard split proposed in the original paper [209] containing 9,537 training and 3,783 testing samples for the 101 action classes.
NTU-60: [195] consists of daily human actions captured in a controlled lab setting with a fixed number actors. Although it has some overlap with Kinetics-400 actions, it is quite different visually due to the setting. We use the cross-subject protocol proposed in [195] to split the data into 40,320 training and 16,560 testing samples for 60 action classes.
Gym-99. We use FineGym version $v1.0$ [197] which is a dataset of fine-grained actions constructed from recorded gymnastic competitions. We use the Gym 99 subset which contains 99 action classes with 20,484 and 8,521 samples in the train and test sets respectively.
SS-v2: [74] is a crowdsourced collection of first-person videos aimed to instill common-sense understanding. It differs significantly with respect to Kinetics-400 in terms of visual appearance and point-of-view. We use the original dataset splits from [74] containing 168,913 training and 24,777 testing samples for 174 action classes.
EPIC-Kitchens-100: [39] is a large-scale egocentric dataset consisting of daily actions performed in a kitchen. It has annotations for verbs (97) and nouns (300) and the action is defined a tuple of these. Like SS-v2, EK-100 also differs significantly from Kinetics-400 in terms of visual appearance and point-of-view. We use standard splits from [39] containing 67,217 samples in training set and 9,668 in the validation set. In the main paper we only aim to recognize the 97 verb classes, we provide results for the noun and action recognition tasks in Appendix B.9.
Training Details In the initial hyper-parameter search, we perform a grid search over various finetuning settings with learning rates between 0.1 - 0.00001, varying total training epochs, data augmentations, and schedulers. We choose the optimal hyper-parameters based on the performances of the pretraining models on the validation sets of each dataset for each downstream task.

During training, we sample a random clip from each video of 32 frames with standard augmentations *i.e.* a random multi-scale crop of size 112x112, random

Dataset	**Finetuning**				**Linear Evaluation**			
	Batch Size	Learning rate	Epochs	Steps	Batch Size	Learning rate	Epochs	Steps
UCF-101	32	0.0001	160	[60,100,140]	64	0.01	100	[40,80]
NTU-60	32	0.0001	180	[90, 140, 160]	64	0.01	120	[40,80,100]
Gym-99	32	0.0001	160	[60,100,140]	64	0.01	120	[40,80,100]
SS-v2	32	0.0001	45	[25, 35, 40]	64	0.01	40	[20,30]
EK-100	32	0.0025	30	[20, 25]	32	0.0025	30	[20, 25]
K-400	-	-	-	-	64	0.01	40	[10,20,30]

TABLE B.2: **Training details** of finetuning and linear evaluation on various downstream datasets. Learning rate is scheduled using a multip-step scheduler with $\gamma = 0.1$ at corresponding steps for each dataset. We train all the models with same hyperparameters for the corresponding dataset.

horizontal flipping and color jittering. We train with the Adam optimizer. The learning rates, scheduling and total number of epochs vary across datasets and are shown in Table B.2. However, each model is trained with the same hyper-parameters for the corresponding dataset. For inference, we use 10 linearly spaced clips of 32 frames each. For each frame we take a center crop which is resized to 112x112 pixels. To calculate the action class prediction of a video, we take the mean of the predictions from each clip and report top-1 accuracy.

B.2.2 Downstream Samples

In Section 4.4 we measure how sensitive current video self-supervised models are to the amount of downstream samples. We do this by varying the size of the training data starting from 1000 examples and doubling it until we reach the full train set. We use the same data splits as in the downstream domain experiments, explained in Appendix B.2.1, and sample a subset of video clips from the respective train sets. We use the same random subset across the different models to make the comparison fair. For each dataset, we use same training and testing procedure as the downstream domain experiments, explained in Appendix B.2.1 and Table B.2.

B.2.3 Downstream Actions

In Section 4.5 we measure how benchmark-sensitive current video self-supervised models are to downstream actions. We do so by measuring performance on different subsets, defined in the FineGym dataset [197], which have increasing semantic similarity. We provide the details of Gym-99, Gym-288 and the four different subsets we use of Gym-99 below:

Gym-99 consists of 29k video clips of 99 different actions across the four different gymnastic events in FineGym: Vault, Floor Exercise, Balance Beam and Uneven Bars. This is a relatively balanced subset of the full FineGym dataset with all actions having more than 80 occurrences. There are a total 20.5k training videos and 8.5k testing videos.

Vault is a subset of Gym 99 containing 1.5k videos of the 6 actions from the Vault event. The training split contains 1.0k examples and the testing split contains 0.5k examples.
Floor contains actions in the Floor Exercise event from Gym-99. It consists of 7.5k instances of over 35 actions with a split of 5.3k for training and 2.2k for testing.
FX-S1 is a subset of actions of leaps, jumps and hops from the Floor event in Gym-99. This subset of 11 actions contains a total of 2.6k video clips with 1.9k for training and 0.7k for testing.
UB-S1 contains 5k videos of 15 actions from the Uneven Bars event with a split of 3.5k for training and 1.5k for testing. The actions consist of different types of circles around the bars.
Gym-288 is a long-tailed version of Gym 99 containing 32k videos with 22.6K training and 9.6K testing samples. It adds 189 infrequent classes to the 99 classes in Gym 99, where actions can have as little as 1 or 2 instances in training. This results in a total of 288 action classes from the four different gymnastic events.

We follow the same training and evaluation procedure as that for finetuning Gym-99 in downstream domain training. In particular, for training we sample a random clip from each video of 32 frames with standard augmentations *i.e.* a random multi-scale crop of size 112x112, random horizontal flipping and color jitter. Each model is trained with the Adam optimizer using a learning rate of 0.0001 and multi-step scheduler with γ=0.1 at epochs [60, 100, 140] for 160 epochs. For inference, we use 10 linearly spaced clips of 32 frames each. For each frame we take a center crop which is resized to 112x112 pixels. To calculate the action class prediction of a video, we take the mean of the predictions from each clip. For each subset, we compute accuracy per action class and report the mean over all action classes as in the original dataset [197].

B.2.4 Downstream Tasks

In Section 4.6 we investigate how sensitive self-supervised methods are to the downstream task and whether they generalize beyond action recognition. We provide details of the experimental setup used for each task below.
Spatio-temporal action detection. The goal of this task is to predict the bounding box of an actor in a given video clip, both spatially and temporally, along with the action class. We use the UCF101-24 benchmark which is a subset of UCF-101 with bounding box annotations for 3,207 videos from 24 action classes. We follow the implementation of Köpüklü *et al.* [118] using only a 3D-CNN branch for spatio-temporal action detection. We initialize the 3D backbone with the pre-trained, self-supervised R(2+1D)-18 models. A clip size of 16 frames is sampled from the video as the input with standard data augmentations *i.e.* horizontal flipping, random scaling and random spatial cropping. Each model is trained using the Adam optimizer with an initial learning rate of 1e-4, weight decay of 5e-4 and batch size 64, for a total of 12 epochs. The learning rate is decayed using a multi-step scheduler with γ=0.5 at

epochs [4,6,8,10]. For testing we also follow [118] and report video-mAP over all the action classes.

Repetition counting. The goal of the this task is to estimate the number of times an action repeats in a video clip. We use the UCFRep benchmark proposed by Zhang *et al*. [277], which is a subset of UCF-101. The dataset consists of 526 videos with 3,506 repetition number annotations. From the annotated videos, 2M sequences of 32 frames and spatial size 112x112 are constructed which are used as the input. We use the implementation from the original benchmark [277] with pre-trained R(2+1)D-18 models as the backbone networks. Each model is trained for 100 epochs with a batch size of 32 using the Adam optimizer with a fixed learning rate of 0.00005. For testing, we follow the protocol from [277] and report mean counting error.

Arrow-of-time. The goal of this task is to predict the direction (forward of backward) of the video. We closely follow the setup used by Ghodrati *et al*. [71]. The full UCF-101 dataset is used with two versions of each video, one normal and one reversed. During training, for each video, we sample 8 frames linearly with a random offset, with batch size of 12 and 112x112 center crops, number of epochs 10, learning rate of $1e^{-5}$. We do not use any augmentations or learning rate schedulers. During testing, we sample 8 frames linearly. We report top-1 binary classification accuracy.

Multi-label classification on Charades. Charades [202] is made up of videos of people recording everyday activities at their homes. Videos in Charades are longer than the other datasets we use and the goal is to recognize multiple different actions in each video. A per-class sigmoid output is used for multi-class prediction. We use the implementation of Feichtenhofer *et al*. [60][1] with the R(2+1)D-18 backbone. During training, we use 32 frames with a sampling rate of 8. Since this task requires longer temporal context, we observe that using more frames with higher sampling rate is beneficial. We use a spatial crop of 112x112 and augmentations such as random short-side scaling, random spatial crop and horizontal flip. We train for 57 epochs in total with a batch size of 16 and a learning rate of 0.0375 with multi-step scheduler with $\gamma = 0.1$ at epochs [41, 49]. During testing, following [60], we spatio-temporally max-pool predictions over 10 clips for a single video. We report mean average precision (mAP) across classes.

Action detection on AVA. AVA [77] consists of clips extracted from films. We use version v2.2 with bounding box annotations for spatio-temporal action detection of temporally fine-grained action classes. The goal of this task is to detect and predict action classes from proposals generated by off-the-shelf person detectors. We again use the implementation of [60] with the R(2+1)D-18 backbone. During training, we use 32 frames with a sampling rate of 2. We use spatial crop of 112x112 and augmentations such as random short-side scaling, random spatial crop, horizontal flip. We train for 20 epochs with learning rate of 0.1 with multi-step scheduler with $\gamma = 0.1$ at epochs [10, 15] and a batch size of 32. During testing, following [60], we use a single clip at the center of the video with 8 frames and sampling rate of 8. We report mean average precision (mAP) across the classes.

[1] https://github.com/facebookresearch/SlowFast

B.3 Correlations of Downstream Performance

As observed from the results of Section 4.3, the performance for both UCF-101 finetuning and Kinetics-400 linear evaluation is not indicative of how well a self-supervised video model generalizes to different downstream domains, samples, actions and tasks. Here, we plot the performance of each pre-trained model for each downstream settings and show the correlation with UCF-101 finetuning and Kinetics-400 linear evaluation performances. The results are shown in Figures B.1-B.8. These plots further demonstrate that the correlations are overall low for each downstream factor *i.e*. domain, samples, actions and tasks, indicating that more thorough testing of video self-supervised methods is needed.

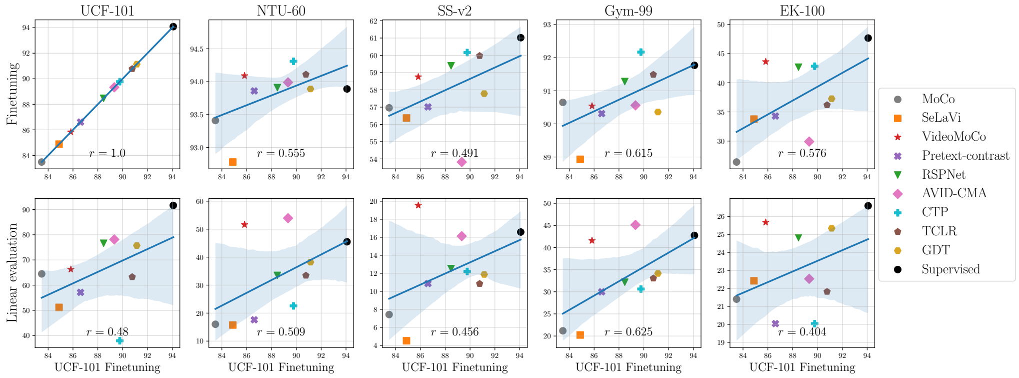


FIGURE B.1: **Downstream domain against UCF-101 finetuning.** We plot the corelations between finetuning performance of video pre-training methods on UCF-101 and performances on finetuning and linear-evaluation on all downstream datasets.

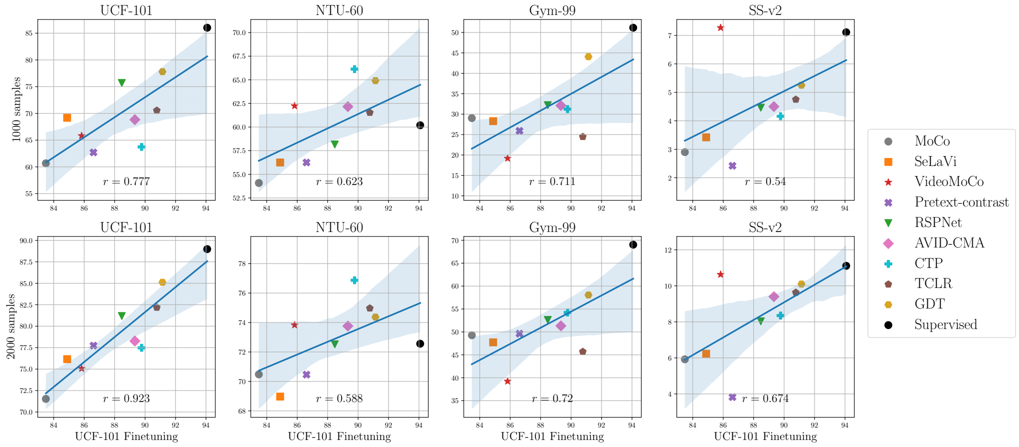


FIGURE B.2: **Downstream samples against UCF-101 finetuning.** For the low data setting (1000-2000 samples), we plot the correlations of performance of video pre-training methods against that for UCF-101 finetuning.

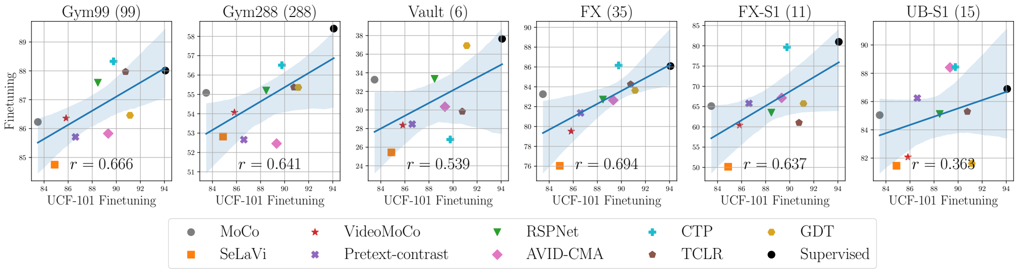


FIGURE B.3: **Downstream actions against UCF-101 finetuning.** We plot the corelations of performances of video pre-training methods between UCF-101 finetuning and FineGym subsets.

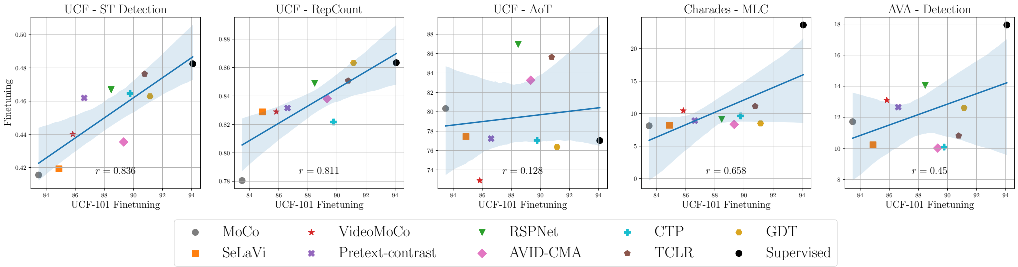


FIGURE B.4: **Downstream tasks against UCF-101 finetuning.** We plot the corelations between performance on UCF-101 finetuning and other downstream tasks for the video pre-training methods.

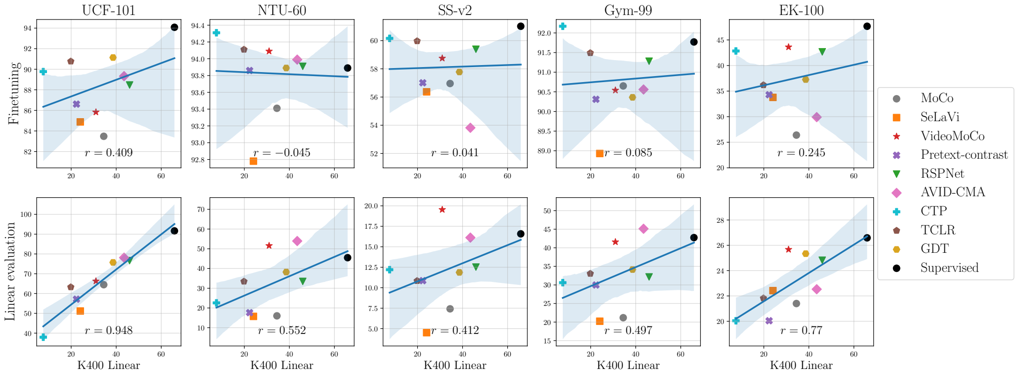


FIGURE B.5: **Downstream domain against Kinetics-400 linear evaluation.** We plot the corelations between finetuning performance of video pre-training methods on Kinetics-400 linear-evaluation and performances on finetuning and linear-evaluation on all downstream datasets.

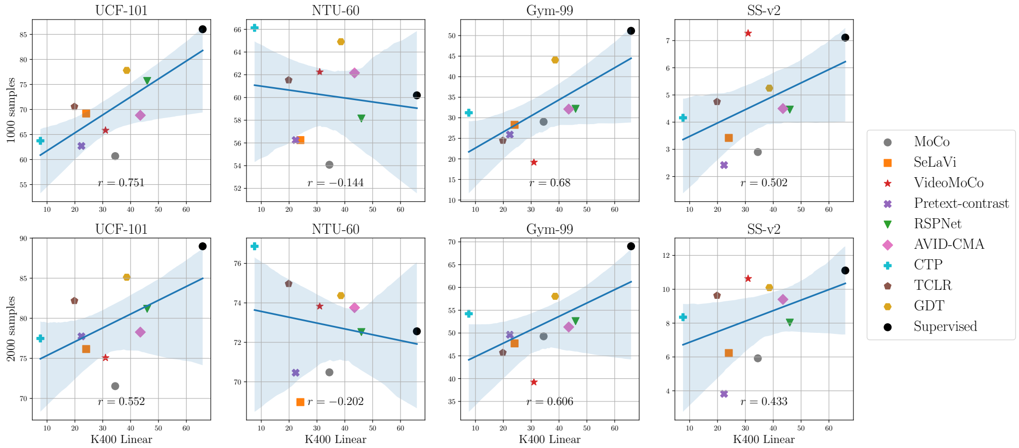


FIGURE B.6: **Downstream samples against Kinetics-400 linear evaluation.** For the low data setting (1000-2000 samples), we plot the correlations of performance of video pre-training methods against that for Kinetics-400 linear-evaluation.

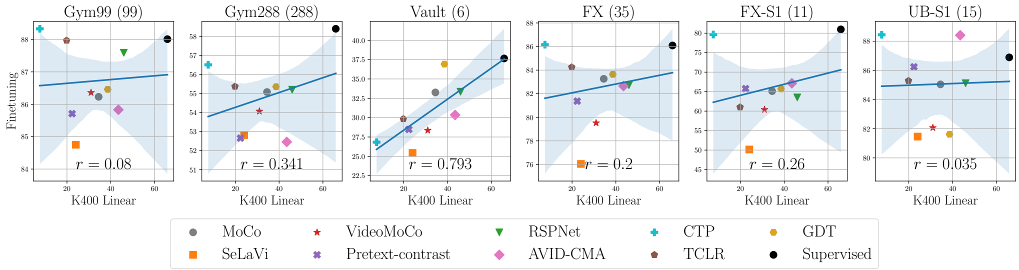


FIGURE B.7: **Downstream actions against Kinetics-400 linear evaluation.** We plot the corelations of performances of video pre-training methods between Kinetics-400 linear-evaluation and FineGym subsets.

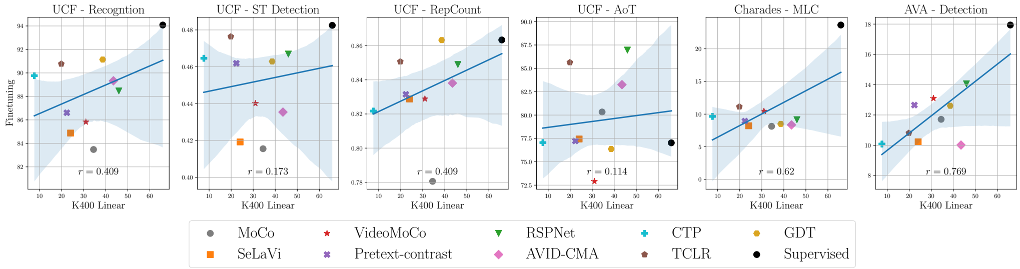


FIGURE B.8: **Downstream tasks against Kinetics-400 linear evaluation.** We plot the corelations between performance on Kinetics-400 linear-evaluation and other downstream tasks for the video pre-training methods.

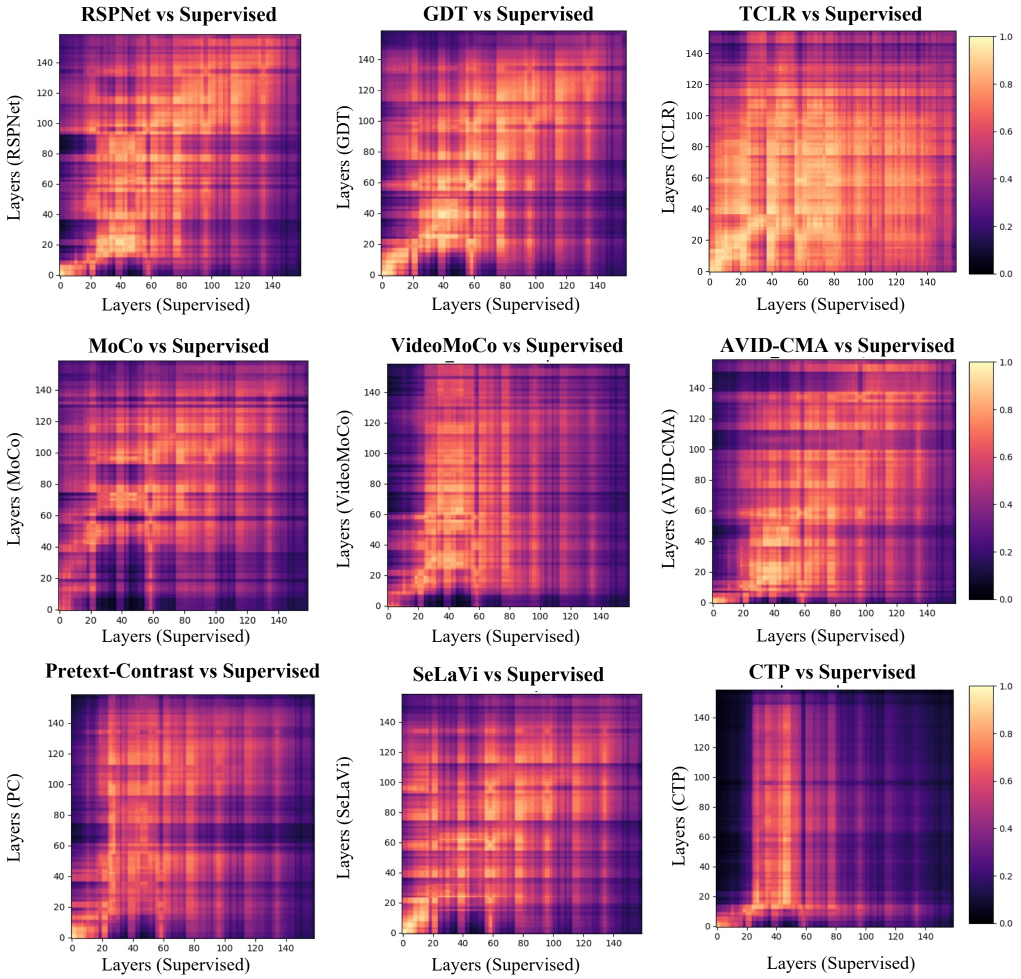


FIGURE B.9: **Representation similarity** between features of self-supervised methods and supervised pre-training on Kinetics-400 validation set using centered kernel alignment. Features of contrastive methods are more closer to the features of supervised pretraining.

B.4 Representation Similarity Matrices

We plot the the feature similarity on Kinetics validation set using centered kernel alignment [162] between supervised pre-training and our evaluated self-supervised pre-training methods in Figure B.9. We showed a subset of these plots in Figure 4.4, here we show the feature similarity for all the self-supervised models we used in our experiments.

B.5 Downstream Dataset Attributes

We define several attributes in Section 4.2.1 in order to characterize differences in domain between the downstream datasets and the Kinetics-400 pre-training dataset

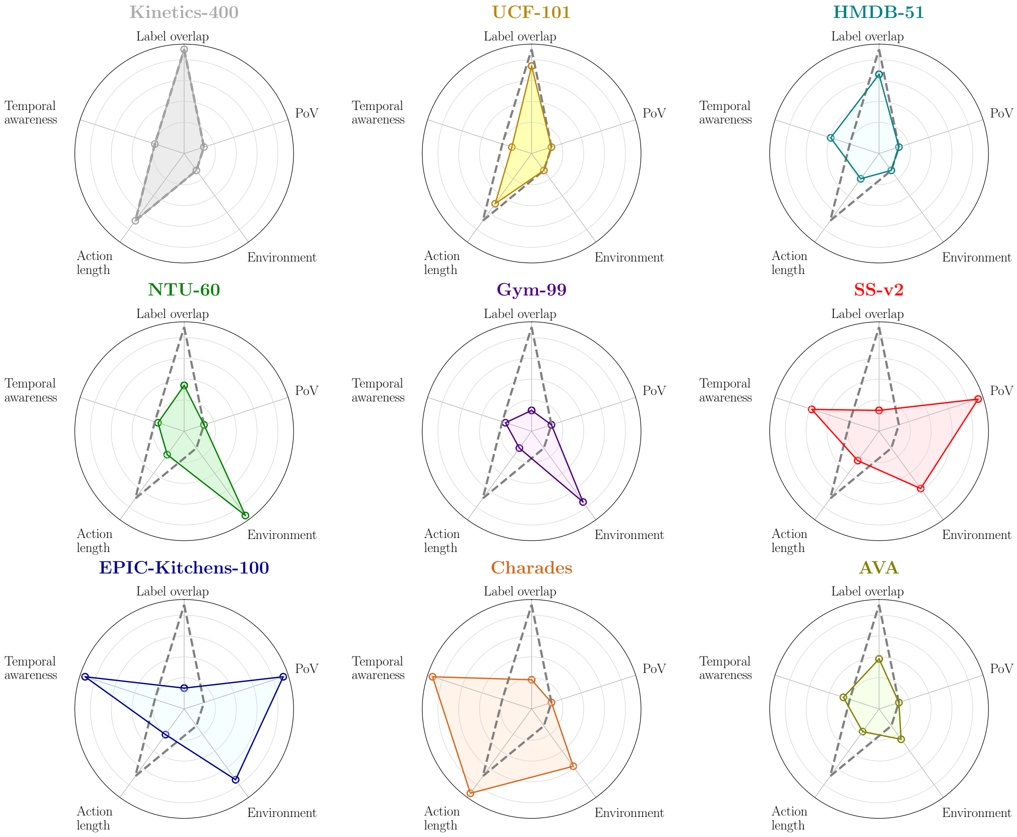


FIGURE B.10: **Radar plots with details**. The radar plots contain details of the values along the axis for every attribute for the datasets we use in this study.

in Figure 4.2. We provide detailed radar plots in Figure B.10 with axes labeled with relevant values for each attribute. The attributes *Point-of-view* and *Environment* are defined qualitatively based on the contents of the target dataset. Examples of videos from each of the datasets are shown in Figure B.11. We can see that FineGym [197] consists of videos of Olympic gymnastic events. Thus, we label it as *stadium* for environment and *third-person* for point-of-view. On the radar plots, we order environment in descending order of variability contained in a given dataset. Kinetics-400 is placed near the origin as it has much higher variability than NTU-60 for example, which is captured in a controlled lab setting. *Action length* is the average duration of the actions in each of the datasets.

We quantify *temporal awareness* as the minimum number of frames (temporal context) required to best recognize the action. We do this by finetuning R(2+1)D with weights initialized from supervised pre-training on Kinetics-400 and we denote temporal awareness (τ) as:

$$\tau = \arg \min_{t \in \{1,2,\ldots,N\}} \left[\left(100 \times \frac{f_{t+1} - f_t}{f_t} \right) < \alpha \right] \tag{B.1}$$

where α is chosen to be 1. This means τ indicates the number of frames after which

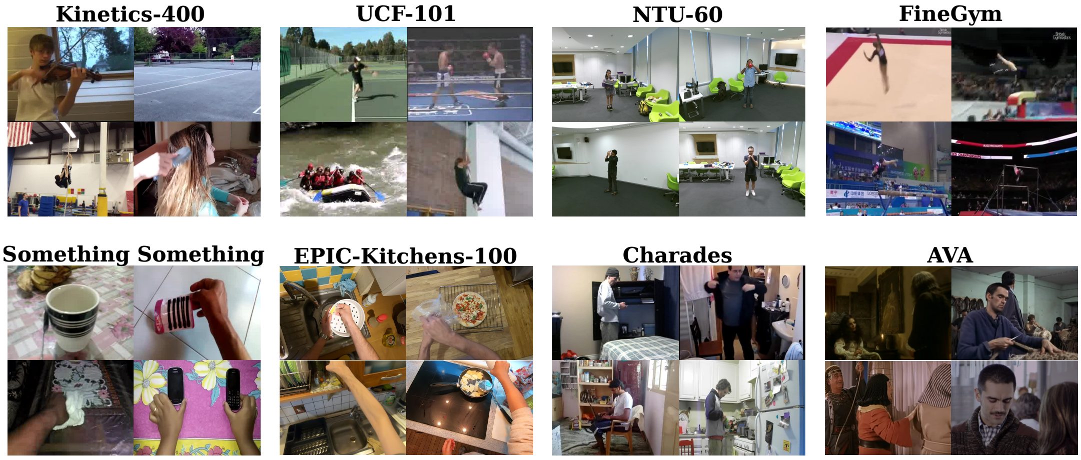


FIGURE B.11: Example video frames from the Kinetics-400 pre-training dataset and the 7 different downstream datasets we consider. Note the differences in the capture setting and point-of-view across these datasets.

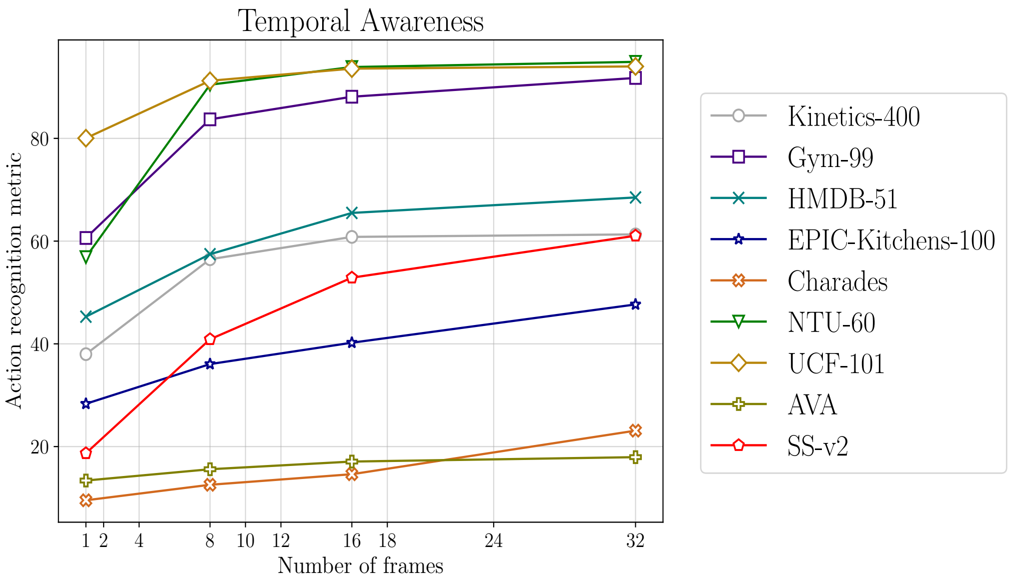


FIGURE B.12: **Temporal awareness**. Illustrating the effect of temporal awareness (increasing temporal-context) on the action recognition performance using a standard 3D-CNN for different action datasets.

relative improvement in performance is lesser than α, *i.e.* when the performance has plateaued. Figure B.12 shows the top-1 action recognition performance against increasing number of frames for each of our downstream datasets. We use bilinear interpolation to estimate performance at given number of frames beyond those that we experiments with. For example, using our method to compute temporal awareness, the performance for UCF-101 plateaus at 7 frames while that for EK-100 plateaus at 32 frames indicating that EK-100 needs much larger temporal context for recognition while UCF-101 may suffice with a shorter temporal context.

Label overlap is the amount of actions which are present in both the downstream dataset and the pretraining dataset (Kinetics-400). We quantify this by matching identical actions as well as manually checking for reworded versions of the same action class. For example, "head massage" in UCF-101 has a corresponding action "massaging person's head" in Kinetics-400. In NTU-60 action class "brushing teeth" has a matching action "brushing teeth" in Kinetics-400.

B.6 Standard deviations for SEVERE-benchmark

In this section, we show the standard deviations of each pretrained method for all downstream setups in our proposed benchmark. The results are obtained over 3 runs initialized with different random seeds. It is clear from Table B.3 that results are consistent over multiple runs with small *std* deviations. Thus our observations and conclusions are not impacted across multiple runs. Moreover, future works can refer to Table B.3 for reproduciblity.

	Existing		**SEVERE-benchmark**							
Pre-training			Domains		Samples		Actions		Tasks	
	UCF101	HMDB51	SS-v2	Gym-99	UCF (10^3)	Gym-99 (10^3)	FX-S1	UB-S1	UCF-RC	Charades-MLC
None	77.3±0.9	47.7±1.6	57.1±1.3	89.8±0.1	38.3±1.4	22.7±3.5	46.6±1.8	82.3±2.1	0.217±0.01	7.9±0.1
MoCo	83.3±0.3	53.6±0.2	57.1±0.1	90.7±0.2	60.4±1.0	30.9±1.0	65.0±1.2	84.5±0.4	0.208±0.01	8.3±0.1
VideoMoCo	84.9±0.5	58.0±1.0	59.0±0.1	90.3±0.3	65.4±1.2	20.6±0.8	57.3±2.9	83.9±1.6	0.185±0.00	10.5±0.1
SeLaVi	85.2±0.3	54.2±0.3	56.2±0.1	88.9±0.1	69.0±1.9	30.2±0.9	51.3±1.0	80.9±1.6	0.162±0.01	8.4±0.1
Pretext-Contrast	87.7±0.6	58.4±0.6	56.9±0.2	90.5±0.1	64.6±2.3	27.5±1.6	66.1±0.3	86.1±0.8	0.164±0.01	8.9±0.1
RSPNet	88.7±0.1	59.2±0.7	59.0±0.3	91.1±0.0	74.7±0.6	32.2±1.5	65.4±1.7	83.6±1.3	0.145±0.01	9.0±0.3
AVID-CMA	88.8±0.3	58.7±1.2	52.0±0.6	90.4±0.4	68.2±0.5	33.4±0.8	68.0±0.9	87.3±1.0	0.148±0.01	8.2±0.2
CtP	90.1±0.1	63.2±0.5	59.6±0.4	92.0±0.1	61.0±1.5	32.9±1.9	79.1±0.5	88.8±0.5	0.178±0.01	9.6±0.1
TCLR	90.8±0.2	60.6±0.9	59.8±0.0	91.6±0.0	72.6±1.9	26.3±1.0	60.7±0.7	84.7±1.1	0.142±0.01	12.2±0.3
GDT	91.3±0.3	64.8±1.0	58.0±0.3	90.5±0.1	78.4±0.2	45.6±0.6	66.0±0.3	83.4±1.6	0.123±0.01	8.5±0.1
Supervised	93.9±0.2	68.5±0.4	60.8±0.1	92.1±0.1	86.6±0.6	51.3±0.1	79.0±2.0	87.1±0.2	0.132±0.01	23.5±0.1

TABLE B.3: **Standard deviations for proposed SEVERE-benchmark**. We compute the *std* of each method for each downstream setup over 3 runs initialized with random seeds.

B.7 HMDB51 Results

For completion we also show the performance of each pretraining method on the HMDB51 [122] dataset in Table B.3. HMDB51 is used in current standard benchmarking along with UCF101 [209]. It is clear from the table that the performances on both datasets are highly correlated to each other and less correlated to other downstream setups. This again highlights the importance of evaluating video self-supervised methods beyond current benchmarks and use a setup which evaluates the generalizability of current models, such as the SEVERE-benchmark.

B.8 Linear Evaluation for Downstream Samples

In Section 4.4 we evaluated our pre-trained models with varying amounts of downstream samples for finetuning. In this section we provide the results for the same experiment but using linear-evaluation instead of finetuning. The results are shown in Figure B.13. We observe that rank changes are not significant between different sample sizes as they are for full finetuning., However similar to finetuning, supervised pretraining is dominant for low data setting as shown by the performance on NTU-60 and GYM-99 with 1000-4000 examples.

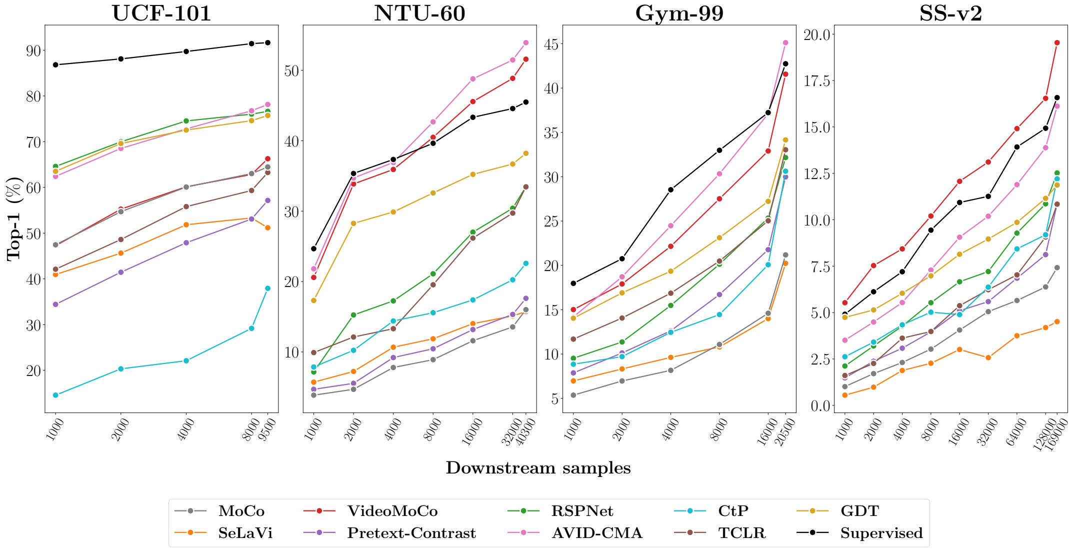


FIGURE B.13: **Linear evaluation for Downstream Samples.** Comparison of video self-supervised learning methods using varying number of samples on linear evaluation for four downstream datasets. Rank changes are less significant with increasing sample size.

B.9 Verb vs. Noun in Downstream Action Recognition

EPIC-Kitchens-100 [39] consists of noun and verb annotations for each video. An action is defined as a tuple of these. In the main paper, we report verb recognition performance across all experiments. In Table B.4 we compare the performance on verb recognition to the performance on noun and action recognition. In general, performance is lower for noun and action recognition in comparison to verb recognition.

	EPIC-Kitchens-100		
Pre-training	Verb	Noun	Action
None	25.7	6.9	1.8
MoCo	26.4	13.9	6.9
SeLaVi	33.8	12.1	5.9
VideoMoCo	43.6	15.1	9.4
Pretext-contrast	34.3	11.4	5.6
RSPNet	42.7	18.7	11.7
AVID-CMA	29.9	8.7	3.6
CtP	42.8	12.0	7.8
TCLR	36.2	11.7	5.8
GDT	37.3	15.5	8.4
Supervised	47.7	24.5	16.0

TABLE B.4: **Ablation on Verb and Noun Recognition.** On EPIC-Kitchens-100, we show results for noun, verb and action recognition. Colors denote relative rankings across methods for each dataset, ranging from low high. Most pre-training methods struggle on noun and action recognition and have little correlation with verb recognition.

This is likely due to the R(2+1)D-18 backbone being insufficient to model the complex actions found in EPIC-Kitchens-100. Interestingly, good performance on verb recognition is not a good indication that the model will perform well at noun or action recognition. Notably, some methods such as VideoMoCo and CtP perform well at verb recognition but struggle on noun recognition. RSPNet seems to perform reasonably well for both verb and noun recognition.

Appendix C

Supplementary Materials for Tubelet-Contrastive Learning

C.1 Generalization on Diving48

To further highlight the generalizability of our method to new domains and fine-grained actions, we finetune and evaluate with the challenging Diving48 dataset [133]. It contains 18K trimmed videos for 48 different diving sequences all of which take place in similar backgrounds and need to be distinguished by subtle differences such as the number of somersaults or the starting position. We use standard train/test split and report top-1 accuracy.

In Table C.1, we show the performance of our model when pretrained on the full Kinetics-400 and on Mini-Kinetics (†). We compare these results to no pretraining, the temporal contrastive baseline pretrained on Kinetics-400, and supervised pretraining on Kinetics-400 with labels. Our method increases the performance over training from scratch by 7.9% and the temporal contrastive baseline by 6.6%. Our method even outperforms the supervised pretraining baseline by 4.5%. This suggests that by contrasting tubelets with different motions, our method is able to learn better video representations for fine-grained actions than supervised pretraining on Kinetics. When pretraining on Mini-Kinetics (3x smaller than Kinetics-400) the performance of our model does not decrease, again demonstrating the data efficiency of our approach.

C.2 Evaluation with R3D and I3D Backbones

In addition to the R(2+1)-18 backbone, we also show the performance of our proposed method with other commonly used video encoders *i.e.*, R3D-18 [229] and I3D [23]. For R3D-18, we use the same tubelet generation and transformation as that of R(2+1)D-18, as described in chapter 5. For I3D, we change the input resolution to 224x224 and sample the patch size $H^{'} \times W^{'}$ uniformly from $[32\times32, 128\times128]$. For both, we follow the same pretraining protocol as described in chapter 5.

We compare with prior works on the standard UCF101 [209] and HMDB51 [122] datasets. Table C.2 shows the results with Kinetics-400 as the pretraining dataset. With the I3D backbone, our method outperforms prior works on both UCF101 and

Pretraining	Top-1
Supervised [229]	84.5
None	81.1
Temporal Contrast Baseline	82.4
This chapter†	**89.4**
This chapter	**89.0**

TABLE C.1: **Generalization on Diving48 [133].** Comparison with temporal contrastive pretraining and supervised pretraining on Diving48 with R(2+1)D-18 backbone. † indicates pretraining on Mini-Kinetics, otherwise all pretraining was done on Kinetics-400.

HMDB51. Similarly, with the R3D-18 backbone, we outperform prior works using the RGB modality on UCF101. We also achieve comparable performance to the best-performing method on HMDB51, improving over the next best method by 6.3%. On HMDB51 we also outperform prior works which pretrain on an additional optical flow modality and achieve competitive results with these methods on UCF101.

C.3 Evaluation on Kinetics Dataset

To show whether our tubelet-contrastive pretraining can improve the performance of downstream tasks when plenty of labeled data is available for finetuning, we evaluate it on the Kinetics-400 [113] dataset for the task of action classification. The dataset contains about 220K labeled videos for training and 18K videos for validation. As evident from Table C.3, such large-scale datasets can still benefit from our pretraining with a 3.4% improvement over training from scratch and 0.7% over the temporal contrast baseline.

C.4 Finetuning Details

During finetuning, we follow the setup from the SEVERE benchmark [224] which is detailed here for completeness. For all tasks, we replace the projection of the pretrained model with a task-dependent head.

Action Recognition. Downstream settings which examine domain shift, sample efficiency, and action granularity all perform action recognition. We use a similar finetuning process for all experiments on these three factors. During the training process, a random clip of 32 frames is taken from each video and standard augmentations are applied: a multi-scale crop of 112x112 size, horizontal flipping, and color jittering. The Adam optimizer is used for training, with the learning rate, scheduling, and total number of epochs for each experiment shown in Table C.4. During inference, 10 linearly spaced clips of 32 frames each are used, with a center crop of 112x112. To

Method	**Modality**	UCF	HMDB
I3D			
SpeedNet [15]	RGB	66.7	43.7
DSM [242]	RGB	74.8	52.5
BE [244]	RGB	86.2	55.4
FAME [45]	RGB	88.6	61.1
***This chapter*†**	RGB	**89.5**	**64.0**
This chapter	RGB	**89.7**	**63.9**
R3D-18			
VideoMoCo [169]	RGB	74.1	43.6
RSPNet [173]	RGB	74.3	41.6
LSFD [14]	RGB	77.2	53.7
MLFO [180]	RGB	79.1	47.6
ASCNet [94]	RGB	80.5	52.3
MCN [138]	RGB	85.4	54.8
TCLR [41]	RGB	85.4	55.4
CtP [237]	RGB	86.2	57.0
TE [101]	RGB	87.1	**63.6**
MSCL [164]	RGB+Flow	90.7	62.3
MaCLR [252]	RGB+Flow	91.3	62.1
***This chapter*†**	RGB	**88.8**	62.0
This chapter	RGB	**90.1**	63.3

TABLE C.2: **Evaluation with I3D and R3D backbones:** on standard UCF101 and HMDB51 benchmarks. Gray lines indicate the use of additional modalities during self-supervised pretraining. † indicates pretraining on Mini-Kinetics, otherwise, all models were pretrained on Kinetics-400.

Pretraining	Top-1
None	61.4
Temporal Contrast Baseline	64.1
This chapter	**64.8**

TABLE C.3: **Kinetics-400 Evaluation.** Comparison with temporal contrastive pretraining for large-scale action recognition. All models use R(2+1)D-18 and pretraining was done on Kinetics-400 training set.

determine the action class prediction for a video, the predictions from each clip are averaged. For domain shift and sample efficiency, we report the top-1 accuracy. For action granularity experiments we report mean class accuracy, which we obtain by computing accuracy per action class and averaging over all action classes.

Evaluation Factor	**Experiment**	**Dataset**	**Batch Size**	**Learning rate**	**Epochs**	**Steps**
Standard	UCF101	UCF 101 [209]	32	0.0001	160	[60,100,140]
	HMDB51	HMDB 51 [122]	32	0.0001	160	[60,100,140]
Domain Shift	SS-v2	Something-Something [74]	32	0.0001	45	[25, 35, 40]
	Gym-99	FineGym [197]	32	0.0001	160	[60,100,140]
Sample Efficiency	UCF (10^3)	UCF 101 [209]	32	0.0001	160	[80,120,140]
	Gym (10^3)	FineGym [197]	32	0.0001	160	[80,120,140]
Action Granularity	FX-S1	FineGym [197]	32	0.0001	160	[70,120,140]
	UB-S1	FineGym [197]	32	0.0001	160	[70,120,140]
Task Shift	UCF-RC	UCFRep [276]	32	0.00005	100	-
	Charades	Charades [202]	16	0.0375	57	[41,49]

TABLE C.4: **Training Details** of finetuning on various downstream datasets and tasks.

Repetition counting. The implementation follows the original repetition counting work proposed in UCFrep work [276]. From the annotated videos, 2M sequences of 32 frames with spatial size 112x112 are constructed. These are used as the input. The model is trained with a batch size of 32 for 100 epochs using the Adam optimizer with a learning rate of 0.00005. For testing, we report mean counting error following[276].
Multi-label classification on Charades. Following [60], a per-class sigmoid output is utilized for multi-class prediction. During the training process, 32 frames are sampled with a stride of 8. Frames are cropped to 112x112 and random short-side scaling, random spatial crop, and horizontal flip augmentations are applied. The model is trained for a total of 57 epochs with a batch size of 16 and a learning rate of 0.0375. A multi-step scheduler with $\gamma = 0.1$ is applied at epochs [41, 49]. During the testing phase, spatiotemporal max-pooling is performed over 10 clips for a single video. We report mean average precision (mAP) across all classes.
SSv2-Sub details. We use a subset of Something-Something v2 for ablations. In particular, we randomly sample 25% of the data from the whole train set and spanning all categories. This results in a subset consisting of 34409 training samples from 174 classes. We use the full validation set of Something-Something v2 for testing.

C.5 Tubelet Transformation Hyperparameters

Table C.5 shows the results when applying multiple tubelet transformations in the tubelet generation. While applying individual transformations improves results, combing multiple transformations doesn't improve the performance further. This is likely because rotation motions are common in the downstream datasets while scaling and shearing are less common.

Table C.6 shows an ablation over Min and Max values for tubelet transformations. In chapter 5., we use scale values between 0.5 and 1.5, shear values between -1.0 and 1.0, and rotation values between -90 and 90. Here, we experiment with values that result in more subtle and extreme variations of these transformations. We observe that all values for each of the transformations improve over no transformation. Our model is reasonably robust to these choices in hyperparameters, but subtle variations *e.g.*,

Transformation	UCF (10^3)	Gym (10^3)
None	63.0	45.6
Scale	65.1	46.5
Shear	65.2	47.5
Rotate	65.5	48.0
Scale + Shear	65.2	46.0
Rotate + Scale	65.4	46.9
Rotate + Shear	65.3	45.7
Rotate + Scale + Shear	65.6	46.0

TABLE C.5: **Tubelet Transformation Combinations.** Combining transformations doesn't give a further increase in performance compared to using individual transformations.

Min	Max	UCF (10^3)	Gym (10^3)
None			
-	-	63.0	45.6
Scale			
0.5	1.25	65.6	45.3
0.5	1.5	65.1	46.5
0.5	2.0	65.6	46.0
Shear			
-0.75	0.75	64.4	47.5
-1.0	1.0	65.2	48.0
-1.5	1.5	65.2	47.5
Rotation			
-45	45	65.2	49.3
-90	90	65.5	48.0
-180	180	65.6	49.6

TABLE C.6: **Tubelet Transformation Hyperparameters.** We change Min and Max values for tubelet transformations. Our model is robust to changes in these parameters, with all choices tested giving an improvement over no tubelet transformation.

scale change between 0.5 to 1.25 or shear from 0.75 to 0.75 tend to be slightly less effective.

C.6 Tubelets vs. Randomly Scaled Crops

To show that our proposed tubelets inject useful motions in the training pipeline, we compare them with randomly scaled crops. In particular, we randomly crop, scale,

	UCF (10^3)	Gym (10^3)	SSv2-Sub	UB-S1
Randomly Scaled Crops	59.5	37.5	44.8	87.0
Tubelets	65.5	48.0	47.9	90.9

TABLE C.7: **Tubelets vs Randomly Scaled Crops**. Our tubelets generate smooth motions to learn better video representations than strongly jittered crops.

and jitter the patches pasted into the video clips when generating positive pairs and pretrain this and our model on Mini-Kinetics. Table C.7 shows that our proposed motion tubelets outperform such randomly scaled crops in all downstream settings. This validates that the spatiotemporal continuity in motion tubelets is important to simulate smooth motions for learning better video representations.

C.7 Per-Class Results

Examining the improvement for individual classes gives us some insight into our model. Figure C.1 shows the difference between our approach and the baseline for the 10 classes in UCF (10^3) with the highest increase and decrease in accuracy. Many of the actions that increase in accuracy are motion-focused, *e.g.*, pullups, lunges and jump rope. Other actions are confused by the baseline because of the similar background, *e.g.*, throw discus is confused with hammer throw and apply eye makeup is confused with haircut. The motion-focused features our model introduces help distinguish these classes. However, our model does lose some useful spatial features for distinguishing classes such as band marching and biking.

C.8 Class Agnostic Activation Maps

Figure C.2 show more examples of class agnostic activation maps [10] for video clips from various downstream datasets. Note that no finetuning is performed, we directly apply the representation from our tubelet contrastive learning pretrained on Kinetics-400. For examples from FineGym, Something Something v2, and UCF101, we observe that our approach attends to regions with motion while the temporal contrastive baseline mostly attends to background.

C.9 Limitations and Future Work

There are several open avenues for future work based on the limitations of this work. First, while we compare to transformer-based approaches, we do not present the results of our tubelet-contrast with a transformer backbone. Our initial experiments with a transformer-based encoder [48] did not converge with off-the-shelf settings.

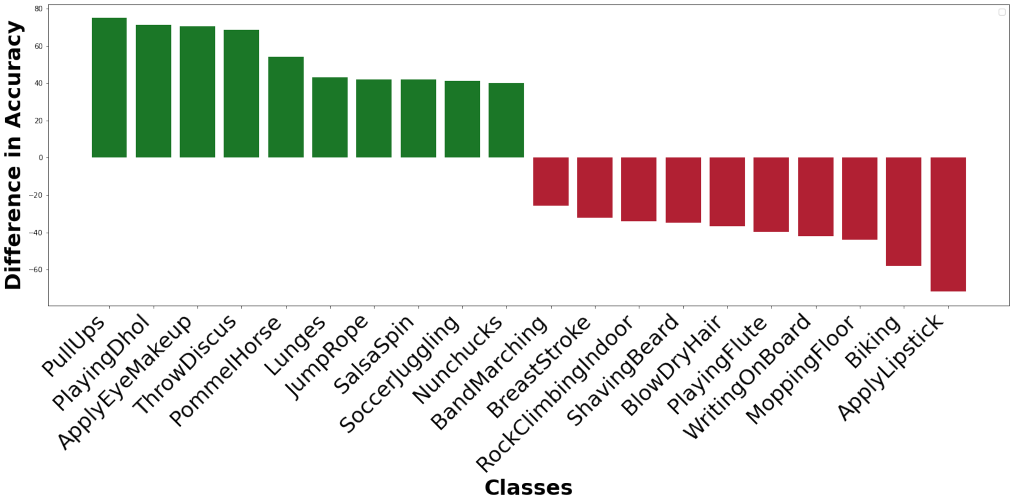


FIGURE C.1: **Per-Class Accuracy Difference** on UCF (10^3) between our model and the temporal contrastive baseline. We show the 10 actions with the highest increase and decrease. Our model can better distinguish classes requiring motion but loses some ability to distinguish spatial classes.

We hope future work can address this problem for an encoder-independent solution. Additionally, we simulate tubelets with random image crops that can come from both background and foreground regions. Explicitly generating tubelets from foreground regions or pre-defined objects is a potential future direction worth investigating. Finally, we only simulate tubelets over short clips, it is also worth investigating whether long-range tubelets can be used for tasks that require long-range motion understanding.

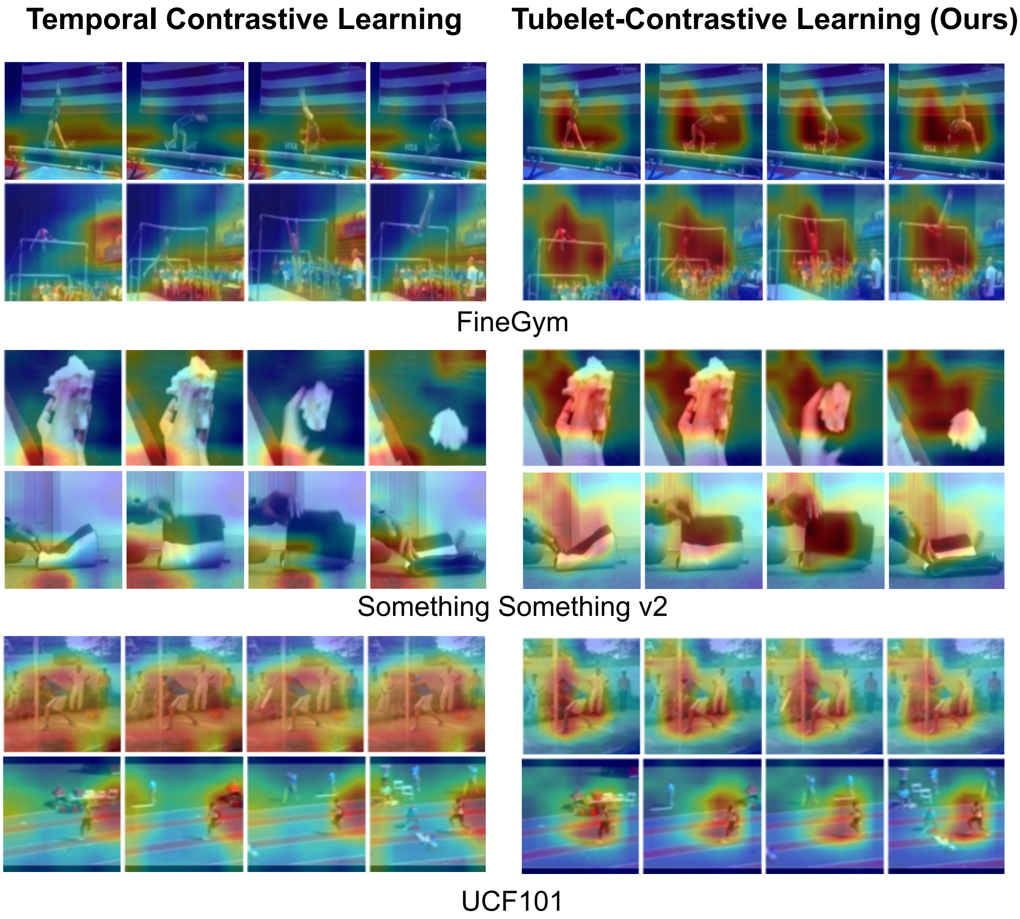

Temporal Contrastive Learning
Tubelet-Contrastive Learning (Ours)
FineGym
Something Something v2
UCF101

FIGURE C.2: **Class-Agnostic Activation Maps Without Finetuning** for the temporal contrastive baseline and our tubelet contrast for different downstream datasets. Our model better attends to regions with motion irrespective of the domain.

Summary and Conclusions

Summary

The thesis strives to endow video-efficiency in video understanding by addressing the research question "*What enables video-efficient video foundation models?*" Video-efficiency encompasses developing video foundation models that are not only accurate but also exhibit label-efficiency *i.e.* require fewer labels, domain-efficiency *i.e.* applicable to a variety of video learning scenarios, and data-efficiency *i.e.* reduce the amount of video data needed for learning. The main research question is addressed for RGB and non-RGB video modalities. A brief summary of each chapter is provided as follows:

In **Chapter** 2, we focus on improving the label- and domain-efficiency of non-RGB action recognition and detection. While there are abundant labeled large-scale action datasets available for RGB video modality, which are extensively used to enhance the performance of new RGB actions, such datasets are scarce for non-RGB modalities like Depth maps and 3D-Skeleton data. To address this, we propose to train action models for a non-RGB target modality, such as depth maps or 3D-skeletons, by extracting knowledge from a large-scale action labeled RGB dataset. Our approach employs a cross-modal teacher-student framework, utilizing unlabeled pairs of RGB and the target modality to transfer action representation knowledge through feature-supervision. The experimental evaluation demonstrates the effectiveness of our approach in improving both label and domain-efficiency for action recognition and detection when utilizing Depth maps and 3D-Skeleton sequences. These findings emphasize the potential of large-scale RGB action datasets in enhancing the video-efficiency of non-RGB video models.

Chapter 3 introduces a new self-supervised approach for learning feature representations for 3D-skeleton video sequences. Existing self-supervised methods for 3D-skeletons often rely on pretext tasks such as motion reconstruction or prediction, which can yield sub-optimal feature representations. To overcome this limitation, we draw inspiration from contrastive learning in the RGB domain and develop a new self-supervised task. Our approach incorporates skeleton-specific augmentations that can capture spatio-temporal dynamics of the skeleton data by generating meaningful positive pairs. Furthermore, we propose inter-skeleton contrast to learn from different input skeleton representations by maximizing the similarities between them. Such formulation avoids any shortcut solutions to the contrastive task and results in learning a better feature space. Experimental results demonstrate that our method outperforms state-of-the-art self-supervised approaches for skeleton data on downstream tasks of action recognition and retrieval. Furthermore, our approach exhibits enhanced label-efficiency compared to the previous self-supervised methods, when only a few labeled samples are available for downstream tasks. These findings underscore the effectiveness of our contrastive self-supervised approach in learning powerful representations for 3D-skeleton video sequences.

In **Chapter** 4, we conduct a large-scale study of existing RGB-based self-supervised video models to assess their performance across different facets of video-efficiency. Existing benchmarks in video self-supervised learning exhibit a high similarity with the datasets used in self-supervised training. This leaves a gap in understanding the generalization capability of video foundation models learned by existing self-supervised tasks beyond such canonical settings. To this end, we evaluate a set of video self-supervised models on a range of downstream setups that encompass variability in environmental conditions, amount of available labeled samples, action granularity, and nature of the task. Our study shows that current benchmarks in video self-supervised learning are not a good indicator of the generalizability across diverse and challenging downstream setups. We observe that video representations learned by vanilla supervised pre-training generalize better than self-supervised representations for most downstream factors. From our experimental analysis, we propose the SEVERE-benchmark that can give some indication of the generalizability of video self-supervised methods across the evaluated downstream factors. Our study demonstrates the lack of label- and domain-efficiency exhibited by the existing video self-supervised foundation models, especially for domains that require finer motion understanding.

Chapter 5 presents a new method for video self-supervision that explicitly aims to learn motion focused video-representations via contrastive learning. Existing works in video contrastive learning aim to increase feature similarity between positive pairs from the same video, resulting in video representations that are skewed toward spatial semantics. In contrast, our method aims to increase feature similarity between video pairs that only share spatio-temporal dynamics in the form of synthetic tubelets. To this end, we simulate synthetic motion tubelets and overlay them on two different video clips to generate positive pairs that have a low spatial bias. Such formulation forces the model to rely on the spatio-temporal dynamics of the tubelets to learn the similarity. Moreover, different tubelet generation and tubelet transformation strategies are proposed to simulate motion patterns beyond what exists in the original training data. Our approach improves the generalizability of the learned video foundation model demonstrating better domain-efficiency, especially to downstream setups from diverse environments and with different action granularities. We also achieve better label-efficiency than prior works for fine-grained action recognition. Furthermore, the experimental evaluation also shows that our method exhibits data efficiency in self-supervised training, retaining its performance when only using 25% of the training data.

Conclusions

This thesis presents several novel approaches to improve the video-efficiency of video foundation models. By demonstrating the effectiveness of video models that use a reduced set of labeled videos for downstream tasks, learn representations from limited amounts of unlabeled data, and adapt to various downstream video domains, we have addressed the main research question for different modalities of video data. Our research highlights the importance of transferring knowledge between RGB and non-RGB video modalities [223], exploring self-supervision for non-RGB video modeling [221], analyzing self-supervised models beyond canonical setups [224] and carefully designing new self-supervised tasks to develop video foundation models that can exhibit all facets of video-efficiency [220]. Our results suggest that video-efficient learning has the potential to train video foundation models that significantly reduce the amount of labeled data required for solving downstream video-based tasks, reduce the computation costs otherwise associated with training video foundation models, and remove the need to build individual domain-specific foundation models for diverse video domains. As a result, it becomes easier to develop video understanding solutions with reduced costs.

However, further research is required to explore the complete potential of video-efficient foundation models and video understanding in general. One intriguing direction involves incorporating additional aspects into video-efficiency, such as ego-centric vision [75], video-text modeling [258], and more complex video tasks like video summarization [6], video segmentation [179], video captioning [273], etc. Another potential direction is to decrease the complexity of video modeling by incorporating temporal redundancy and designing efficient network architectures [57], enabling the viability of video applications with low resources. Moreover, with stricter data regulations from government bodies and the rise in misuse of AI by bad actors, developing privacy-preserving solutions [42, 251] for video understanding is also an important future direction. Similarly, generative AI [231, 88] could be explored to synthesize more realistic video training data for developing new video foundation models. In conclusion, this thesis provides a contribution to video understanding, demonstrating the potential of video-efficient learning and providing insights into how it can be achieved for different modalities of video data. We hope that our work will inspire further research and development in this area, leading to even more efficient and responsible solutions in the future.

Bibliography

[1] Triantafyllos Afouras et al. "Self-Supervised Learning of Audio-Visual Objects from Video". In: *ECCV*. 2020.

[2] Unaiza Ahsan, Rishi Madhok, and Irfan Essa. "Video jigsaw: Unsupervised learning of spatiotemporal context for video action recognition". In: *WACV*. 2019.

[3] S. Albanie et al. "Emotion Recognition in Speech using Cross-Modal Transfer in the Wild". In: *ACM Multimedia*. 2018.

[4] Humam Alwassel et al. "Self-Supervised Learning by Cross-Modal Audio-Video Clustering". In: *NeurIPS*. 2020.

[5] Humam Alwassel et al. "Self-supervised learning by cross-modal audio-video clustering". In: *NeurIPS*. 2020.

[6] Evlampios Apostolidis et al. "Video summarization using deep neural networks: A survey". In: *Proceedings of the IEEE* 109.11 (2021), pp. 1838–1863.

[7] Yuki M Asano, Christian Rupprecht, and Andrea Vedaldi. "A critical analysis of self-supervision, or what we can learn from a single image". In: *ICLR*. 2020.

[8] Yuki M. Asano et al. "Labelling unlabelled videos from scratch with multi-modal self-supervision". In: *NeurIPS*. 2020.

[9] Mahmoud Assran et al. "Masked siamese networks for label-efficient learning". In: *ECCV*. 2022.

[10] Kyungjune Baek, Minhyun Lee, and Hyunjung Shim. "Psynet: Self-supervised approach to object localization using point symmetric transformation". In: *AAAI*. 2020.

[11] Yutong Bai et al. "Can temporal information help with contrastive self-supervised learning?" In: *arXiv*. 2020.

[12] Zhongxin Bai and Xiao-Lei Zhang. "Speaker recognition based on deep learning: An overview". In: *Neural Networks* 140 (2021), pp. 65–99.

[13] Davide Baltieri, Roberto Vezzani, and Rita Cucchiara. "3dpes: 3d people dataset for surveillance and forensics". In: *Proceedings of the 2011 joint ACM workshop on Human gesture and behavior understanding*. 2011.

[14] Nadine Behrmann et al. “Long short view feature decomposition via contrastive video representation learning”. In: *ICCV*. 2021.

[15] Sagie Benaim et al. “Speednet: Learning the speediness in videos”. In: *CVPR*. 2020.

[16] Yoshua Bengio, Réjean Ducharme, and Pascal Vincent. “A neural probabilistic language model”. In: *Advances in neural information processing systems* 13 (2000).

[17] Rishi Bommasani et al. “On the opportunities and risks of foundation models”. In: *arXiv* (2021).

[18] Paulo Vinicius Koerich Borges, Nicola Conci, and Andrea Cavallaro. “Video-based human behavior understanding: A survey”. In: *IEEE transactions on circuits and systems for video technology* 23.11 (2013), pp. 1993–2008.

[19] Barry Brown, Mathias Broth, and Erik Vinkhuyzen. “The Halting problem: Video analysis of self-driving cars in traffic”. In: *Proceedings of Conference on Human Factors in Computing Systems* (2023), pp. 1–14.

[20] Tom Brown et al. “Language models are few-shot learners”. In: *NeurIPS* (2020).

[21] Thomas Brox and Jitendra Malik. “Object segmentation by long term analysis of point trajectories”. In: *ECCV*. 2010.

[22] Liangliang Cao, Zicheng Liu, and Thomas S Huang. “Cross-dataset action detection”. In: *IEEE Computer Society Conference on Computer Vision and Pattern Recognition*. 2010.

[23] Joao Carreira and Andrew Zisserman. “Quo vadis, action recognition? a new model and the kinetics dataset”. In: *CVPR*. 2017.

[24] João Carreira et al. “A Short Note about Kinetics-600”. In: *arXiv* (2018).

[25] João Carreira et al. “A Short Note on the Kinetics-700 Human Action Dataset”. In: *arXiv* (2019).

[26] Brian Chen et al. “Multimodal clustering networks for self-supervised learning from unlabeled videos”. In: *CVPR*. 2021.

[27] Guobin Chen et al. “Learning Efficient Object Detection Models with Knowledge Distillation”. In: *NIPS*. 2017.

[28] Shuo Chen et al. “Social Fabric: Tubelet Compositions for Video Relation Detection”. In: *ICCV*. 2021.

[29] Ting Chen et al. “A Simple Framework for Contrastive Learning of Visual Representations”. In: *ICML*. 2020.

[30] Xinlei Chen et al. “Improved baselines with momentum contrastive learning”. In: *arXiv* (2020).

[31] Feng Cheng and Gedas Bertasius. “TallFormer: Temporal Action Localization with a Long-Memory Transformer”. In: *ECCV*. 2022.

[32] Ke Cheng et al. “Skeleton-Based Action Recognition with Shift Graph Convolutional Network”. In: *CVPR*. 2020.

[33] Hyeon Cho et al. “Self-Supervised Visual Learning by Variable Playback Speeds Prediction of a Video”. In: *IEEE Access* 9 (2021), pp. 79562–79571.

[34] Liu Chunhui et al. “PKU-MMD: A Large Scale Benchmark for Continuous Multi-Modal Human Action Understanding”. In: *ACM Multimedia workshop* (2017).

[35] Ondřej Cífka, Umut Şimşekli, and Gaël Richard. “Groove2groove: One-shot music style transfer with supervision from synthetic data”. In: *IEEE/ACM Transactions on Audio, Speech, and Language Processing* 28 (2020), pp. 2638–2650.

[36] Elijah Cole et al. “When does contrastive visual representation learning work?” In: *CVPR*. 2022.

[37] Emanuele Colleoni, Philip Edwards, and Danail Stoyanov. “Synthetic and real inputs for tool segmentation in robotic surgery”. In: *MICCAI*. 2020.

[38] Nieves Crasto et al. “MARS: Motion-Augmented RGB Stream for Action Recognition”. In: *CVPR*. 2019.

[39] Dima Damen et al. “Rescaling egocentric vision: Collection, pipeline and challenges for epic-kitchens-100”. In: *International Journal of Computer Vision* (2022), pp. 1–23.

[40] Nhan Cach Dang, María N Moreno-García, and Fernando De la Prieta. “Sentiment analysis based on deep learning: A comparative study”. In: *Electronics* 9.3 (2020), p. 483.

[41] Ishan Dave et al. “Tclr: Temporal contrastive learning for video representation”. In: *Computer Vision and Image Understanding* 219 (2022), p. 103406.

[42] Ishan Rajendrakumar Dave, Chen Chen, and Mubarak Shah. “SPAct: Self-Supervised Privacy Preservation for Action Recognition”. In: *CVPR*. 2022.

[43] Jacob Devlin et al. “Bert: Pre-training of deep bidirectional transformers for language understanding”. In: *arXiv* (2018).

[44] Ali Diba et al. “Vi2clr: Video and image for visual contrastive learning of representation”. In: *ICCV*. 2021.

[45] Shuangrui Ding et al. “Motion-Aware Contrastive Video Representation Learning via Foreground-Background Merging”. In: *CVPR*. 2022.

[46] Carl Doersch, Abhinav Gupta, and Alexei A. Efros. “Unsupervised Visual Representation Learning by Context Prediction”. In: *ICCV*. 2015.

[47] Hao-Wen Dong et al. “Multitrack Music Transformer”. In: *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. 2023, pp. 1–5.

[48] Alexey Dosovitskiy et al. “An image is worth 16x16 words: Transformers for image recognition at scale”. In: *arXiv* (2020).

[49] Hazel Doughty and Cees G M Snoek. “How do you do it? fine-grained action understanding with pseudo-adverbs”. In: *CVPR*. 2022.

[50] Hazel Doughty et al. “Action modifiers: Learning from adverbs in instructional videos”. In: *CVPR*. 2020.

[51] Yong Du, Yun Fu, and Liang Wang. “Skeleton based action recognition with convolutional neural network”. In: *ACPR*. 2015.

[52] Haodong Duan et al. “Revisiting skeleton-based action recognition”. In: *CVPR*. 2022.

[53] Linus Ericsson, Henry Gouk, and Timothy M Hospedales. “How well do self-supervised models transfer?” In: *CVPR*. 2021.

[54] Linus Ericsson, Henry Gouk, and Timothy M Hospedales. “Why Do Self-Supervised Models Transfer? Investigating the Impact of Invariance on Downstream Tasks”. In: *BMVC*. 2022.

[55] Eyrun Eyjolfsdottir et al. “Detecting social actions of fruit flies”. In: *ECCV*. 2014.

[56] Bernard Ghanem Fabian Caba Heilbron Victor Escorcia and Juan Carlos Niebles. “ActivityNet: A Large-Scale Video Benchmark for Human Activity Understanding”. In: *CVPR*. 2015.

[57] Christoph Feichtenhofer. “X3d: Expanding architectures for efficient video recognition”. In: *CVPR*. 2020.

[58] Christoph Feichtenhofer, Axel Pinz, and Andrew Zisserman. “Convolutional two-stream network fusion for video action recognition”. In: *CVPR*. 2016.

[59] Christoph Feichtenhofer, Axel Pinz, and Andrew Zisserman. “Detect to track and track to detect”. In: *ICCV*. 2017.

[60] Christoph Feichtenhofer et al. “A large-scale study on unsupervised spatiotemporal representation learning”. In: *CVPR*. 2021.

[61] Christoph Feichtenhofer et al. “Masked autoencoders as spatiotemporal learners”. In: *arXiv* (2022).

[62] Christoph Feichtenhofer et al. “SlowFast Networks for Video Recognition”. In: *ICCV*. 2019.

[63] Christoph Feichtenhofer et al. “Slowfast networks for video recognition”. In: *ICCV*. 2019.

[64] Basura Fernando et al. “Self-supervised video representation learning with odd-one-out networks”. In: *CVPR*. 2017.

[65] Nuno C Garcia, Pietro Morerio, and Vittorio Murino. “Learning with privileged information via adversarial discriminative modality distillation”. In: *IEEE transactions on pattern analysis and machine intelligence* 42.10 (2019), pp. 2581–2593.

[66] Nuno C. Garcia, Pietro Morerio, and Vittorio Murino. “Modality Distillation with Multiple Stream Networks for Action Recognition”. In: *ECCV*. 2018.

[67] Nuno C Garcia et al. “DMCL: Distillation Multiple Choice Learning for Multimodal Action Recognition”. In: *arXiv* (2019).

[68] Kirill Gavrilyuk et al. “Motion-Augmented Self-Training for Video Recognition at Smaller Scale”. In: *ICCV*. 2021.

[69] Kirill Gavrilyuk et al. “Motion-Augmented Self-Training for Video Recognition at Smaller Scale”. In: *ICCV*. 2021.

[70] Jan van Gemert et al. “APT: Action localization proposals from dense trajectories”. In: *BMVC*. 2015.

[71] Amir Ghodrati, Efstratios Gavves, and Cees GM Snoek. “Video Time: Properties, Encoders and Evaluation”. In: *BMVC*. 2018.

[72] Spyros Gidaris, Praveer Singh, and Nikos Komodakis. “Unsupervised Representation Learning by Predicting Image Rotations”. In: *ICLR*. 2018.

[73] Priya Goyal et al. “Scaling and benchmarking self-supervised visual representation learning”. In: *ICCV*. 2019.

[74] Raghav Goyal et al. “The “something something” video database for learning and evaluating visual common sense”. In: *ICCV*. 2017.

[75] Kristen Grauman et al. “Ego4D: Around the World in 3,000 Hours of Egocentric Video”. In: *CVPR*. 2022.

[76] Jean-Bastien Grill et al. “Bootstrap your own latent-a new approach to self-supervised learning”. In: *NeurIPS*. 2020.

[77] Chunhui Gu et al. “AVA: A Video Dataset of Spatio-Temporally Localized Atomic Visual Actions”. In: *CVPR*. 2018.

[78] Saurabh Gupta, Judy Hoffman, and Jitendra Malik. “Cross Modal Distillation for Supervision Transfer”. In: *CVPR*. 2016.

[79] Tengda Han, Weidi Xie, and Andrew Zisserman. “Memory-augmented Dense Predictive Coding for Video Representation Learning”. In: *ECCV*. 2020.

[80] Tengda Han, Weidi Xie, and Andrew Zisserman. “Self-supervised Co-training for Video Representation Learning”. In: *NeurIPS*. 2020.

[81] Tengda Han, Weidi Xie, and Andrew Zisserman. “Video representation learning by dense predictive coding”. In: *ICCVW*. 2019.

[82] Kensho Hara, Hirokatsu Kataoka, and Yutaka Satoh. “Can Spatiotemporal 3D CNNs Retrace the History of 2D CNNs and ImageNet?” In: *CVPR*. 2018.

[83] Kensho Hara, Hirokatsu Kataoka, and Yutaka Satoh. “Can Spatiotemporal 3D CNNs Retrace the History of 2D CNNs and ImageNet?” In: *CVPR*. 2018.

[84] Kaiming He et al. “Masked autoencoders are scalable vision learners”. In: *CVPR*. 2022.

[85] Kaiming He et al. “Momentum contrast for unsupervised visual representation learning”. In: *CVPR*. 2020.

[86] Alejandro Hernandez Ruiz et al. “3D CNNs on Distance Matrices for Human Action Recognition”. In: *ACM Multimedia*. 2017.

[87] Geoffrey Hinton, Oriol Vinyals, and Jeffrey Dean. “Distilling the Knowledge in a Neural Network”. In: *NeurIPS Workshop*. 2015.

[88] Jonathan Ho et al. “Imagen video: High definition video generation with diffusion models”. In: *arXiv* (2022).

[89] Judy Hoffman, Saurabh Gupta, and Trevor Darrell. “Learning with side information through modality hallucination”. In: *CVPR*. 2016.

[90] James Hong et al. “Video pose distillation for few-shot, fine-grained sports action recognition”. In: *ICCV*. 2021.

[91] Rui Hou, Chen Chen, and Mubarak Shah. “Tube convolutional neural network (T-CNN) for action detection in videos”. In: *ICCV*. 2017.

[92] Di Hu, Feiping Nie, and Xuelong Li. “Deep multimodal clustering for unsupervised audiovisual learning”. In: *CVPR*. 2019.

[93] Huazhang Hu et al. “TransRAC: Encoding Multi-scale Temporal Correlation with Transformers for Repetitive Action Counting”. In: *CVPR*. 2022.

[94] Deng Huang et al. “Ascnet: Self-supervised video representation learning with appearance-speed consistency”. In: *ICCV*. 2021.

[95] Zhen Huang et al. “Spatio-Temporal Inception Graph Convolutional Networks for Skeleton-Based Action Recognition”. In: *ACM Multimedia*. 2020.

[96] Ziyuan Huang et al. “Self-supervised motion learning from static images”. In: *CVPR*. 2021.

[97] Yuqi Huo et al. “Self-Supervised Video Representation Learning with Constrained Spatiotemporal Jigsaw”. In: *Proceedings of the Thirtieth International Joint Conference on Artificial Intelligence, IJCAI-21* (2021), pp. 751–757.

[98] Haroon R Idrees et al. “The THUMOS challenge on action recognition for videos ”in the wild””. In: *Computer Vision and Image Understanding* (2016).

[99] Ashraful Islam et al. “A broad study on the transferability of visual representations with contrastive learning”. In: *ICCV*. 2021.

[100] Mihir Jain et al. "Action localization with tubelets from motion". In: *CVPR*. 2014.

[101] Simon Jenni and Hailin Jin. "Time-equivariant contrastive video representation learning". In: *ICCV*. 2021.

[102] Simon Jenni, Givi Meishvili, and Paolo Favaro. "Video Representation Learning by Recognizing Temporal Transformations". In: *ECCv*. 2020.

[103] Shuiwang Ji et al. "3D Convolutional Neural Networks for Human Action Recognition." In: *IEEE Transactions on Pattern Analysis and Machine Intelligence* (2013).

[104] Shulei Ji, Xinyu Yang, and Jing Luo. "A Survey on Deep Learning for Symbolic Music Generation: Representations, Algorithms, Evaluations, and Challenges". In: *ACM Comput. Surv.* (2023).

[105] Chao Jia et al. "Scaling up visual and vision-language representation learning with noisy text supervision". In: *ICML*. 2021.

[106] Longlong Jing et al. "Self-Supervised Spatiotemporal Feature Learning via Video Rotation Prediction". In: *arXiv* (2018).

[107] Vicky Kalogeiton et al. "Action tubelet detector for spatio-temporal action localization". In: *ICCV*. 2017.

[108] Kai Kang et al. "Object detection in videos with tubelet proposal networks". In: *ICCV*. 2017.

[109] Kai Kang et al. "T-cnn: Tubelets with convolutional neural networks for object detection from videos". In: *IEEE Transactions on Circuits and Systems for Video Technology* (2017).

[110] Andrej Karpathy et al. "Large-scale video classification with convolutional neural networks". In: *CVPR*. 2014.

[111] Andrej Karpathy et al. "Large-scale Video Classification with Convolutional Neural Networks". In: *CVPR*. 2014.

[112] Hirokatsu Kataoka et al. "Would mega-scale datasets further enhance spatiotemporal 3D CNNs?" In: *arXiv* (2020).

[113] Will Kay et al. "The Kinetics Human Action Video Dataset". In: *arXiv* (2017).

[114] Dahun Kim, Donghyeon Cho, and In So Kweon. "Self-supervised video representation learning with space-time cubic puzzles". In: *AAAI*. 2019.

[115] Dahun Kim, Donghyeon Cho, and In So Kweon. "Self-supervised video representation learning with space-time cubic puzzles". In: *AAAI*. 2019.

[116] Manjin Kim et al. "Relational self-attention: What's missing in attention for video understanding". In: *NeurIPS*. 2021.

[117] Alexander Kolesnikov, Xiaohua Zhai, and Lucas Beyer. "Revisiting self-supervised visual representation learning". In: *CVPR*. 2019.

[118] Okan Köpüklü, Xiangyu Wei, and Gerhard Rigoll. “You Only Watch Once: A Unified CNN Architecture for Real-Time Spatiotemporal Action Localization”. In: *arXiv* (2019).

[119] Bruno Korbar, Du Tran, and Lorenzo Torresani. “Cooperative learning of audio and video models from self-supervised synchronization”. In: *NeurIPS*. 2018.

[120] Klemen Kotar et al. “Contrasting contrastive self-supervised representation learning pipelines”. In: *ICCV*. 2021.

[121] Hildegard Kuehne et al. “HMDB: a large video database for human motion recognition”. In: *ICCV*. 2011.

[122] H. Kuhne et al. “HMDB: A Large Video Database for Human Motion Recognition”. In: *ICCV*. 2011.

[123] Heeseung Kwon et al. “Learning self-similarity in space and time as generalized motion for video action recognition”. In: *ICCV*. 2021.

[124] Zihang Lai, Erika Lu, and Weidi Xie. “MAST: A Memory-Augmented Self-Supervised Tracker”. In: *CVPR*. 2020.

[125] Hsin-Ying Lee et al. “Unsupervised representation learning by sorting sequences”. In: *ICCV*. 2017.

[126] Shi Lei et al. “Skeleton-Based Action Recognition With Directed Graph Neural Networks”. In: *CVPR*. 2019.

[127] Ofir Levy and Lior Wolf. “Live repetition counting”. In: *ICCV*. 2015.

[128] Chao Li et al. “Co-occurrence Feature Learning from Skeleton Data for Action Recognition and Detection with Hierarchical Aggregation”. In: *Proceedings of the Twenty-Seventh International Joint Conference on Artificial Intelligence, IJCAI-18* (2018), pp. 786–792.

[129] Maosen Li et al. “Actional-Structural Graph Convolutional Networks for Skeleton-Based Action Recognition”. In: *CVPR*. 2019.

[130] Tianhong Li et al. “Making the Invisible Visible: Action Recognition Through Walls and Occlusions”. In: *ICCV*. 2019.

[131] Tianjiao Li et al. “Dynamic Spatio-Temporal Specialization Learning for Fine-Grained Action Recognition”. In: *ECCV*. 2022.

[132] Yingwei Li, Yi Li, and Nuno Vasconcelos. “Resound: Towards action recognition without representation bias”. In: *ECCV*. 2018.

[133] Yingwei Li, Yi Li, and Nuno Vasconcelos. “Resound: Towards action recognition without representation bias”. In: *ECCV*. 2018.

[134] Yixuan Li et al. “Actions as moving points”. In: *ECCV*. 2020.

[135] Ji Lin, Chuang Gan, and Song Han. “Tsm: Temporal shift module for efficient video understanding”. In: *ICCV*. 2019.

[136] Lilang Lin et al. "MS2L: Multi-Task Self-Supervised Learning for Skeleton Based Action Recognition". In: *ACM Multimedia*. 2020.

[137] Yuanze Lin, Xun Guo, and Yan Lu. "Self-supervised video representation learning with meta-contrastive network". In: *ICCV*. 2021.

[138] Yuanze Lin, Xun Guo, and Yan Lu. "Self-supervised video representation learning with meta-contrastive network". In: *ICCV*. 2021.

[139] Zhang Lin et al. "Inter-intra Variant Dual Representations for Self-supervised Video Recognition". In: *BMVC*. 2021.

[140] Jun Liu et al. "Global context-aware attention lstm networks for 3d action recognition". In: *CVPR*. 2017.

[141] Jun Liu et al. "NTU RGB+D 120: A Large-Scale Benchmark for 3D Human Activity Understanding". In: *IEEE Trans. Pattern Anal. Mach. Intell.* 42.10 (2020), pp. 2684–2701.

[142] Jun Liu et al. "Spatio-Temporal LSTM with Trust Gates for 3D Human Action Recognition". In: *ECCV*. 2016.

[143] Mengyuan Liu, Hong Liu, and Chen Chen. "Enhanced skeleton visualization for view invariant human action recognition". In: *Pattern Recognition* (2017).

[144] Zhi Liu, Chenyang Zhang, and Yingli Tian. "3D-based Deep Convolutional Neural Network for action recognition with depth sequences". In: *Image Vis. Comput.* 55 (2016), pp. 93–100.

[145] Ziyu Liu et al. "Disentangling and Unifying Graph Convolutions for Skeleton-Based Action Recognition". In: *CVPR*. 2020.

[146] Ilya Loshchilov and Frank Hutter. "Sgdr: Stochastic gradient descent with warm restarts". In: *ICLR*. 2017.

[147] Dezhao Luo et al. "Video cloze procedure for self-supervised spatio-temporal learning". In: *AAAI*. 2020.

[148] Fengjun Lv and Ramakant Nevatia. "Recognition and Segmentation of 3-d Human Action Using HMM and Multi-Class Adaboost". In: *ECCV*. 2006.

[149] Shuang Ma et al. "Active Contrastive Learning of Audio-Visual Video Representations". In: *ICLR*. 2021.

[150] Khoi-Nguyen C Mac et al. "Learning motion in feature space: Locally-consistent deformable convolution networks for fine-grained action detection". In: *ICCV*. 2019.

[151] Iacopo Masi et al. "Deep face recognition: A survey". In: *conference on graphics, patterns and images (SIBGRAPI)*. 2018.

[152] Effrosyni Mavroudi et al. "End-to-end fine-grained action segmentation and recognition using conditional random field models and discriminative sparse coding". In: *WACV*. 2018.

[153] Pascal Mettes, Jan C van Gemert, and Cees G M Snoek. "Spot on: Action localization from pointly-supervised proposals". In: *ECCV*. 2016.

[154] Shervin Minaee et al. "Image segmentation using deep learning: A survey". In: *IEEE Transactions on Pattern Analysis and Machine Intelligence* 44.7 (2021), pp. 3523–3542.

[155] Riccardo Miotto et al. "Deep learning for healthcare: review, opportunities and challenges". In: *Briefings in bioinformatics* 19.6 (2018), pp. 1236–1246.

[156] Ishan Misra and Laurens van der Maaten. "Self-Supervised Learning of Pretext-Invariant Representations". In: *CVPR*. 2020.

[157] Ishan Misra, C Lawrence Zitnick, and Martial Hebert. "Shuffle and learn: unsupervised learning using temporal order verification". In: *ECCV*. 2016.

[158] Pedro Morgado, Nuno Vasconcelos, and Ishan Misra. "Audio-visual instance discrimination with cross-modal agreement". In: *CVPR*. 2021.

[159] Artem Moskalev et al. "Contrasting quadratic assignments for set-based representation learning". In: *ECCV*. 2022.

[160] Alejandro Newell and Jia Deng. "How useful is self-supervised pretraining for visual tasks?" In: *CVPR*. 2020.

[161] Xun Long Ng et al. "Animal Kingdom: A Large and Diverse Dataset for Animal Behavior Understanding". In: *CVPR*. 2022.

[162] Thao Nguyen, Maithra Raghu, and Simon Kornblith. "Do Wide and Deep Networks Learn the Same Things? Uncovering How Neural Network Representations Vary with Width and Depth". In: *ICLR*. 2021.

[163] Bingbing Ni, Vignesh R Paramathayalan, and Pierre Moulin. "Multiple granularity analysis for fine-grained action detection". In: *CVPR*. 2014.

[164] Jingcheng Ni et al. "Motion Sensitive Contrastive Learning for Self-supervised Video Representation". In: *ECCV*. 2022.

[165] Qiang Nie, Ziwei Liu, and Yunhui Liu. "Unsupervised 3D Human Pose Representation with Viewpoint and Pose Disentanglement". In: *ECCV*. 2020.

[166] Mehdi Noroozi and Paolo Favaro. "Unsupervised Learning of Visual Represensentions by solving Jigsaw Puzzles". In: *ECCV*. 2016.

[167] Aaron van den Oord, Yazhe Li, and Oriol Vinyals. "Representation learning with contrastive predictive coding". In: *arXiv* (2018).

[168] Myle Ott et al. "Scaling Neural Machine Translation". In: *Proceedings of the Third Conference on Machine Translation (WMT)*. 2018.

[169] Tian Pan et al. "Videomoco: Contrastive video representation learning with temporally adversarial examples". In: *CVPR*. 2021.

[170] Deepak Pathak et al. "Context Encoders: Feature Learning by Inpainting". In: *CVPR*. 2016.

[171] Mandela Patrick et al. “Multi-modal Self-Supervision from Generalized Data Transformations”. In: *ICCV*. 2021.

[172] Malte Pedersen et al. “3d-zef: A 3d zebrafish tracking benchmark dataset”. In: *CVPR*. 2020.

[173] Chen Peihao et al. “RSPNet: Relative Speed Perception for Unsupervised Video Representation Learning”. In: *AAAI*. 2021.

[174] Wei Peng et al. “Mix dimension in poincaré geometry for 3d skeleton-based action recognition”. In: *ACM Multimedia*. 2020.

[175] Federico Perazzi et al. “A benchmark dataset and evaluation methodology for video object segmentation”. In: *CVPR*. 2016.

[176] Karol J Piczak. “ESC: Dataset for environmental sound classification”. In: *ACM Multimedia*. 2015.

[177] AJ Piergiovanni, Anelia Angelova, and Michael S Ryoo. “Evolving losses for unsupervised video representation learning”. In: *CVPR*. 2020.

[178] AJ Piergiovanni and Michael S Ryoo. “Fine-grained activity recognition in baseball videos”. In: *CVPRW*. 2018.

[179] Jordi Pont-Tuset et al. “The 2017 davis challenge on video object segmentation”. In: *arXiv* (2017).

[180] Rui Qian et al. “Enhancing Self-supervised Video Representation Learning via Multi-level Feature Optimization”. In: *ICCV*. 2021.

[181] Rui Qian et al. “Spatiotemporal contrastive video representation learning”. In: *CVPR*. 2021.

[182] Rui Qian et al. “Spatiotemporal contrastive video representation learning”. In: *CVPR*. 2021.

[183] Alec Radford et al. “Improving language understanding by generative pre-training”. In: *OpenAI blog* (2018).

[184] Alec Radford et al. “Language models are unsupervised multitask learners”. In: *OpenAI blog* 1.8 (2019), p. 9.

[185] Alec Radford et al. “Learning transferable visual models from natural language supervision”. In: *ICML*. 2021.

[186] Alec Radford et al. “Robust speech recognition via large-scale weak supervision”. In: *ICML*. 2023.

[187] Aditya Ramesh et al. “Hierarchical text-conditional image generation with clip latents”. In: *arXiv* (2022).

[188] Kanchana Ranasinghe et al. “Self-supervised video transformer”. In: *CVPR*. 2022.

[189] Adria Recasens et al. “Broaden your views for self-supervised video learning”. In: *ICCV*. 2021.

[190] Tom FH Runia, Cees GM Snoek, and Arnold WM Smeulders. “Real-world repetition estimation by div, grad and curl”. In: *CVPR*. 2018.

[191] Mert Bulent Sariyildiz et al. “Concept generalization in visual representation learning”. In: *ICCV*. 2021.

[192] N. Sayed, Biagio Brattoli, and Björn Ommer. “Cross and Learn: Cross-Modal Self-Supervision”. In: *GCPR*. 2018.

[193] Madeline C Schiappa, Yogesh S Rawat, and Mubarak Shah. “Self-supervised learning for videos: A survey”. In: *ACM Computing Surveys* (2022).

[194] Sayed Khushal Shah, Zeenat Tariq, and Yugyung Lee. “Iot based urban noise monitoring in deep learning using historical reports”. In: *IEEE International Conference on Big Data*. 2019.

[195] Amir Shahroudy et al. “Ntu rgb+ d: A large scale dataset for 3d human activity analysis”. In: *CVPR*. 2016.

[196] Amir Shahroudy et al. “NTU RGB+D: A Large Scale Dataset for 3D Human Activity Analysis”. In: *CVPR*. 2016.

[197] Dian Shao et al. “FineGym: A Hierarchical Video Dataset for Fine-grained Action Understanding”. In: *CVPR*. 2020.

[198] Dian Shao et al. “Intra-and inter-action understanding via temporal action parsing”. In: *CVPR*. 2020.

[199] Lei Shi et al. “Two-Stream Adaptive Graph Convolutional Networks for Skeleton-Based Action Recognition”. In: *CVPR*. 2019.

[200] Zheng Shou, Dongang Wang, and Shih-Fu Chang. “Temporal action localization in untrimmed videos via multi-stage cnns”. In: *CVPR*. 2016.

[201] Chenyang Si et al. “Adversarial Self-Supervised Learning for Semi-Supervised 3D Action Recognition”. In: *ECCV*. 2020.

[202] Gunnar A. Sigurdsson et al. “Hollywood in Homes: Crowdsourcing Data Collection for Activity Understanding”. In: *ECCV*. 2016.

[203] Karen Simonyan and Andrew Zisserman. “Two-stream convolutional networks for action recognition in videos”. In: *NeurIPS*. 2014.

[204] Karen Simonyan and Andrew Zisserman. “Two-Stream Convolutional Networks for Action Recognition in Videos”. In: *NeurIPS*. 2014.

[205] Sijie Song et al. “An End-to-End Spatio-Temporal Attention Model for Human Action Recognition from Skeleton Data”. In: *AAAI*. 2017.

[206] Yi-Fan Song et al. “Richly Activated Graph Convolutional Network for Robust Skeleton-based Action Recognition”. In: *IEEE Transactions on Circuits and Systems for Video Technology* (2021). In press.

[207] Yi-Fan Song et al. “Stronger, Faster and More Explainable: A Graph Convolutional Baseline for Skeleton-Based Action Recognition”. In: *ACM Multimedia*. 2020.

[208] Tae Soo Kim and Austin Reiter. “Interpretable 3d human action analysis with temporal convolutional networks”. In: *CVPRW*. 2017.

[209] Khurram Soomro, Amir Roshan Zamir, and Mubarak Shah. “UCF101: A dataset of 101 human action classes from videos in the wild”. In: *CRCV-TR-12-01*. 2012.

[210] Jonathan Stroud et al. “D3D: Distilled 3D Networks for Video Action Recognition”. In: *WACV*. 2020.

[211] Kun Su, Xiulong Liu, and Eli Shlizerman. “Predict & cluster: Unsupervised skeleton based action recognition”. In: *CVPR*. 2020.

[212] Baoli Sun et al. “Fine-grained Action Recognition with Robust Motion Representation Decoupling and Concentration”. In: *ACM Multimedia*. 2022.

[213] Chen Sun et al. “Composable Augmentation Encoding for Video Representation Learning”. In: *ICCV*. 2021.

[214] Jennifer J Sun et al. “The Multi-Agent Behavior Dataset: Mouse Dyadic Social Interactions”. In: *Proceedings of the Neural Information Processing Systems Track on Datasets and Benchmarks*. 2021.

[215] Xinyu Sun et al. “Masked Motion Encoding for Self-Supervised Video Representation Learning”. In: *CVPR*. 2023.

[216] Tomoyuki Suzuki et al. “Learning Spatiotemporal 3D Convolution with Video Order Self-Supervision”. In: *ECCV*. 2018.

[217] Li Tao, Xueting Wang, and Toshihiko Yamasaki. “Pretext-Contrastive Learning: Toward Good Practices in Self-supervised Video Representation Leaning”. In: *arXiv* (2021).

[218] Li Tao, Xueting Wang, and Toshihiko Yamasaki. “Pretext-Contrastive Learning: Toward Good Practices in Self-supervised Video Representation Leaning”. In: *arXiv* (2021).

[219] Li Tao, Xueting Wang, and Toshihiko Yamasaki. “Self-supervised video representation learning using inter-intra contrastive framework”. In: *ACM Multimedia*. 2020.

[220] Fida Mohammad Thoker, Hazel Doughty, and Cees Snoek. “Tubelet-Contrastive Self-Supervision for Video-Efficient Generalization”. In: *ICCV* (2023).

[221] Fida Mohammad Thoker, Hazel Doughty, and Cees GM Snoek. “Skeleton-contrastive 3D action representation learning”. In: *ACM Multimedia*. 2021.

[222] Fida Mohammad Thoker and Juergen Gall. “Cross-modal Knowledge Distillation for Action Recognition”. In: *ICIP*. 2019.

[223] Fida Mohammad Thoker and Cees GM Snoek. “Feature-Supervised Action Modality Transfer”. In: *ICPR*. 2020.

[224] Fida Mohammad Thoker et al. “How Severe Is Benchmark-Sensitivity in Video Self-supervised Learning?” In: *ECCV*. 2022.

[225] Yicong Tian, Rahul Sukthankar, and Mubarak Shah. “Spatiotemporal deformable part models for action detection”. In: *Proceedings of the IEEE conference on computer vision and pattern recognition*. 2013.

[226] Yonglong Tian, Dilip Krishnan, and Phillip Isola. “Contrastive multiview coding”. In: *arXiv preprint arXiv:1906.05849* (2019).

[227] Zhan Tong et al. “Videomae: Masked autoencoders are data-efficient learners for self-supervised video pre-training”. In: *Advances in Neural Information Processing Systems (NeurIPS)*. 2022.

[228] D. Tran et al. “A Closer Look at Spatiotemporal Convolutions for Action Recognition”. In: *CVPR*. 2018.

[229] Du Tran et al. “A closer look at spatiotemporal convolutions for action recognition”. In: *CVPR*. 2018.

[230] Du Tran et al. “Learning Spatiotemporal Features with 3D Convolutional Networks”. In: *ICCV*. 2015.

[231] Sergey Tulyakov et al. “MoCoGAN: Decomposing Motion and Content for Video Generation”. In: *CVPR*. 2018.

[232] Grant Van Horn et al. “Benchmarking representation learning for natural world image collections”. In: *CVPR*. 2021.

[233] Raviteja Vemulapalli, Felipe Arrate, and Rama Chellappa. “Human Action Recognition by Representing 3D Skeletons as Points in a Lie Group”. In: *CVPR*. 2014.

[234] Raviteja Vemulapalli and Rama Chellapa. “Rolling Rotations for Recognizing Human Actions From 3D Skeletal Data”. In: *CVPR*. 2016.

[235] Antonio W. Vieira et al. “STOP: Space-Time Occupancy Patterns for 3D Action Recognition from Depth Map Sequences”. In: *Pattern Recognition, Image Analysis, Computer Vision, and Applications*. 2012.

[236] Bram Wallace and Bharath Hariharan. “Extending and analyzing self-supervised learning across domains”. In: *ECCV*. 2020.

[237] Guangting Wang et al. “Unsupervised Visual Representation Learning by Tracking Patches in Video”. In: *CVPR*. 2021.

[238] Heng Wang et al. “Dense trajectories and motion boundary descriptors for action recognition”. In: *International Journal of Computer Vision* (2013).

[239] Jiangliu Wang, Jianbo Jiao, and Yun-Hui Liu. “Self-supervised video representation learning by pace prediction”. In: *ECCV*. 2020.

[240] Jiangliu Wang, Jianbo Jiao, and Yun-Hui Liu. “Self-supervised video representation learning by pace prediction”. In: *ECCV*. 2020.

[241] Jiangliu Wang et al. “Self-supervised spatio-temporal representation learning for videos by predicting motion and appearance statistics”. In: *CVPR*. 2019.

[242] Jinpeng Wang et al. “Enhancing unsupervised video representation learning by decoupling the scene and the motion”. In: *AAAI*. 2021.

[243] Jinpeng Wang et al. “Removing the Background by Adding the Background: Towards Background Robust Self-supervised Video Representation Learning”. In: *CVPR*. 2021.

[244] Jinpeng Wang et al. “Removing the background by adding the background: Towards background robust self-supervised video representation learning”. In: *CVPR*. 2021.

[245] Limin Wang et al. “Temporal Segment Networks: Towards Good Practices for Deep Action Recognition”. In: *ECCV*. 2016.

[246] Pichao Wang et al. “Deep Convolutional Neural Networks for Action Recognition Using Depth Map Sequences”. In: *arXiv* (2015).

[247] Xin Wang, Yuan-Fang Wang, and William Yang Wang. “Watch, Listen, and Describe: Globally and Locally Aligned Cross-Modal Attentions for Video Captioning”. In: *NAACL*. 2018.

[248] Donglai Wei et al. “Learning and using the arrow of time”. In: *CVPR*. 2018.

[249] Cong Wu, Xiao-Jun Wu, and Josef Kittler. “Spatial Residual Layer and Dense Connection Block Enhanced Spatial Temporal Graph Convolutional Network for Skeleton-Based Action Recognition”. In: *ICCVW*. 2019.

[250] Shih-Lun Wu and Yi-Hsuan Yang. “MuseMorphose: Full-song and fine-grained piano music style transfer with one transformer VAE”. In: *IEEE/ACM Transactions on Audio, Speech, and Language Processing* 31 (2023), pp. 1953–1967.

[251] Zhenyu Wu et al. “Privacy-preserving deep action recognition: An adversarial learning framework and a new dataset”. In: *IEEE Transactions on Pattern Analysis and Machine Intelligence* 44.4 (2020), pp. 2126–2139.

[252] Fanyi Xiao, Joseph Tighe, and Davide Modolo. “MaCLR: Motion-Aware Contrastive Learning of Representations for Videos”. In: *ECCV*. 2022.

[253] Fanyi Xiao, Joseph Tighe, and Davide Modolo. “Modist: Motion distillation for self-supervised video representation learning”. In: *arXiv*. 2021.

[254] Saining Xie et al. “Rethinking spatiotemporal feature learning: Speed-accuracy trade-offs in video classification”. In: *ECCV*. 2018.

[255] Dejing Xu et al. “Self-supervised spatiotemporal learning via video clip order prediction”. In: *CVPR*. 2019.

[256] Huijuan Xu, Abir Das, and Kate Saenko. “R-C3D: region convolutional 3d network for temporal activity detection”. In: *ICCV*. 2017.

[257] Huijuan Xu, Abir Das, and Kate Saenko. “R-C3D: region convolutional 3d network for temporal activity detection”. In: *ICCV*. 2017.

[258] Jun Xu et al. “Msr-vtt: A large video description dataset for bridging video and language”. In: *CVPR*. 2016.

[259] Kai Xu et al. “Spatiotemporal CNN for video object segmentation”. In: *CVPR*. 2019.

[260] Sijie Yan, Yuanjun Xiong, and Dahua Lin. “Spatial Temporal Graph Convolutional Networks for Skeleton-Based Action Recognition”. In: *AAAI*. 2018.

[261] Ceyuan Yang et al. “Temporal pyramid network for action recognition”. In: *CVPR*. 2020.

[262] Ceyuan Yang et al. “Video representation learning with visual tempo consistency”. In: *arXiv* (2020).

[263] Xiaodong Yang, Chenyang Zhang, and Yingli Tian. “Recognizing actions using depth motion maps-based histograms of oriented gradients”. In: *ACM Multimedia*. 2012.

[264] Xingyi Yang et al. “Transfer learning or self-supervised learning? A tale of two pretraining paradigms”. In: *arXiv* (2020).

[265] Xitong Yang et al. “Step: Spatio-temporal progressive learning for video action detection”. In: *CVPR*. 2019.

[266] Ting Yao et al. “Seco: Exploring sequence supervision for unsupervised representation learning”. In: *AAAI*. 2021.

[267] Yuan Yao et al. “Video playback rate perception for self-supervised spatio-temporal representation learning”. In: *CVPR*. 2020.

[268] Yuan Yao et al. “Video Playback Rate Perception for Self-Supervised Spatio-Temporal Representation Learning”. In: *CVPR*. 2020.

[269] Fanfan Ye et al. “Dynamic GCN: Context-enriched Topology Learning for Skeleton-based Action Recognition”. In: *ACM Multimedia*. 2020.

[270] Junsong Yuan, Zicheng Liu, and Ying Wu. “Discriminative Video Pattern Search for Efficient Action Detection”. In: *IEEE Transactions on Pattern Analysis and Machine Intelligence* 33.9 (2011), 1728–1743.

[271] Syed Sahil Abbas Zaidi et al. “A survey of modern deep learning based object detection models”. In: *Digital Signal Processing* 126 (2022), p. 103514.

[272] Xiaohua Zhai et al. “A large-scale study of representation learning with the visual task adaptation benchmark”. In: *arXiv* (2019).

[273] Bowen Zhang, Hexiang Hu, and Fei Sha. “Cross-Modal and Hierarchical Modeling of Video and Text”. In: *ECCV*. 2018.

[274] Chen-Lin Zhang, Jianxin Wu, and Yin Li. “ActionFormer: Localizing Moments Of Actions With Transformers”. In: *ECCV*. 2022.

[275] Chuhan Zhang, Ankush Gupta, and Andrew Zisserman. “Temporal query networks for fine-grained video understanding”. In: *CVPR*. 2021.

[276] Huaidong Zhang et al. “Context-aware and scale-insensitive temporal repetition counting”. In: *CVPR*. 2020.

[277] Huaidong Zhang et al. “Context-Aware and Scale-Insensitive Temporal Repetition Counting”. In: *CVPR*. 2020.

[278] Pengfei Zhang et al. “View adaptive recurrent neural networks for high performance human action recognition from skeleton data”. In: *ICCV*. 2017.

[279] Richard Zhang, Phillip Isola, and Alexei A Efros. “Colorful Image Colorization”. In: *ECCV*. 2016.

[280] Richard Zhang, Phillip Isola, and Alexei A Efros. “Split-Brain Autoencoders: Unsupervised Learning by Cross-Channel Prediction”. In: *CVPR*. 2017.

[281] Yujia Zhang et al. “Contrastive Spatio-Temporal Pretext Learning for Self-supervised Video Representation”. In: *AAAI*. 2022.

[282] Yunhua Zhang, Ling Shao, and Cees G M Snoek. “Repetitive activity counting by sight and sound”. In: *CVPR*. 2021.

[283] Jiaojiao Zhao and Cees G. M. Snoek. “Dance with Flow: Two-in-One Stream Action Detection”. In: *CVPR*. 2019.

[284] Jiaojiao Zhao et al. “Tuber: Tubelet transformer for video action detection”. In: *CVPR*. 2022.

[285] Mingmin Zhao et al. “Through-Wall Human Pose Estimation Using Radio Signals”. In: *CVPR*. 2018.

[286] Nenggan Zheng et al. “Unsupervised Representation Learning with Long-Term Dynamics for Skeleton Based Action Recognition”. In: *AAAI*. 2018.

Samenvatting en Conclusies

Samenvatting

Het proefschrift streeft ernaar om video-efficiëntie bij te brengen in video-begrip door zich te richten op de onderzoeksvraag "Wat maakt video-efficiënte video foundation-modellen mogelijk?". Video-efficiëntie omvat het ontwikkelen van video foundation-modellen die niet alleen nauwkeurig zijn, maar ook label-efficiënt zijn, d.w.z. minder labels vereisen, domein-efficiënt zijn, d.w.z. toepasbaar zijn in verschillende video-leerscenario's, en data-efficiënt zijn, d.w.z. de hoeveelheid video-data die nodig is voor het leren verminderen. De belangrijkste onderzoeksvraag wordt behandeld voor zowel RGB- als niet-RGB-video-modaliteiten. Een beknopte samenvatting van elk hoofdstuk wordt hieronder gegeven:

In **Hoofdstuk** 2 richten we ons op het verbeteren van de label- en domein-efficiëntie van niet-RGB actieherkenning en detectie. Hoewel er in overvloed gelabelde, grootschalige actiedatasets beschikbaar zijn voor RGB-video's, die uitgebreid worden gebruikt om de prestaties van nieuwe RGB-acties te verbeteren, zijn dergelijke datasets schaars voor niet-RGB-modaliteiten zoals dieptekaarten en 3D-skeletgegevens. Om dit aan te pakken, stellen we voor om actiemodellen te trainen voor een niet-RGB-doelmodaliteit, zoals dieptekaarten of 3D-skeletten, door kennis te extraheren uit een grootschalige actie-gelabelde RGB-dataset. Onze aanpak maakt gebruik van een cross-modal student-teacher framework, waarbij ongelabelde paren van RGB en de doelmodaliteit worden gebruikt om actie-representatiekennis over te dragen via feature-supervisie. De experimentele evaluatie toont de effectiviteit van onze aanpak aan bij het verbeteren van zowel de label- als domein-efficiëntie voor actieherkenning en detectie bij het gebruik van dieptekaarten en 3D-skeletsequenties. Deze bevindingen benadrukken het potentieel van grootschalige RGB-actiedatasets bij het verbeteren van de video-efficiëntie van niet-RGB-video-modellen.

In **Hoofdstuk** 3 wordt een nieuwe self-supervised aanpak geïntroduceerd voor het leren van feature-representaties voor 3D-skelet videosequenties. Bestaande self-supervised methoden voor 3D-skeletten bouwen vaak op pretext taken zoals bewegingsreconstructie of voorspelling, wat kan leiden tot suboptimale feature-representaties. Om deze beperking te overkomen, halen we inspiratie uit contrastive learning in het RGB-domein en ontwikkelen we een nieuwe self-supervised taak. Onze aanpak omvat skelet-specifieke augmentaties die de spatio-temporele dynamiek van de skeletgegevens kunnen vastleggen door betekenisvolle positieve paren te genereren. Bovendien stellen we inter-skelet contrast voor om te leren van verschillende input skeletrepresentaties door de overeenkomsten tussen hen te maximaliseren. Een

dergelijke formulering vermijdt shortcut-oplossingen van de contrastieve taak en resulteert in het leren van een betere feature space. Experimentele resultaten tonen aan dat onze methode beter presteert dan state-of-the-art self-supervised methodes voor skeletgegevens op downstream-taken zoals actieherkenning en retrieval. Bovendien vertoont onze aanpak een verbeterde label-efficiëntie in vergelijking met eerdere self-supervised methoden, wanneer slechts een paar gelabelde voorbeelden beschikbaar zijn voor downstream-taken. Deze bevindingen benadrukken de effectiviteit van onze contrastieve self-supervised aanpak bij het leren van krachtige representaties voor 3D-skelet videosequenties.

In **Hoofdstuk** 4 voeren we een grootschalige studie uit naar bestaande op RGB gebaseerde self-supervised videomodellen om hun prestaties te beoordelen op verschillende aspecten van video-efficiëntie. Bestaande benchmarks in self-supervised video leren laten een hoge gelijkenis zien met de datasets die worden gebruikt in self-supervised training. Dit laat een gat achter in het begrijpen van de generalisatiecapaciteit van video foundation-modellen die zijn geleerd door bestaande self-supervised taken buiten dergelijke standaardopstellingen. Om dit te bereiken, evalueren we een set van self-supervised videomodellen op verschillende downstream-opstellingen die variabiliteit omvatten in omgevingsomstandigheden, hoeveelheid beschikbare gelabelde voorbeelden, actiegranulariteit en aard van de taak. Onze studie toont aan dat huidige benchmarks in self-supervised videoleren geen goede indicator zijn van de generaliseerbaarheid over diverse en uitdagende downstream-opstellingen. We constateren dat videorepresentaties geleerd door gewone supervised pre-training beter generaliseren dan self-supervised representaties voor de meeste downstream-factoren. Op basis van onze experimentele analyse stellen we de SEVERE-benchmark voor die enige indicatie kan geven van de generaliseerbaarheid van self-supervised methoden voor video over de geëvalueerde downstream-factoren. Onze studie toont het gebrek aan label- en domein-efficiëntie aan van de bestaande self-supervised video foundation-modellen, vooral voor domeinen die een fijner bewegingsbegrip vereisen.

In **Hoofdstuk** 5 presenteren we een nieuwe methode voor video self-supervision die expliciet gericht is op het leren van op beweging gerichte videorepresentaties via contrastive learning. Bestaande werken op het gebied van video contrastive learning hebben tot doel de feature-overeenkomst tussen positieve paren van dezelfde video te vergroten, wat resulteert in videorepresentaties die naar ruimtelijke semantiek neigen. In tegenstelling hiermee heeft onze methode als doel de feature-overeenkomst te vergroten tussen videoparen die alleen spatio-temporele dynamiek delen in de vorm van synthetische tubelets. Hiertoe simuleren we synthetische bewegings-tubelets en leggen ze over twee verschillende videoclips om positieve paren te genereren met een lage ruimtelijke bias. Een dergelijke formulering dwingt het model om te vertrouwen op de spatio-temporele dynamiek van de tubelets om de overeenkomst te leren. Bovendien worden verschillende strategieën voorgesteld voor het genereren en transformeren van tubelets om bewegingspatronen te simuleren die verder gaan dan wat er in de oorspronkelijke trainingsgegevens bestaat. Onze aanpak verbetert de generaliseerbaarheid van het geleerde video foundation-model en toont een betere domein-efficiëntie, vooral voor downstream-opstellingen in diverse omgevingen en met verschillende actiegranulariteiten. We bereiken ook een betere label-efficiëntie dan eerdere methoden voor fijnkorrelige actieherkenning. Bovendien toont de experimentele evaluatie ook aan dat onze methode data-efficiëntie vertoont in self-supervised training, waarbij de prestaties behouden blijven wanneer slechts 25% van de trainingsgegevens worden gebruikt.

Conclusies

Deze scriptie presenteert verschillende nieuwe methodes om de video-efficiëntie van video foundation-modellen te verbeteren. Door de effectiviteit van video-modellen aan te tonen die een verminderde set gelabelde video's gebruiken voor downstream-taken, representaties leren uit beperkte hoeveelheden ongelabelde data, en zich aanpassen aan verschillende downstream video-domeinen, hebben we de belangrijkste onderzoeksvraag behandeld voor verschillende modaliteiten van videodata. Ons onderzoek benadrukt het belang van het overdragen van kennis tussen RGB- en niet-RGB-videomodaliteiten [223], het verkennen van self-supervision voor niet-RGB-video-modellering [221], het analyseren van self-supervised modellen buiten standaardopstellingen [224], en het zorgvuldig ontwerpen van nieuwe self-supervised taken om video foundation-modellen te ontwikkelen die alle facetten van video-efficiëntie kunnen vertonen [220]. Onze resultaten suggereren dat video-efficiënt leren het potentieel heeft om video foundation-modellen te trainen die aanzienlijk de hoeveelheid gelabelde data verminderen die nodig is voor het oplossen van downstream video-gebaseerde taken, de computationele kosten verminderen die normaal gepaard gaan met het trainen van video foundation-modellen, en de noodzaak wegnemen om individuele domein-specifieke foundation-modellen te bouwen voor diverse video-domeinen. Hierdoor wordt het gemakkelijker om video-begripsoplossingen te ontwikkelen met verminderde kosten.

Niettemin is verder onderzoek nodig om het volledige potentieel van video-efficiënte foundation-modellen en video-begrip in het algemeen te verkennen. Een intrigerende richting omvat het opnemen van aanvullende aspecten in video-efficiëntie, zoals egocentrische visie [75], video-tekst modellering [258], en complexere video-taken zoals video-samenvatting [6], video-segmentatie [179], video-onderschriften [273], enzovoort. Een andere potentiële richting is het verminderen van de complexiteit van video-modellering door het opnemen van temporele redundantie en het ontwerpen van efficiënte netwerkarchitecturen [57], waardoor video-toepassingen met weinig middelen haalbaar worden. Bovendien, met strengere gegevensreguleringen van overheidsinstanties en de toename van het misbruik van AI door kwaadwillende actoren, is het ontwikkelen van privacybehoudende oplossingen [42, 251] voor videobegrip ook een belangrijke toekomstige richting. Op dezelfde manier zou generatieve AI [231, 88] kunnen worden verkend om realistischere video-trainingsgegevens te synthetiseren voor het ontwikkelen van nieuwe video foundation-modellen. Samengevat levert deze scriptie een bijdrage aan video-begrip, waarbij het potentieel van video-efficiënt leren wordt aangetoond en inzichten worden gegeven in hoe dit kan worden bereikt voor verschillende modaliteiten van videodata. We hopen dat ons werk verdere onderzoek en ontwikkeling op dit gebied zal inspireren, wat zal leiden tot nog efficiëntere en verantwoordelijke oplossingen in de toekomst.

Acknowledgments

I would like to take this opportunity to express my heartfelt gratitude to all the people whose support, unwavering belief, and love propelled me to complete this Ph.D. journey.

This journey would not have been possible without the unwavering support, guidance, and encouragement of my supervisor, Cees Snoek. His wealth of knowledge, critical thinking, meticulous attention to detail, and unwavering dedication have proven to be indispensable. I am genuinely thankful for the opportunity to thrive and develop under his mentorship. He has not only guided me from being an aspiring researcher but also transformed me into an independent thinker with a deep-seated passion for science. More significantly, Cees is an extraordinary individual. I cannot stress enough how supportive he has been in times when I needed to focus on my personal commitments. Whether it was embarking on a journey to India for my wedding during the height of the coronavirus pandemic or taking extra time off when my son was born, I never sensed any work-related pressures during these challenging periods. I consider myself extremely fortunate to have him as my supervisor.

I also deeply thank Hazel Doughty who is my co-supervisor. Over the past three years, she has been my go-to person for almost every work-related issue *e.g.* sharing ideas, brainstorming on new research ideas, engaging in discussions, and much more. I've gained valuable soft skills from her, particularly in the areas of presentation and scientific writing, which are crucial for a researcher's growth. On a personal note, Hazel is an exceptional individual who has consistently provided assistance whenever I've required help, whether it's in the form of favors, guidance on career choices, or anything else. I eagerly anticipate further collaboration with her in the future

My Ph.D. journey has greatly benefited from the indispensable assistance of Dennis Koelma. He serves as the cornerstone of the VISLab and consistently provides unwavering support to all Ph.D. students. His expertise in both hardware and software is extensive, and he is ever-willing to assist, sometimes even going above and beyond. Dennis is one of the nicest people that I've ever had the pleasure of meeting, despite making me the most infamous PH.D. student in the IVI slack channel. His readiness to engage in conversations on a wide range of topics has been a source of immense enjoyment for me throughout the years.

Furthermore, I would like to express my gratitude to my former lab mates, including Gjorgji Strezoski, William Thong, Devanshu Arya, Yunlu Chen, Shuo Chen, Mehmet Altinkaya, Noureldien Hussein, Riaan Zoetmulder, Sadaf Gulshad, Zenglin Shi, Tao Hue, Sarah Ibrahimi, David Zhang, Teng Long, Jia-Hong Huang, and Jao Jao Zhao. My Ph.D. journey began with all of you, and I cherished those initial moments, as they made me feel welcome in a new environment and provided me with valuable learning experiences. I also want to extend my thanks to numerous colleagues from VISLab, such as Efstratios Gavves, Iris Groen, Yuki M. Asano, Pascal Mettes, Adeel Pervez, Artem Moskalev, Amber Brands, Clemens Georg Bartnik, Mina Ghadimiatigh, Yunhua Zhang, Sarah Rastegar, Aritra Bhowmik, Mohammad Mahdi Derakhshani, Tejaswi Kasarla, Pengwan Yang, Michael Dorkenwald, Mohammadreza Salehi, Miltiadis Kofinas, Alex Gabel, and many others not listed here. Witnessing your achievements has been a great source of inspiration for me, motivating me to work even harder and become a more proficient researcher. I would like to offer special thanks to Piyush Bagad for his collaboration on the ECCV 2022 paper, and to Max van Spengler who assisted me in translating the summary of this thesis from English to Dutch. I hope that the friendships I have cultivated over these four years will endure and remain meaningful for a lifetime.

Lastly, I want to express my deep gratitude to my family, whose boundless support has been a cornerstone of my success. I'd like to extend my appreciation to my parents and my older brother Irfan for their encouragement in pursuing studies abroad and a Ph.D. I am also thankful to my younger brothers Shahid and Naveed for assuming family responsibilities during my absence. Most significantly, my heartfelt thanks go to my incredible wife, Samiya, for her unwavering support throughout my Ph.D. journey and her patience during the demanding conference deadlines. I am also indebted to Samiya for granting me the most joyful moment of my life, the birth of our son, Ibaad, and for caring for him over the past year and a half. His laughter and playful antics brought a sense of joy and adventure to this demanding journey.