Counting Out Time: Class Agnostic Video Repetition Counting in the Wild

Debidatta Dwibedi [1], Yusuf Aytar [2], Jonathan Tompson [1], Pierre Sermanet [1], and Andrew Zisserman [2]

[1] Google Research [2] DeepMind
{debidatta, yusufaytar, tompson, sermanet, zisserman}@google.com

Abstract

We present an approach for estimating the period with which an action is repeated in a video. The crux of the approach lies in constraining the period prediction module to use temporal self-similarity as an intermediate representation bottleneck that allows generalization to unseen repetitions in videos in the wild. We train this model, called RepNet, with a synthetic dataset that is generated from a large unlabeled video collection by sampling short clips of varying lengths and repeating them with different periods and counts. This combination of synthetic data and a powerful yet constrained model, allows us to predict periods in a class-agnostic fashion. Our model substantially exceeds the state of the art performance on existing periodicity (PERTUBE) and repetition counting (QUVA) benchmarks. We also collect a new challenging dataset called Countix (∼90 times larger than existing datasets) which captures the challenges of repetition counting in real-world videos. Project webpage: https://sites.google.com/view/repnet.

Figure 1: We present RepNet, which leverages a temporal self-similarity matrix as an intermediate layer to predict the period length and periodicity of each frame in the video.

1. Introduction

Picture the most mundane of scenes – a person eating by themselves in a cafe. They might be stirring sugar in their coffee while chewing their food, and tapping their feet to the background music. This person is doing at least three periodic activities in parallel. Repeating actions and processes are ubiquitous in our daily lives. These range from organic cycles, such as heart beats and breathing, through programming and manufacturing, to planetary cycles like the day-night cycle and seasons. Thus the need for recognizing repetitions in videos is pervasive, and a system that is able to identify and count repetitions in video will benefit any perceptual system that aims to observe and understand our world for an extended period of time.

Repetitions are also interesting for the following reasons: (1) there is usually an intent or a driving cause behind something happen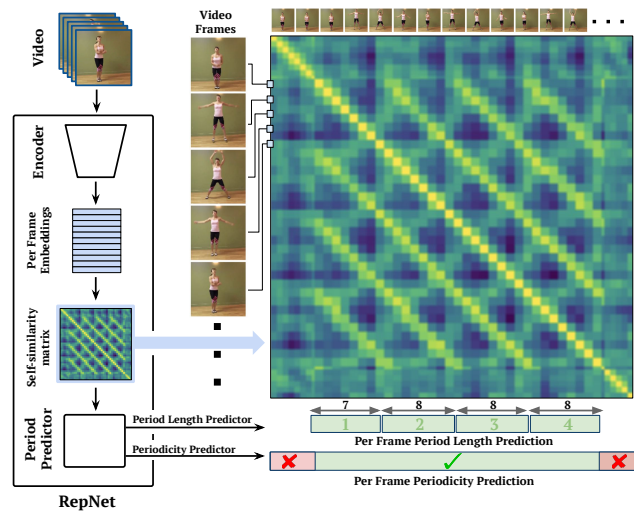ing multiple times; (2) the same event can be observed again but with slight variations; (3) there may be gradual changes in the scene as a result of these repetitions; (4) they provide us with unambiguous *action units*, a subsequence in the action that can be segmented in time (for example if you are chopping an onion, the action unit is the manipulation action that is repeated to produce additional slices). Due to the above reasons, any agent interacting with the world would benefit greatly from such a system. Furthermore, repetition counting is pertinent for many computer vision applications; such as counting the number of times an exercise was done, measurement of biological events (like heartbeats), etc.

Yet research in periodic video understanding has been limited, potentially due to the lack of a large scale labeled video repetition dataset. In contrast, for action recognition there are large scale datasets, like Kinetics [21], but their collection at large scale is enabled by the availability of keywords/text associated with the videos. Unfortunately it is rare for videos to be labeled with annotations related to

repeated activity as the text is more likely to describe the semantic content. For this reason, we use a dataset with semantic action labels typically used for action recognition (*Kinetics*) and manually choose videos of those classes with periodic motion (*bouncing, clapping* etc.). We proceed to label the selected videos with the number of repetitions present in each clip.

Manual labelling limits the number of videos that can be annotated – labelling is tedious and expensive due to the temporally fine-grained nature of the task. In order to increase the amount of training data, we propose a method to create synthetic repetition videos by repeating clips from existing videos with different periods. Since we are synthesizing these videos, we also have precise annotations for the period and count of repetitions in the videos, which can be used for training models using supervised learning. However, as we find in our work, such synthetic videos fail to capture all the nuances of real repeated videos and are prone to over-fitting by high-capacity deep learning models. To address this issue, we propose a data augmentation strategy for synthetic videos so that models trained on them transfer to real videos with repetitions. We use a combination of real and synthetic data to develop our model.

In this paper, our objective is a single model that works for many classes of periodic videos, and indeed, also for classes of videos unseen during training. We achieve this by using an intermediate representation that encourages generalization to unseen classes. This representation – a temporal self-similarity matrix – is used to predict the period with which an action is repeating in the video. This common representation is used across different kinds of repeating videos enabling the desired generalization. For example, whether a person is doing push ups, or a kid is swinging in a playground, the self-similarity matrix is the shared parameterization from which the number of repetitions is inferred. This extreme bottleneck (the number of channels in the feature map reduces from 512 to 1) also aids generalization from synthetic data to real data. The other advantage of this representation is that model interpretability is baked into the network architecture as we force the network to predict the period from the self-similarity matrix only, as opposed to inferring the period from latent high-dimensional features.

We focus on two tasks: (i) *Repetition counting*, identifying the number of repeats in the video. We rephrase this problem as first estimating per frame period lengths, and then converting them to a repetition count; (ii) *Periodicity detection*, identifying if the current frame is a part of a repeating temporal pattern or not. We approach this as a per-frame binary classification problem. A visual explanation of these tasks and the overview of our solution is shown in Figure 1.

Our main contributions in this paper are: (i) RepNet, a neural network architecture designed for counting repetitions in videos in the wild. (ii) A method to generate and augment synthetic repetition videos from unlabeled videos. (iii) By training RepNet on the synthetic dataset we outperform the state-of-the-art methods on both repetition counting and periodicity detection tasks over existing benchmarks by a substantial margin. (iv) A new video repetition counting dataset, *Countix*, which is ~ 90 times larger than the previous largest dataset.

2. Related Work

Periodicity Estimation. Extracting periodicity (detection of periodic motion) and the period by leveraging the auto-correlation in time series is a well-studied problem [40, 45]. Period estimation in videos has been done using periodograms on top of auto-correlation [9] or Wavelet transforms on hand-designed features derived from optical flow [35]. The extracted periodic motion has supported multiple tasks including 3D reconstruction [4, 27] and bird species classification [26]. Periodicity has been used for various applications [9, 30, 32, 36] including temporal pattern classification [33].

Temporal Self-similarity Matrix (TSM). TSMs are useful representations for human action recognition [20, 22, 41] and gait analysis [5, 6] due to their robustness against large viewpoint changes when paired with appropriate feature representations. A TSM based on Improved Dense Trajectories [46] is used in [31] for unsupervised identification of periodic segments in videos using special filters. Unlike these approaches, we use TSM as an intermediate layer in an end-to-end neural network architecture, which acts as an information bottleneck.

Synthetic Training Data. The use of synthetic training data in computer vision is becoming more common place. Pasting object patches on real images has been shown to be effective as training data for object detection [12, 15, 42] and human pose estimation [43]. Blending multiple videos or multiple images together has been useful for producing synthetic training data for specific tasks [2] as well as regularizing deep learning models [50, 51]. Synthetic data for training repetition counting was first proposed by [25]. They introduce a dataset of synthetic repeating patterns and use this to train a deep learning based counting model. However, the data they use for training consists of hand-designed random patterns that do not appear realistic. As shown in [35], these patterns are not diverse enough to capture all the nuances of repetitions in real videos. Instead, we propose to create synthetic training dataset of realistic video repetitions from existing video datasets.

Counting in Computer Vision. Counting objects and people in images [3, 7, 24, 28, 49] is an active area in computer vision. On the other hand, video repetition counting [25, 35] has attracted less attention from the community in the deep learning era. We build on the idea of [25] of predicting the period (cycle length), though [25] did not use a TSM.

Temporally Fine-grained Tasks. Repetition counting and periodicity detection are temporally fine-grained tasks like temporal action localization [8, 38], per-frame phase clas-
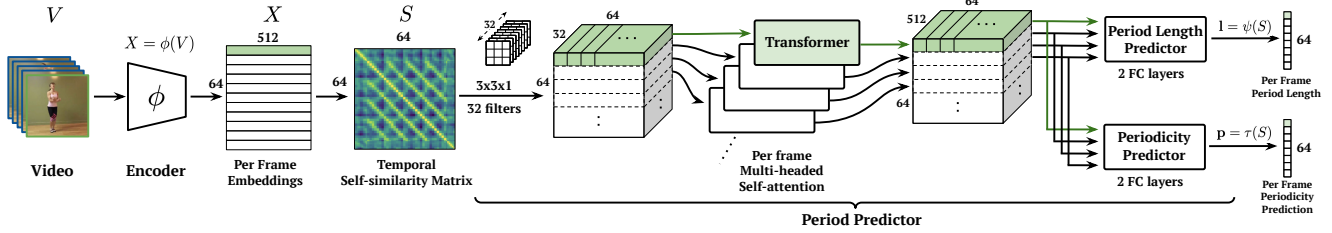
Figure 2: RepNet architecture. The features produced by a single video frame is highlighted with the green color throughout the network.

sification [11] and future anticipation [10]. We leverage the interfaces previously used to collect action localization datasets such as [16, 23, 39] to create our repetition dataset Countix. Instead of annotating semantic segments, we label the extent of the periodic segments in videos and the number of repetitions in each segment.

3. RepNet Model

In this section we introduce our RepNet architecture, which is composed of two learned components, the encoder and the period predictor, with a temporal self-similarity layer in between them.

Assume we are given a video $V = [v_1, v_2, ..., v_N]$ as a sequence of N frames. First we feed the video V to an image encoder ϕ as $X = \phi(V)$ to produce per-frame embeddings $X = [x_1, x_2, ..., x_N]^\top$. Then, using the embeddings X we obtain the self-similarity matrix S by computing pairwise similarities S_{ij} between all pairs of embeddings. Finally, S is fed to the period predictor module which outputs two elements for each frame: period length estimate $l = \psi(S)$ and periodicity score $p = \tau(S)$. The period length is the rate at which a repetition is occurring while the periodicity score indicates if the frame is within a periodic portion of the video or not. The overall architecture can be viewed in the Figure 1 and a more detailed version can be seen in Figure 2.

3.1. Encoder

Our encoder ϕ is composed of three main components:
Convolutional feature extractor: We use ResNet-50[18] architecture as our base convolutional neural network (CNN) to extract 2D convolutional features from individual frames v_i of the input video. These frames are $112 \times 112 \times 3$ in size. We use the output of `conv4_block3` layer to have a larger spatial 2D feature map. The resulting per-frame features are of size $7 \times 7 \times 1024$.
Temporal Context: We pass these convolutional features through a layer of 3D convolutions to add local temporal information to the per-frame features. We use 512 filters of size $3 \times 3 \times 3$ with ReLU activation. The temporal context helps modeling short-term motion [13, 48] and enables the model to distinguish between similar looking frames but with different motion (e.g. hands moving up or down while exercising).
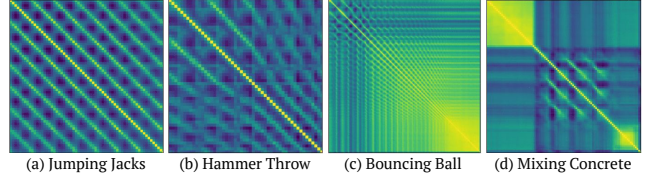

(a) Jumping Jacks (b) Hammer Throw (c) Bouncing Ball (d) Mixing Concrete

Figure 3: Diversity of temporal self-similarity matrices found in real-world repetition videos (yellow means high similarity, blue means low similarity). (a) Uniformly repeated periodic motion (jumping jacks) (b) Repetitions with acceleration (athlete performing hammer throw) (c) Repetitions with decreasing period (a bouncing ball losing speed due to repeated bounces) (d) Repeated motion preceded and succeeded by no motion (waiting to mix concrete, mixing concrete, stopped mixing). A complex model is needed to predict the period and periodicity from such diverse self-similarity matrices.

Dimensionality reduction: We reduce the dimensionality of extracted spatio-temporal features by using Global 2D Max-pooling over the spatial dimensions and to produce embedding vectors x_i corresponding to each frame v_i in the video. By collapsing the spatial dimensions we remove the need for tracking the region of interest as done explicitly in prior methods [6, 9, 33].

3.2. Temporal Self-similarity Matrix (TSM)

After obtaining latent embeddings x_i for each frame v_i, we construct the self-similarity matrix S by computing all pairwise similarities $S_{ij} = f(x_i, x_j)$ between pairs of embeddings x_i and x_j, where $f(.)$ is the similarity function. We use the negative of the squared euclidean distance as the similarity function, $f(a, b) = -||a - b||^2$, followed by row-wise softmax operation.

As the TSM has only one channel, it acts as an information bottleneck in the middle of our network and provides regularization. TSMs also make the model *temporally* interpretable which brings further insights to the predictions made by the model. Some examples can be viewed in Figure 3.

3.3. Period Predictor

The final module of RepNet is the period predictor. This module accepts the self-similarity matrix $S =$

$[s_1, s_2, ..., s_N]^\mathsf{T}$ where each row s_i is the per frame self-similarity representation, and generates two outputs: per frame period length estimation $l = \psi(S)$, and per-frame binary periodicity classification $p = \tau(S)$. Note that both l and p are vectors and their elements are per frame predictions (i.e. l_i is the predicted period length for the ith frame).

The architecture of the period predictor module can be viewed in Figure 2. Note that predictors ψ and τ share a common architecture and weights until the last classification phase. The shared processing pipeline starts with 32 2D convolutional filters of size 3×3, followed by a transformer [44] layer which uses a multi-headed attention with trainable positional embeddings in the form of a 64 length variable that is learned by training. We use 4 heads with 512 dimensions in the transformer with each head being 128 dimensions in size. After the shared pipeline, we have two classifiers, period length classifier ψ and periodicity classifier τ. Each of them consists of two fully connected layers of size 512.

3.4. Losses

Our periodicity classifier τ outputs per frame periodicity classification p_i and uses a binary classification loss (binary cross-entropy) for optimization. Our period length estimator ψ outputs per frame period length estimation $l_i \in L$ where the classes are discrete period lengths $L = \{2, 3, ..., \frac{N}{2}\}$ where N is the number of input frames. We use a multi-class classification objective (softmax cross-entropy) for optimizing our model. For all our experiments we use $N = 64$. We sample the input video with different frame rates as described below to predict larger period lengths.

3.5. Inference

Inferring the count of repetitions robustly for a given video requires two main operations:

Count from period length predictions: We sample consecutive non-overlapping windows of N frames and provide it as input to RepNet which outputs per-frame periodicity p_i and period lengths l_i. We define *per-frame count* as $\frac{p_i}{l_i}$. The overall repetition count is computed as the sum of all per-frame counts: $\sum_{i=1}^{N} \frac{p_i}{l_i}$. The evaluation datasets for repetition counting have only periodic segments. Hence, we set p_i to 1 as default for counting experiments.

Multi-speed evaluation: As our model can predict period lengths up to 32, for covering much longer period lengths we sample input video with different frame rates. (i.e. we play the video at $1\times$, $2\times$, $3\times$, and $4\times$ speeds). We choose the frame rate which has the highest score for the predicted period. This is similar to what [25] do at test time.

4. Training with Synthetic Repetitions

A potential supervised approach to period estimation would be collecting a large training set of periodic videos
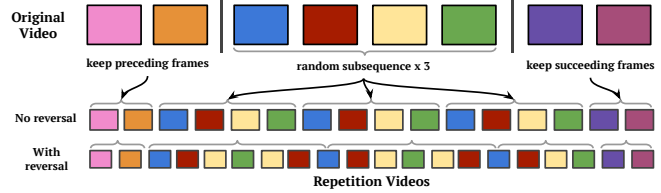

Figure 4: Our synthetic data generation pipeline that produces videos with repetitions from any video. We randomly sample a portion of a video that we repeat N times to produce synthetic repeating videos. More details in Section 4

and annotating the beginning and the end of every period in all repeating actions. However, collecting such a dataset is expensive due to the fine-grained nature of the task.

As a cheaper and more scalable alternative, we propose a training strategy that makes use of synthetically generated repetitions using unlabeled videos in the wild (e.g. YouTube). We generate synthetic periodic videos using randomly selected videos, and predict per frame periodicity and period lengths. Next, we'll explain how we generate synthetic repetitions, and introduce camera motion augmentations which are crucial for training effective counting models from synthetic videos.

4.1. Synthetic Repetition Videos

Given a large set of unlabeled videos, we propose a simple yet effective approach for creating synthetic repetition videos (shown in Figure 4) from them. The advantage of using real videos to create synthetic data is that the training data is much closer to real repeated videos when compared to using synthetic patterns. Another advantage of using real videos is that using a big dataset like Kinetics ensures that the diversity of data seen by the model is huge. This allows us to train big complex models that can work on real repetition videos.

Our pipeline starts with sampling a random video *V* from a dataset of videos. We use the training set of Kinetics [21] **without any labels**. Then, we sample a clip *C* of random length *P* frames from *V*. This clip *C* is repeated *K* times (where $K > 1$) to simulate videos with repetitions. We randomly concatenate the reversed clip before repeating to simulate actions where the motion is done in reverse in the period (like jumping jacks). Then, we pre-pend and append the repeating frames with other non-repeating segments from *V*, which are just before and after *C*, respectively. The lengths of these aperiodic segments are chosen randomly and can potentially be zero too. This operation makes sure that there are both periodic and non-periodic segments in the generated video. Finally, each frame in the repeating part of the generated video is assigned a period length label *P*. A periodicity label is also generated indicating whether the frame is inside or outside the repeating portion of the generated video.
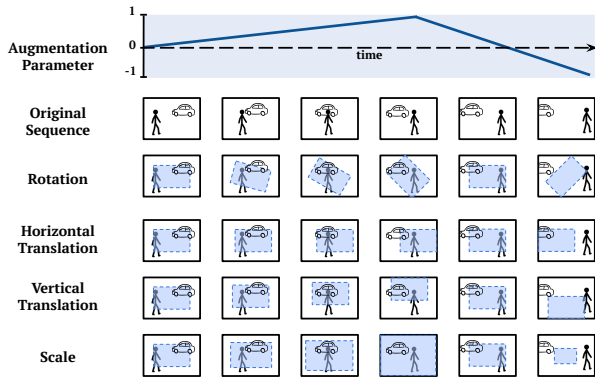
Figure 5: Camera motion augmentation. We vary the augmentation parameters for each type of camera motion smoothly over time as opposed to randomly sampling them independently for each frame. This ensures that the augmented sequence still retains the temporal coherence naturally present in videos.

4.2. Camera Motion Augmentation

A crucial step in the synthetic video generation is camera motion augmentation (shown in Figure 5). Although it is not feasible to predict views of an arbitrarily moving camera without knowing the 3D structure, occluded parts and lighting sources in the scene, we can approximate it using affine image transformations. Here we consider the affine motion of a viewing frame over the video, which includes temporally smooth changes in rotation, translation, and scale. As we will show in section 6, when we train without these augmentations, the training loss quickly decreases but the model does not transfer to real repetition videos. We empirically find camera motion augmentation is a vital part of training effective models with synthetic videos.

To achieve camera motion augmentations, we temporally vary the parameters for various motion types in a continuous manner as the video proceeds. For example, we change the angle of rotation smoothly over time. This ensures that the video is temporally coherent even after the augmentation. Figure 5 illustrates how temporal augmentation parameter drives viewing frame (shown in blue rectangle) for each motion type. This results in videos with fewer near duplicates across the repeating segments.

5. Countix Dataset

Existing datasets for video repetition counting [25, 35] are mostly utilized for testing purposes, mainly due to their limited size. The most recent and challenging benchmark on this task is the QUVA repetition dataset [35] which includes realistic repetition videos with occlusion, camera movement, and changes in speed of the repeated actions. It is composed of 100 class-agnostic test videos, annotated with the count of repeated actions. Despite being challenging, its limited size makes it hard to cover diverse semantic categories of repetitions. Also training supervised deep

models with this scale of data is not feasible.

To increase the semantic diversity and scale up the size of counting datasets, we introduce the **Countix** dataset: a real world dataset of repetition videos collected in the wild (i.e. YouTube) covering a wide range of semantic settings with significant challenges such as camera and object motion, diverse set of periods and counts, and changes in the speed of repeated actions. Countix include repeated videos of workout activities (squats, pull ups, battle rope training, exercising arm), dance moves (pirouetting, pumping fist), playing instruments (playing ukulele), using tools repeatedly (hammer hitting objects, chainsaw cutting wood, slicing onion), artistic performances (hula hooping, juggling soccer ball), sports (playing ping pong and tennis) and many others. Figure 6 illustrates some examples from the dataset as well as the distribution of repetition counts and period lengths.

Dataset Collection: The Countix dataset is a subset of the Kinetics [21] dataset annotated with segments of repeated actions and corresponding counts. During collection we first manually choose a subset of classes from Kinetics which have a higher chance of repetitions happening in them for e.g. *jumping jacks*, *slicing onion* etc., rather than classes like *head stand* or *alligator wrestling*. We crowdsource the labels for repetition segments and counts for the selected classes. The interface used is similar to what is typically used to mark out temporal segments for fine-grained action recognition[16, 34]. The annotators are asked to first segment the part of the video that contains valid repetitions with unambiguous counts. The annotators then proceed to count the number of repetitions in each segment. This count serves as the label for the entire clip. We reject segments with insignificant overlap in the temporal extents marked out by 3 different annotators. For the remaining segments, we use the median of the count annotations and segment extents as the ground truth. The *Countix* dataset is about 90 times bigger than the previous largest repetition counting dataset (QUVA Repetition Dataset). The detailed statistics can be viewed in Table 1. The dataset is available on the project webpage.

Note that we retain the train/val/test splits from the Kinetics dataset. Hence, models pre-trained with Kinetics may be used for training counting models without data leakage.

	QUVA	Countix
No. of Videos in Train set	0	4588
No. of Videos in Val. set	0	1450
No. of Videos in Test set	100	2719
Duration Avg. $\pm$ Std (s)	17.6 ± 13.3	6.13 ± 3.08
Duration Min./Max. (s)	2.5 / 64.2	0.2 / 10.0
Count Avg $\pm$ Std	12.5 ± 10.4	6.84 ± 6.76
Count Min./Max.	4 / 63	2 / 73

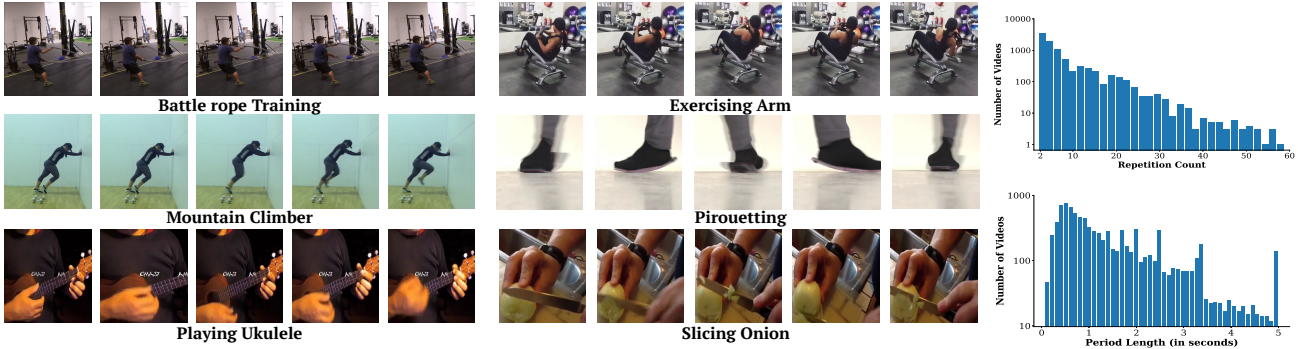**Table 1:** Statistics of Countix and QUVA Repetition datasets.

Figure 6: Countix dataset. In the left two columns, we present examples of repeating videos from the Countix dataset. The last column shows the distribution of the number of the videos in the dataset with respect to the count and the period length labels.

6. Experiments

We start by explaining the existing benchmarks and the evaluation metrics used in repetition counting. We next present a series of ablation studies that demonstrate which components and design choices are crucial. Then we compare our performance on the existing benchmarks and show that RepNet clearly outperforms the state-of-the-art methods on repetition counting and periodicity detection. Finally, through qualitative analysis, we bring more insight into our model.

6.1. Benchmarks and Evaluation Metrics

Here we discuss two established benchmark datasets for periodicity detection and repetition counting together with the commonly used evaluation metrics.

Periodicity detection: The benchmark dataset for this task is the PERTUBE dataset [31], which has per frame labels identifying periodicity, if the frame is a part of a repeating action or not. [31] casts the problem as a binary per frame classification task and reports precision, recall, F1 score and overlap. We follow the same metrics for evaluation.

Repetition counting: As discussed in Section 5, the QUVA dataset [35] is the largest available dataset for repetition counting. The existing literature uses two main metrics for evaluating repetition counting in videos:

Off-By-One (OBO) count error. If the predicted count is within one count of the ground truth value, then the video is considered to be classified correctly, otherwise it is a mis-classification. The OBO error is the mis-classification rate over the entire dataset.

Mean Absolute Error (MAE) of count. This metric measures the absolute difference between the ground truth count and the predicted count, and then normalizes it by dividing with the ground truth count. The reported MAE error is the mean of the normalized absolute differences over the entire dataset.

Both in our ablation experiments and state-of-the-art comparisons we follow [25, 35] and report OBO and MAE errors over the QUVA and Countix validation set. We also

provide a final score on the Countix test set in Table 7.

6.2. Implementation Details

We implement our method in Tensorflow [1]. We initialize the encoder with weights from an ImageNet pre-trained ResNet-50 checkpoint. We train the model for $400K$ steps with a learning rate of 6×10^{-6} with the ADAM optimizer and batch size of 5 videos (each with 64 frames). For all ablation studies we train the model on the synthetic repetition data unless otherwise stated. Additional details are provided on the project webpage.

6.3. Ablations

We perform a number of ablations to justify the decisions made while designing RepNet.

Temporal Self-similarity Matrix (TSM): In Table 2 we compare the impact of adding the TSM to the model. Models without the TSM apply the transformer directly on the per-frame embeddings produced by the encoder. The temporal self-similarity matrix substantially improves performance on all metrics and validation datasets whether we train the model using synthetic repetition videos, real Countix videos or a mix of both. Moreover, the TSM layer helps in generalizing to real repetition videos even when the model has only seen synthetic repetition videos (rows 1 and 2 in Table 2).

Training Data Source: We vary the training data sources in Table 2 while comparing our synthetic repetition videos with real ones from the Countix dataset. We find that RepNet achieves similar performance on the Countix dataset when trained with synthetic videos or with the real repetition videos of the Countix dataset. But the model trained on Countix dataset is worse on the QUVA dataset compared to training on synthetic repeating videos. This shows using a synthetic repeating dataset results in a model that performs competitively on unseen classes as well. The best performance in terms of OBO error is achieved when the model is trained with both the datasets.

Alternative Period Prediction Architectures: In Table 3,

we compare the transformer architecture with other contemporary sequence models like LSTM and Temporal CNNs. We also compare it with a model that uses a 2D CNN on the self-similarity matrix itself. We find that the transformer architecture performs better than these alternatives.

Camera Motion Augmentation: In Table 4 we show the value of camera motion augmentation when using the synthetic repeating dataset. We observe that performance on both datasets improves when the fraction of samples in the batch with camera motion augmentation is increased.

TSM	Training Data Source	QUVA		Countix (Val)	
		MAE	OBO	MAE	OBO
	Synthetic	1.2853	0.64	1.1671	0.5510
✓	Synthetic	**0.1035**	**0.17**	**0.3100**	**0.2903**
	Countix	0.7584	0.72	0.6483	0.5448
✓	Countix	**0.3225**	**0.34**	**0.3468**	**0.2949**
	Synthetic + Countix	0.6388	0.57	0.8889	0.4848
✓	Synthetic + Countix	**0.1315**	**0.15**	**0.3280**	**0.2752**

Table 2: Ablation of architecture with or without the temporal self-similarity matrix (TSM) with different training data sources.

Architecture	QUVA		Countix (Val)	
	MAE	OBO	MAE	OBO
Transformer	**0.1035**	**0.17**	**0.3100**	**0.2903**
LSTM [19]	0.1395	0.18	0.6895	0.3579
2D CNN	0.1186	**0.17**	0.4440	0.3310
1D Temporal CNN	0.3229	0.23	0.7077	0.3641

Table 3: Performance of different period prediction architectures when trained with synthetic data.

Augmentation Fraction	QUVA		Countix (Val)	
	MAE	OBO	MAE	OBO
0.00	0.7178	0.32	1.2629	0.4683
0.25	0.1414	0.17	0.4430	0.3303
0.50	0.1202	**0.15**	0.3729	0.2993
0.75	**0.1035**	0.17	**0.3100**	0.2903
1.00	0.1710	0.17	0.3346	**0.2848**

Table 4: Impact of camera motion augmentation when trained with synthetic data.

6.4. Evaluation on Benchmarks

We compare our system with the current state-of-the-art methods on periodicity detection and repetition counting on the established benchmarks described in Section 6.1.

Periodicity Detection. We report the performance for measuring periodicity classification by choosing the threshold that maximizes the F1 score. As done in [31] we calculate the metrics on a per video basis and average the scores. We also report Area Under the Curve (AUC) of the precision-recall curve which is independent of the threshold chosen. Our model produces an AUC of 0.969. We outperform the previous work without using any hand-designed filtering methods mentioned in [31] (see Table 5). Our model trained entirely on synthetic data works out of the box for the task of periodicity detection in real videos.
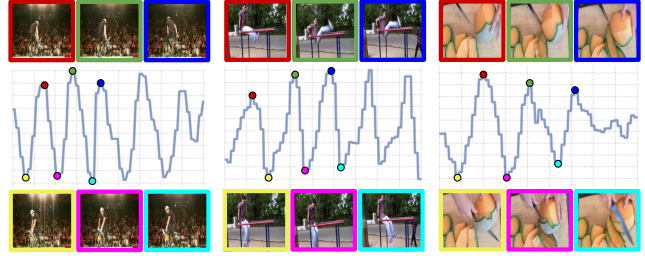

Figure 7: 1D PCA projections of the encoder features over time. Note that even 1D projections of the learned features are encoding the periodicity fairly well. Frames with similar embeddings across different periods show similar states in the video (angle of rotation of biker, position of legs of person and position of knife). Best viewed with zoom. Video version on webpage.

Repetition Counting. In Table 6 we compare our RepNet model with previous models and show it outperforms existing methods by a significant margin and therefore establishing a new state-of-the-art for this dataset. Experimental results on the test set of Countix dataset indicate that RepNet is an effective baseline for the video repetition counting task (see Table 7).

Model	Recall	Precision	F1	Overlap
Power spectrum baseline [31]	0.793	0.611	0.668	0.573
P-MUCOS [31]	0.841	0.757	0.77	0.677
RepNet (Ours)	**0.859**	**0.821**	**0.820**	**0.731**

Table 5: Periodicity detection results on the PERTUBE Dataset

Model	MAE	OBO
Visual quasi-periodicity [33]	0.385	0.51
Live Repetition Counting [25]	0.482	0.55
Div-Grad-Curl [35]	0.232	0.38
RepNet (Ours)	**0.104**	**0.17**

Table 6: Counting Results on the QUVA dataset.

Model	MAE	OBO
RepNet	0.3641	0.3034

Table 7: Counting Results on the Countix test set.

6.5. Qualitative analysis

Temporal Self-similarity Matrix. TSM provides us with meaningful interpretations about the model's predictions. It also contains additional information regarding acceleration and deceleration of the action. We show some examples of self-similarity matrices in Figure 3.

1D PCA Embeddings. We also investigate the learned embeddings which are used to produce the TSM. In Figure 7, we project the 512 dimensional vector to 1 dimension using
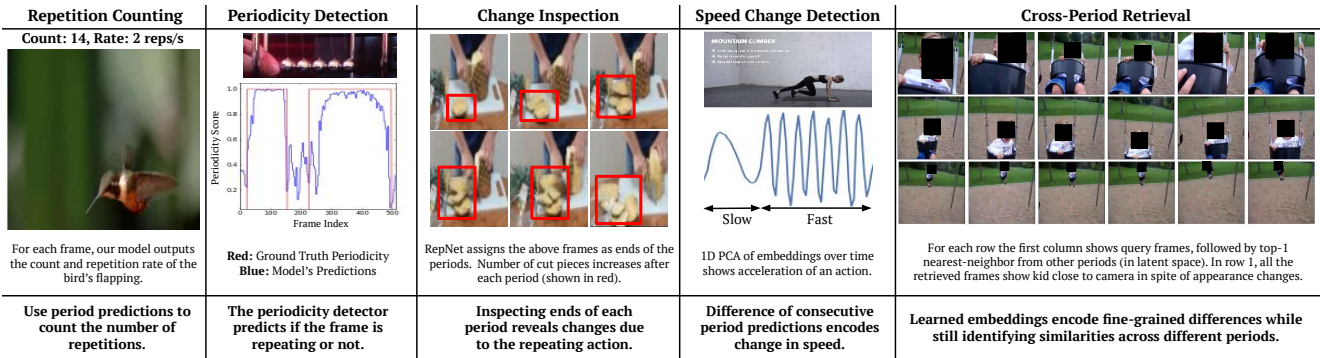
Repetition Counting	Periodicity Detection	Change Inspection	Speed Change Detection	Cross-Period Retrieval
For each frame, our model outputs the count and repetition rate of the bird's flapping.	**Red:** Ground Truth Periodicity **Blue:** Model's Predictions	RepNet assigns the above frames as ends of the periods. Number of cut pieces increases after each period (shown in red).	1D PCA of embeddings over time shows acceleration of an action.	For each row the first column shows query frames, followed by top-1 nearest-neighbor from other periods (in latent space). In row 1, all the retrieved frames show kid close to camera in spite of appearance changes.
Use period predictions to count the number of repetitions.	**The periodicity detector predicts if the frame is repeating or not.**	**Inspecting ends of each period reveals changes due to the repeating action.**	**Difference of consecutive period predictions encodes change in speed.**	**Learned embeddings encode fine-grained differences while still identifying similarities across different periods.**

Figure 8: One model, many domains and applications. A single model is capable of performing these tasks over videos from many diverse domains (animal movement, physics experiments, humans manipulating objects, people exercising, child swinging) in a class-agnostic manner. Please see the project webpage for videos showcasing these tasks.

the first principal component of the per-frame embeddings for each video. This reveals interesting quasi-sinusoidal patterns traced out by the embeddings in time. We plot the frames when the embeddings are changing directions and observe that the retrieved frames show the person or object in a similar state but in different periods.

Double Counting Errors. We observe that a common failure mode of our model is that for some actions (e.g. juggling soccer ball), it predicts half the count reported by annotators. This happens when the model considers left and right legs' motion for counting while people tend to consider the ball's up/down motion resulting in people double counting the repetitions. We believe such errors are difficult to isolate in a class-agnostic manner. But they can be fixed easily with either labeled data or post-processing methods if the application is known.

7. Applications

Predict speed changes of repetitions. Our method takes in a video clip and predicts the period of any repeated action. The consecutive difference of predicted rates encodes the rate of speed change of the repetitions. Monitoring speed changes is useful for exercise tracking applications where it might be important to know if someone is speeding up or slowing down (Column 4 in Figure 8).

Estimating frequency of processes from videos. Our model can be used to predict the count and frequency of repeating phenomena from videos for e.g. biological processes (heartbeats). [47] presented a method to reveal subtle changes by magnifying the difference in frames. We find that the output from the above system can be fed directly into our model to predict the frequency of these changes. A class-agnostic period estimator removes the need to explicitly train on these videos. On our project webpage, we show examples of repetition counting on echo-cardiogram videos which look very different from Kinetics videos .

Fine-grained cross-period retrieval. The learned embeddings are useful for performing cross-period retrieval. In

other words, the features capture similarities present across different periods while still encoding subtle differences between similar looking frames. Examples of these retrievals are shown in Figure 7 and the last column in Figure 8.

Repetitions with longer temporal extent. Many repeating phenomena occur over a longer temporal scale (in the order of days or years). Even though our model has been trained on short videos (∼10s), it can still work on videos with slow periodic events by automatically choosing a higher input frame stride. On the project webpage, we show videos where RepNet predicts the period length of a day from videos of the earth captured by satellites.

Aid self-supervised video representation learning. Self-supervised learning methods for video embeddings, e.g. Shuffle and Learn [29], Odd-One-Out networks [14], DPC [17], TCC [11] and TCN [37] are not designed to handle repetitions in sequences. RepNet can identify the repeating sections and may help in training on videos with repetitions without modifying the proposed objectives.

8. Conclusion

We have shown a simple combination of synthetic training data, together with an architecture using temporal self-similarity, results in a powerful class-agnostic repetition counting model. This model successfully detects periodicity and predicts counts over a diverse set of *actors* (objects, humans, animals, the earth) and sensors (standard camera, ultrasound, laser microscope) and has been evaluated on a vast collection of videos. With this we have addressed the case of simple repetitions, and the next step is to consider more complex cases such as multiple simultaneous repeating signals and temporal arrangements of repeating sections such as in dance steps and music.

References

[1] Martín Abadi, Paul Barham, Jianmin Chen, Zhifeng Chen, Andy Davis, Jeffrey Dean, Matthieu Devin, Sanjay Ghemawat, Geoffrey Irving, Michael Isard, et al. Tensorflow: A system for large-scale machine learning. In *12th {USENIX} Symposium on Operating Systems Design and Implementation ({OSDI} 16)*, pages 265–283, 2016. 6

[2] Jean-Baptiste Alayrac, Joao Carreira, and Andrew Zisserman. The visual centrifuge: Model-free layered video representations. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 2457–2466, 2019. 2

[3] Carlos Arteta, Victor Lempitsky, and Andrew Zisserman. Counting in the wild. In *European conference on computer vision*, pages 483–498. Springer, 2016. 2

[4] Serge Belongie and Josh Wills. Structure from periodic motion. In *International Workshop on Spatial Coherence for Visual Motion Analysis*, pages 16–24. Springer, 2004. 2

[5] Chiraz BenAbdelkader, Ross Cutler, Harsh Nanda, and Larry Davis. Eigengait: Motion-based recognition of people using image self-similarity. In *International Conference on Audio- and Video-Based Biometric Person Authentication*, pages 284–294. Springer, 2001. 2

[6] Chiraz BenAbdelkader, Ross G Cutler, and Larry S Davis. Gait recognition using image self-similarity. *EURASIP Journal on Advances in Signal Processing*, 2004(4):721765, 2004. 2, 3

[7] Lokesh Boominathan, Srinivas SS Kruthiventi, and R Venkatesh Babu. Crowdnet: A deep convolutional network for dense crowd counting. In *Proceedings of the 24th ACM international conference on Multimedia*, pages 640–644. ACM, 2016. 2

[8] Yu-Wei Chao, Sudheendra Vijayanarasimhan, Bryan Seybold, David A Ross, Jia Deng, and Rahul Sukthankar. Rethinking the faster r-cnn architecture for temporal action localization. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 1130–1139, 2018. 2

[9] Ross Cutler and Larry S. Davis. Robust real-time periodic motion detection, analysis, and applications. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 22(8):781–796, 2000. 2, 3

[10] Dima Damen, Hazel Doughty, Giovanni Maria Farinella, Sanja Fidler, Antonino Furnari, Evangelos Kazakos, Davide Moltisanti, Jonathan Munro, Toby Perrett, Will Price, et al. Scaling egocentric vision: The epic-kitchens dataset. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 720–736, 2018. 3

[11] Debidatta Dwibedi, Yusuf Aytar, Jonathan Tompson, Pierre Sermanet, and Andrew Zisserman. Temporal cycle-consistency learning. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 1801–1810, 2019. 3, 8

[12] Debidatta Dwibedi, Ishan Misra, and Martial Hebert. Cut, paste and learn: Surprisingly easy synthesis for instance detection. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 1301–1310, 2017. 2

[13] Debidatta Dwibedi, Jonathan Tompson, Corey Lynch, and Pierre Sermanet. Learning actionable representations from visual observations. In *2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 1577–

1584. IEEE, 2018. 3

[14] Basura Fernando, Hakan Bilen, Efstratios Gavves, and Stephen Gould. Self-supervised video representation learning with odd-one-out networks. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 3636–3645, 2017. 8

[15] Georgios Georgakis, Arsalan Mousavian, Alexander C Berg, and Jana Kosecka. Synthesizing training data for object detection in indoor scenes. *arXiv preprint arXiv:1702.07836*, 2017. 2

[16] Chunhui Gu, Chen Sun, David A Ross, Carl Vondrick, Caroline Pantofaru, Yeqing Li, Sudheendra Vijayanarasimhan, George Toderici, Susanna Ricco, Rahul Sukthankar, et al. Ava: A video dataset of spatio-temporally localized atomic visual actions. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 6047–6056, 2018. 3, 5

[17] Tengda Han, Weidi Xie, and Andrew Zisserman. Video representation learning by dense predictive coding. In *Proceedings of the IEEE International Conference on Computer Vision Workshops*, pages 0–0, 2019. 8

[18] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778, 2016. 3

[19] Sepp Hochreiter and Jürgen Schmidhuber. Long short-term memory. *Neural computation*, 9(8):1735–1780, 1997. 7

[20] Imran N Junejo, Emilie Dexter, Ivan Laptev, and Patrick Perez. View-independent action recognition from temporal self-similarities. *IEEE transactions on pattern analysis and machine intelligence*, 33(1):172–185, 2010. 2

[21] Will Kay, Joao Carreira, Karen Simonyan, Brian Zhang, Chloe Hillier, Sudheendra Vijayanarasimhan, Fabio Viola, Tim Green, Trevor Back, Paul Natsev, Mustafa Suleyman, and Andrew Zisserman. The kinetics human action video dataset. *arXiv preprint arXiv:1705.06950*, 2017. 1, 4, 5

[22] Marco Körner and Joachim Denzler. Temporal self-similarity for appearance-based action recognition in multi-view setups. In *International Conference on Computer Analysis of Images and Patterns*, pages 163–171. Springer, 2013. 2

[23] Hilde Kuehne, Ali Arslan, and Thomas Serre. The language of actions: Recovering the syntax and semantics of goal-directed human activities. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 780–787, 2014. 3

[24] Victor Lempitsky and Andrew Zisserman. Learning to count objects in images. In *Advances in neural information processing systems*, pages 1324–1332, 2010. 2

[25] Ofir Levy and Lior Wolf. Live repetition counting. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 3020–3028, 2015. 2, 4, 5, 6, 7

[26] Wen Li and Dezhen Song. Automatic bird species detection using periodicity of salient extremities. In *2013 IEEE International Conference on Robotics and Automation*, pages 5775–5780. IEEE, 2013. 2

[27] Xiu Li, Hongdong Li, Hanbyul Joo, Yebin Liu, and Yaser Sheikh. Structure from recurrent motion: From rigidity to recurrency. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 3032–3040, 2018. 2

[28] Erika Lu, Weidi Xie, and Andrew Zisserman. Class-agnostic

counting. In *Asian Conference on Computer Vision*, pages 669–684. Springer, 2018. 2

[29] Ishan Misra, C Lawrence Zitnick, and Martial Hebert. Shuffle and learn: unsupervised learning using temporal order verification. In *European Conference on Computer Vision*, pages 527–544. Springer, 2016. 8

[30] Sourabh A Niyogi and Edward H Adelson. Analyzing gait with spatiotemporal surfaces. In *Proceedings of 1994 IEEE Workshop on Motion of Non-rigid and Articulated Objects*, pages 64–69. IEEE, 1994. 2

[31] Costas Panagiotakis, Giorgos Karvounas, and Antonis Argyros. Unsupervised detection of periodic segments in videos. In *2018 25th IEEE International Conference on Image Processing (ICIP)*, pages 923–927. IEEE, 2018. 2, 6, 7

[32] Silvia L Pintea, Jian Zheng, Xilin Li, Paulina JM Bank, Jacobus J van Hilten, and Jan C van Gemert. Hand-tremor frequency estimation in videos. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 0–0, 2018. 2

[33] Erik Pogalin, Arnold WM Smeulders, and Andrew HC Thean. Visual quasi-periodicity. In *2008 IEEE Conference on Computer Vision and Pattern Recognition*, pages 1–8. IEEE, 2008. 2, 3, 7

[34] Joseph Roth, Sourish Chaudhuri, Ondrej Klejch, Radhika Marvin, Andrew Gallagher, Liat Kaver, Sharadh Ramaswamy, Arkadiusz Stopczynski, Cordelia Schmid, Zhonghua Xi, et al. Ava-activespeaker: An audio-visual dataset for active speaker detection. *arXiv preprint arXiv:1901.01342*, 2019. 5

[35] Tom FH Runia, Cees GM Snoek, and Arnold WM Smeulders. Real-world repetition estimation by div, grad and curl. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 9009–9017, 2018. 2, 5, 6, 7

[36] Steven M Seitz and Charles R Dyer. View-invariant analysis of cyclic motion. *International Journal of Computer Vision*, 25(3):231–251, 1997. 2

[37] Pierre Sermanet, Corey Lynch, Yevgen Chebotar, Jasmine Hsu, Eric Jang, Stefan Schaal, Sergey Levine, and Google Brain. Time-contrastive networks: Self-supervised learning from video. In *2018 IEEE International Conference on Robotics and Automation (ICRA)*, pages 1134–1141. IEEE, 2018. 8

[38] Zheng Shou, Dongang Wang, and Shih-Fu Chang. Temporal action localization in untrimmed videos via multi-stage cnns. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 1049–1058, 2016. 2

[39] Gunnar A Sigurdsson, Gül Varol, Xiaolong Wang, Ali Farhadi, Ivan Laptev, and Abhinav Gupta. Hollywood in homes: Crowdsourcing data collection for activity understanding. In *European Conference on Computer Vision*, pages 510–526. Springer, 2016. 3

[40] Petre Stoica, Randolph L Moses, et al. Spectral analysis of signals. 2

[41] Chuan Sun, Imran Nazir Junejo, Marshall Tappen, and Hassan Foroosh. Exploring sparseness and self-similarity for action recognition. *IEEE Transactions on Image Processing*, 24(8):2488–2501, 2015. 2

[42] Jonathan Tremblay, Aayush Prakash, David Acuna, Mark Brophy, Varun Jampani, Cem Anil, Thang To, Eric Cameracci, Shaad Boochoon, and Stan Birchfield. Training deep networks with synthetic data: Bridging the reality gap by domain randomization. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops*, pages 969–977, 2018. 2

[43] Gul Varol, Javier Romero, Xavier Martin, Naureen Mahmood, Michael J. Black, Ivan Laptev, and Cordelia Schmid. Learning from synthetic humans. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, July 2017. 2

[44] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. In *Advances in neural information processing systems*, pages 5998–6008, 2017. 4

[45] Michail Vlachos, Philip Yu, and Vittorio Castelli. On periodicity detection and structural periodic similarity. In *Proceedings of the 2005 SIAM international conference on data mining*, pages 449–460. SIAM, 2005. 2

[46] Heng Wang and Cordelia Schmid. Action recognition with improved trajectories. In *Proceedings of the IEEE international conference on computer vision*, pages 3551–3558, 2013. 2

[47] Hao-Yu Wu, Michael Rubinstein, Eugene Shih, John Guttag, Frédo Durand, and William Freeman. Eulerian video magnification for revealing subtle changes in the world. 2012. 8

[48] Saining Xie, Chen Sun, Jonathan Huang, Zhuowen Tu, and Kevin Murphy. Rethinking spatiotemporal feature learning: Speed-accuracy trade-offs in video classification. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 305–321, 2018. 3

[49] Weidi Xie, J Alison Noble, and Andrew Zisserman. Microscopy cell counting and detection with fully convolutional regression networks. *Computer methods in biomechanics and biomedical engineering: Imaging & Visualization*, 6(3):283–292, 2018. 2

[50] Sangdoo Yun, Dongyoon Han, Seong Joon Oh, Sanghyuk Chun, Junsuk Choe, and Youngjoon Yoo. Cutmix: Regularization strategy to train strong classifiers with localizable features. *arXiv preprint arXiv:1905.04899*, 2019. 2

[51] Hongyi Zhang, Moustapha Cisse, Yann N Dauphin, and David Lopez-Paz. mixup: Beyond empirical risk minimization. *arXiv preprint arXiv:1710.09412*, 2017. 2