Appendix

In Appendix A, we provide details of the video self-supervised models we use in our evaluation study. Appendix B 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 C. Feature similarities between supervised pre-training and each self-supervised pre-training method are shown in Appendix D. In Appendix E, 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 F and also compare the SEVERE benchmark to results on HMDB51 action recognition in Appendix G. Finally, we report results of some additional experiments in Appendix H and Appendix I that we did not have room for in the main paper.

A 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 [10] 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 [4] views the audio and visual modalities as different augmentations of a video and learns with a cross-modal clustering pretext task.

VideoMoCo [53] 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 [70] combines the pretext task approach with contrastive learning. As its pretext task it uses video cloze procedure [45] 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 [56] 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 [49] 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 [75] 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 [14] 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 [54] 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 5.

Table 5: **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 epoches 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.

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

B Downstream Experimental Details

B.1 Downstream Domain

In Section 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 [65] 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 first standard split proposed in the original paper [65] containing 9,537 training and 3,783 testing samples for the 101 action classes.
NTU-60: [62] 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 [62] to split the data into 40,320 training and 16,560 testing samples for 60 action classes.
Gym-99. We use FineGym version $v1.0$ [63] 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: [25] 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 [25] containing 168,913 training and 24,777 testing samples for 174 action classes.
EPIC-Kitchens-100: [13] 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 [13] 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 I.
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 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 6. 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.

Table 6: **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.

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]

B.2 Downstream Samples

In Section 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.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.1 and Table 6.

B.3 Downstream Actions

In Section 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 [63], 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 [63].

B.4 Downstream Tasks

In Section 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.* [39] 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 [39] 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.* [87], 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 [87] 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 [87] 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.* [23]. 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 [64] 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.* [20][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 [20], 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 [27] 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 [20] 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 [20], 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

C Correlations of Downstream Performance

As observed from the results of Section 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 Figs. 5 to 12. 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.

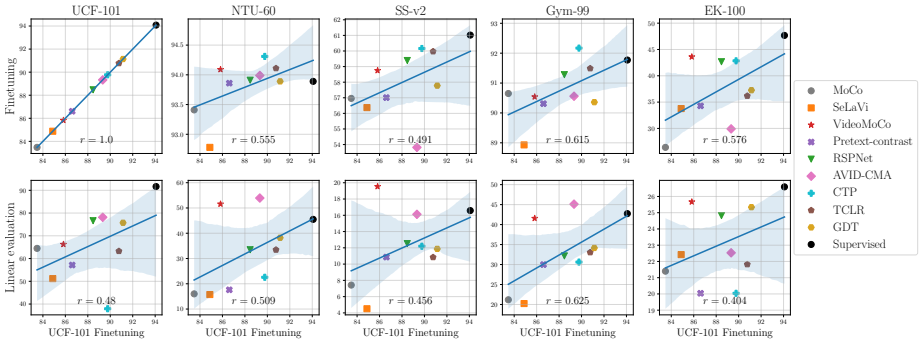

Fig. 5: **Downstream domain against UCF-101 finetuning.** We plot the corelations between finetuning performance of video pre-training methods on UCF-101 and performances on finetuning and linear-evaluation on all downstream datasets.

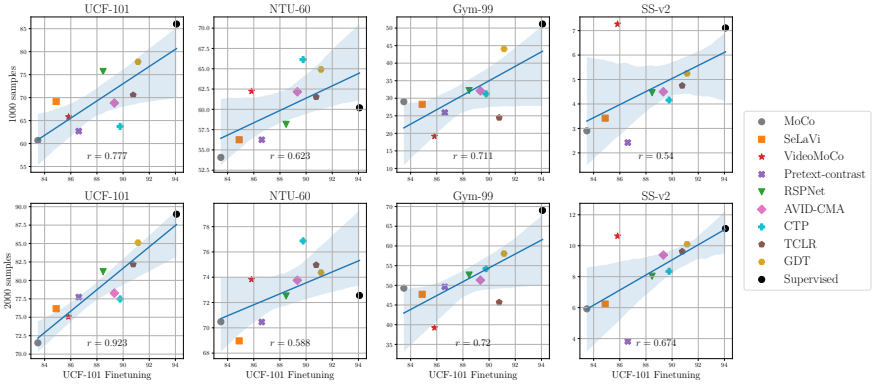

Fig. 6: **Downstream samples against UCF-101 finetuning.** For the low data setting (1000-2000 samples), we plot the correlations of performance of video pre-training methods against that for UCF-101 finetuning.

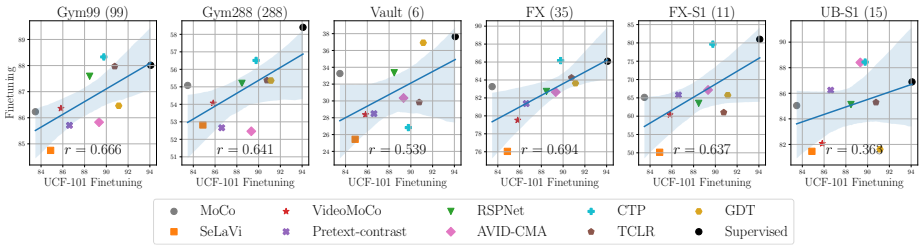

Fig. 7: **Downstream actions against UCF-101 finetuning.** We plot the corelations of performances of video pre-training methods between UCF-101 finetuning and FineGym subsets.

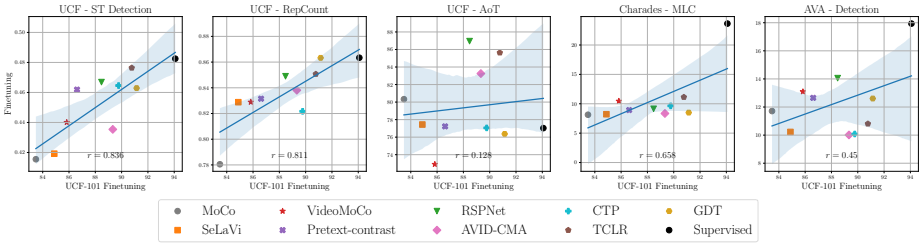

Fig. 8: **Downstream tasks against UCF-101 finetuning.** We plot the corelations between performance on UCF-101 finetuning and other downstream tasks for the video pre-training methods.

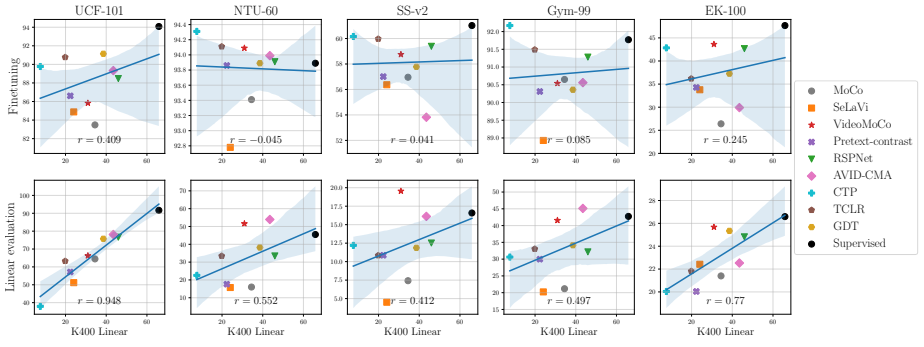

Fig. 9: **Downstream domain against Kinetics-400 linear evaluation.** We plot the corelations between finetuning performance of video pre-training methods on Kinetics-400 linear-evaluation and performances on finetuning and linear-evaluation on all downstream datasets.

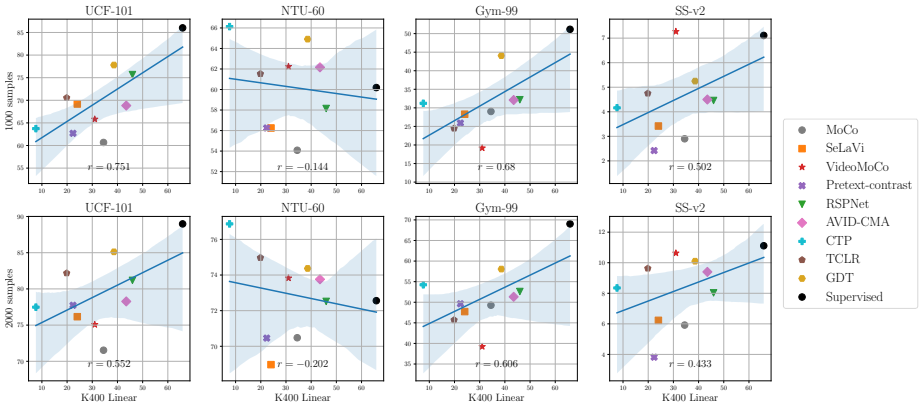

Fig. 10: **Downstream samples against Kinetics-400 linear evaluation.** For the low data setting (1000-2000 samples), we plot the correlations of performance of video pre-training methods against that for Kinetics-400 linear-evaluation.

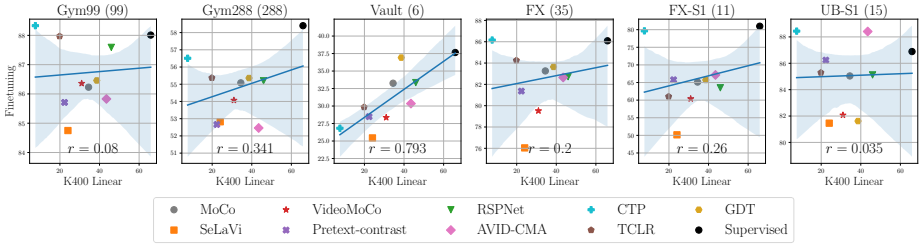

Fig. 11: **Downstream actions against Kinetics-400 linear evaluation.** We plot the corelations of performances of video pre-training methods between Kinetics-400 linear-evaluation and FineGym subsets.

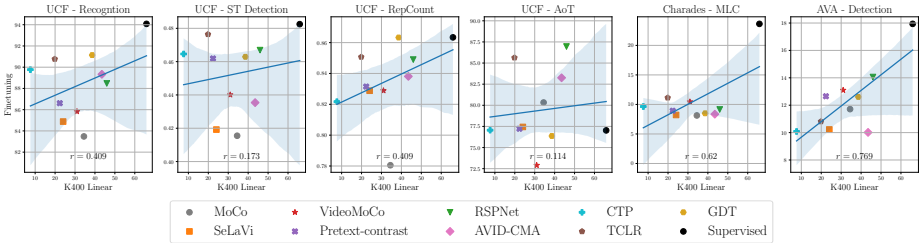

Fig. 12: **Downstream tasks against Kinetics-400 linear evaluation.** We plot the corelations between performance on Kinetics-400 linear-evaluation and other downstream tasks for the video pre-training methods.

D Representation Similarity Matrices

We plot the the feature similarity on Kinetics validation set using centered kernel alignment [52] between supervised pre-training and our evaluated self-supervised pre-training methods in Fig. 13. We showed a subset of these plots in Fig. 4, here we show the feature similarity for all the self-supervised models we used in our experiments.

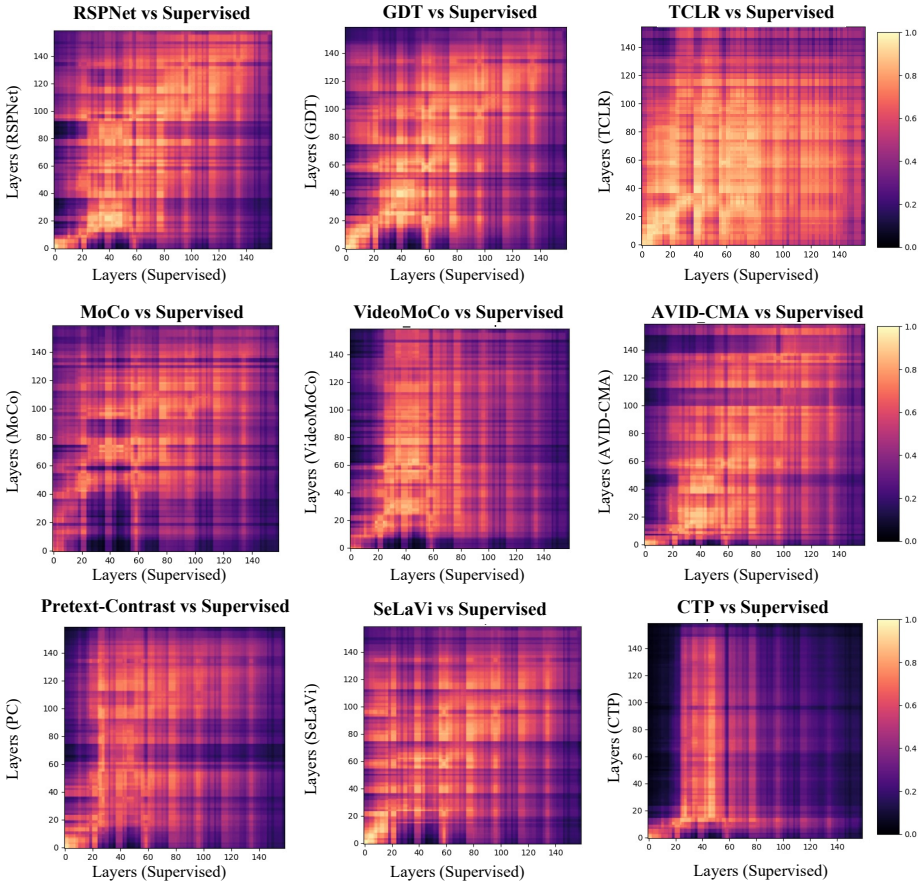

Fig. 13: **Representation similarity** between features of self-supervised methods and supervised pre-training on Kinetics-400 validation set using centered kernel alignment. Features of contrastive methods are more closer to the features of supervised pretraining.

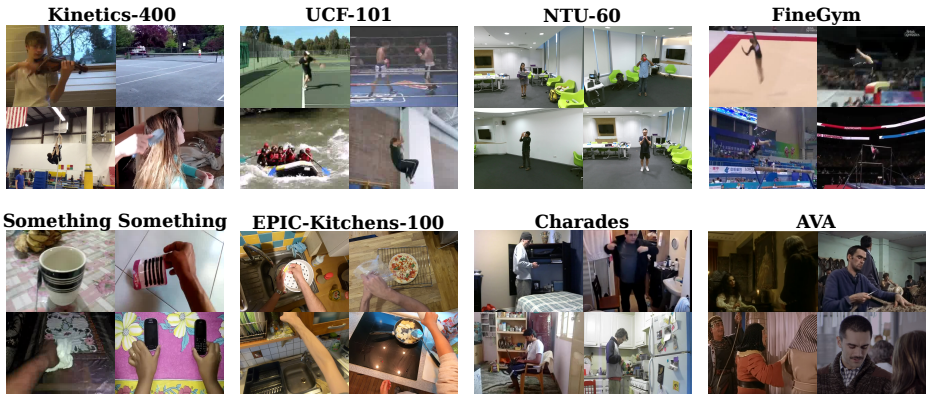

Fig. 14: Example video frames from the Kinetics-400 pre-training dataset and the 7 different downstream datasets we consider. Note the differences in the capture setting and point-of-view across these datasets.

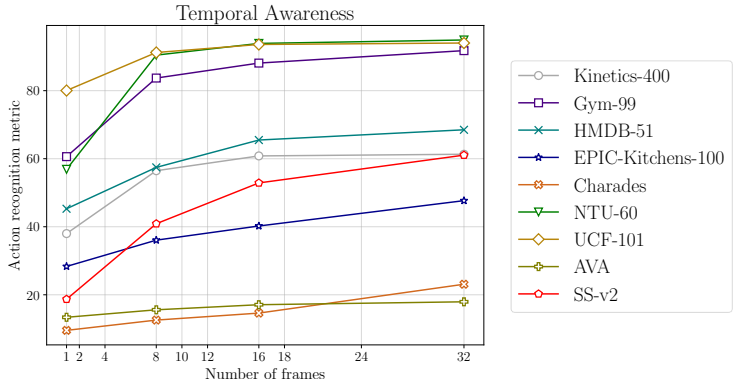

Fig. 15: **Temporal awareness**. Illustrating the effect of temporal awareness (increasing temporal-context) on the action recognition performance using a standard 3D-CNN for different action datasets.

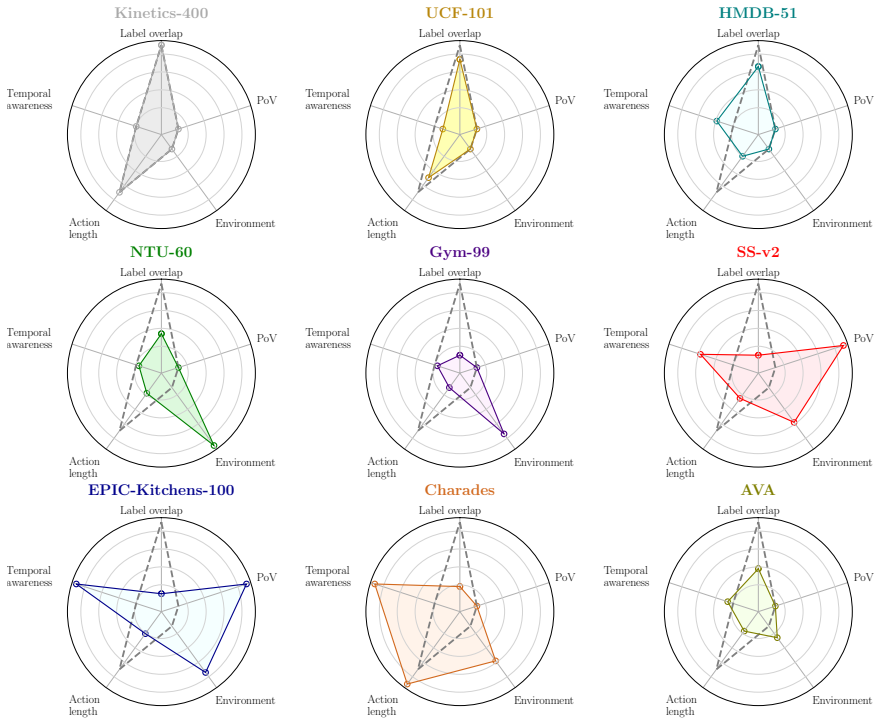

Fig. 16: **Radar plots with details**. The radar plots contain details of the values along the axis for every attribute for the datasets we use in this study.

E Downstream Dataset Attributes

We define several attributes in Section 2.1 in order to characterize differences in domain between the downstream datasets and the Kinetics-400 pre-training dataset in Fig. 2. We provide detailed radar plots in Fig. 16 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 Fig. 14. We can see that FineGym [63] 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:

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

where α is chosen to be 1. This means τ indicates the number of frames after which relative improvement in performance is lesser than α, *i.e.* when the performance has plateaued. Fig. 15 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.

F Standard deviations for proposed 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 7 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 7 for reproduciblity.

Table 7: **Standard deviations for proposed SEVERE-benchmark**. We compute the *std* of each method for each downstream setup over 3 runs initialized with random seeds.

Pre-training	Existing		SEVERE-benchmark							
			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

G HMDB51 Results

For completion we also show the performance of each pretraining method on HMDB51 [42] dataset in Table 7. HMDB51 is used in current standard benchmarking along with UCF101 [65]. 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.

H Linear Evaluation for Downstream Samples

In Section 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 Fig. 17. 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.

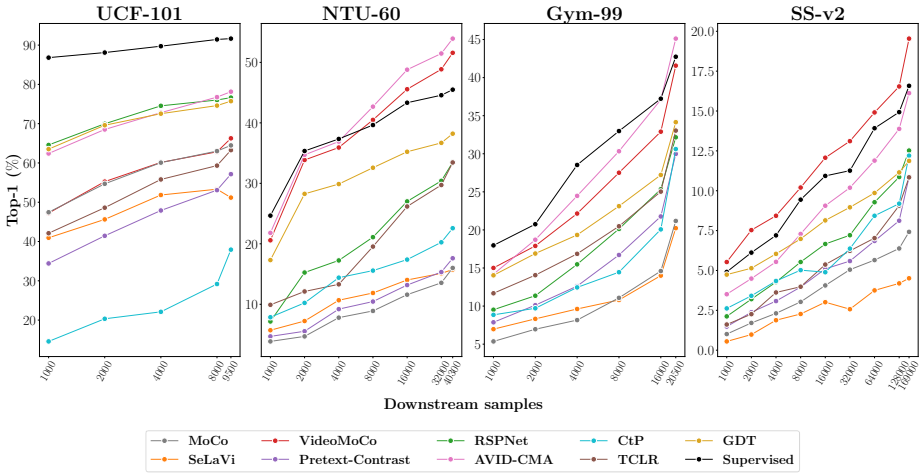

Fig. 17: **Linear evaluation for Downstream Samples.** Comparison of video self-supervised learning methods using varying number of samples on linear evaluation for four downstream datasets. Rank changes are less significant with increasing sample size.

Table 8: **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.

	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

I Verb vs. Noun in Downstream Action Recognition

EPIC-Kitchens-100 [13] 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 8 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. 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.