Shuffle and Learn: Unsupervised Learning using Temporal Order Verification

Ishan Misra[1] C. Lawrence Zitnick[2] Martial Hebert[1]

[1] The Robotics Institute, Carnegie Mellon University
[2] Facebook AI Research
{imisra, hebert}@cs.cmu.edu, zitnick@fb.com

Abstract. In this paper, we present an approach for learning a visual representation from the raw spatiotemporal signals in videos. Our representation is learned without supervision from semantic labels. We formulate our method as an unsupervised sequential verification task, i.e., we determine whether a sequence of frames from a video is in the correct temporal order. With this simple task and no semantic labels, we learn a powerful visual representation using a Convolutional Neural Network (CNN). The representation contains complementary information to that learned from supervised image datasets like ImageNet. Qualitative results show that our method captures information that is temporally varying, such as human pose. When used as pre-training for action recognition, our method gives significant gains over learning without external data on benchmark datasets like UCF101 and HMDB51. To demonstrate its sensitivity to human pose, we show results for pose estimation on the FLIC and MPII datasets that are competitive, or better than approaches using significantly more supervision. Our method can be combined with supervised representations to provide an additional boost in accuracy.

Keywords: Unsupervised learning; Videos; Sequence Verification; Action Recognition; Pose Estimation; Convolutional Neural Networks

1 Introduction

Sequential data provides an abundant source of information in the form of auditory and visual percepts. Learning from the observation of sequential data is a natural and implicit process for humans [1–3]. It informs both low level cognitive tasks and high level abilities like decision making and problem solving [4]. For instance, answering the question "Where would the moving ball go?", requires the development of basic cognitive abilities like prediction from sequential data like video [5].

In this paper, we explore the power of spatiotemporal signals, *i.e.*, videos, in the context of computer vision. To study the information available in a video signal in isolation, we ask the question: How does an agent learn from the spatiotemporal structure present in video without using supervised semantic labels?

Are the representations learned using the unsupervised spatiotemporal information present in videos meaningful? And finally, are these representations complementary to those learned from strongly supervised image data? In this paper, we explore such questions by using a sequential learning approach.

Sequential learning is used in a variety of areas such as speech recognition, robotic path planning, adaptive control algorithms, *etc.* These approaches can be broadly categorized [6] into two classes: prediction and verification. In sequential prediction, the goal is to predict the signal given an input sequence. A popular application of this in Natural Language Processing (NLP) is 'word2vec' by Mikolov *et al.* [7, 8] that learns distributional representations [9]. Using the continuous bag-of-words (CBOW) task, the model learns to predict a missing word given a sequence of surrounding words. The representation that results from this task has been shown to be semantically meaningful [7]. Unfortunately, extending the same technique to predict video frames is challenging. Unlike words that can be represented using limited-sized vocabularies, the space of possible video frames is extremely large [10], *eg.*, predicting pixels in a small 256×256 image leads to $256^{2\times3\times256}$ hypotheses! To avoid this complex task of predicting high-dimensional video frames, we use sequential verification.

In sequential verification, one predicts the 'validity' of the sequence, rather than individual items in the sequence. In this paper, we explore the task of determining whether a given sequence is 'temporally valid', *i.e.*, whether a sequence of video frames are in the correct temporal order, Figure 1. We demonstrate that this binary classification problem is capable of learning useful visual representations from videos. Specifically, we explore their use in the well understood tasks of human action recognition and pose estimation. But why are these simple sequential verification tasks useful for learning? Determining the validity of a sequence requires reasoning about object transformations and relative locations through time. This in turn forces the representation to capture object appearances and deformations.

We use a Convolutional Neural Network (CNN) [11] for our underlying feature representation. The CNN is applied to each frame in the sequence and trained "end-to-end" from random initialization. The sequence verification task encourages the CNN features to be both visually and temporally grounded. We demonstrate the effectiveness of our unsupervised method on benchmark action recognition datasets UCF101 [12] and HMDB51 [13], and the FLIC [14] and MPII [15] pose estimation datasets. Using our simple unsupervised learning approach for pre-training, we show a significant boost in accuracy over learning CNNs from scratch with random initialization. In fact, our unsupervised approach even outperforms pre-training with some supervised training datasets. In action recognition, improved performance can be found by combining existing supervised image-based representations with our unsupervised representation. By training on action videos with humans, our approach learns a representation sensitive to human pose. Remarkably, when applied to pose estimation, our representation is competitive with pre-training on significantly larger supervised training datasets [16].

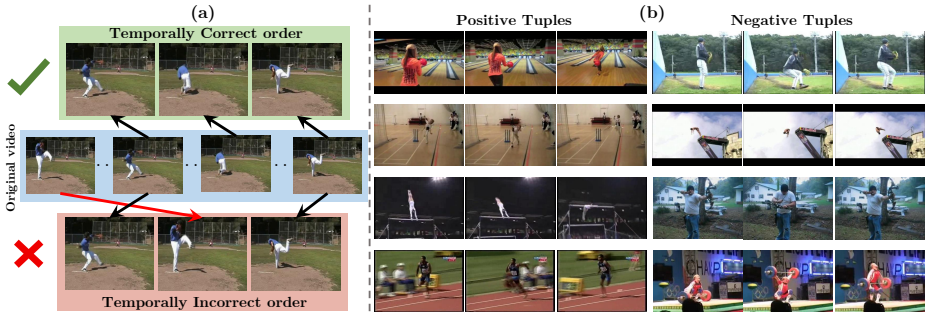

Fig. 1: **(a)** A video imposes a natural temporal structure for visual data. In many cases, one can easily verify whether frames are in the correct temporal order (shuffled or not). Such a simple sequential verification task captures important spatiotemporal signals in videos. We use this task for unsupervised pre-training of a Convolutional Neural Network (CNN). **(b)** Some examples of the automatically extracted positive and negative tuples used to formulate a classification task for a CNN.

2 Related Work

Our work uses unlabeled video sequences for learning representations. Since this source of supervision is 'free', our work can be viewed as a form of unsupervised learning. Unsupervised representation learning from single images is a popular area of research in computer vision. A significant body of unsupervised learning literature uses hand-crafted features and clustering based approaches to discover objects [17–19], or mid-level elements [20–24]. Deep learning methods like auto-encoders [25–27], Deep Boltzmann Machines [28], variational methods [29, 30], stacked auto-encoders [31, 32], and others [33, 34] learn representations directly from images. These methods learn a representation by estimating latent parameters that help reconstruct the data, and may regularize the learning process by priors such as sparsity [25]. Techniques in [10, 35] scale unsupervised learning to large image datasets showing its usefulness for tasks such as pedestrian detection [35] and object detection [10]. In terms of using 'context' for learning, our work is most similar to [10] which uses the spatial context in images. While these approaches are unsupervised, they do not use videos and cannot exploit the temporal structure in them. Our work is most related to work in unsupervised learning from videos [36–40]. Traditional methods in this domain utilize the spatiotemporal continuity as regularization for the learning process. Since visual appearance changes smoothly in videos, a common constraint is enforcing temporal smoothness of features [38, 40–43]. Zhang *et al.* [44], in particular, show how such constraints are useful for action recognition. Moving beyond just temporal smoothness, [37] enforces additional 'steadiness' constraints on the features so that the change of features across frames is meaningful. Our work, in contrast, does not explicitly impose any regularizations on the features. Other reconstruction-based learning approaches include that of Goroshin *et al.* [43] who use a generative model to predict video frames and Srivastava *et al.* [45] who use

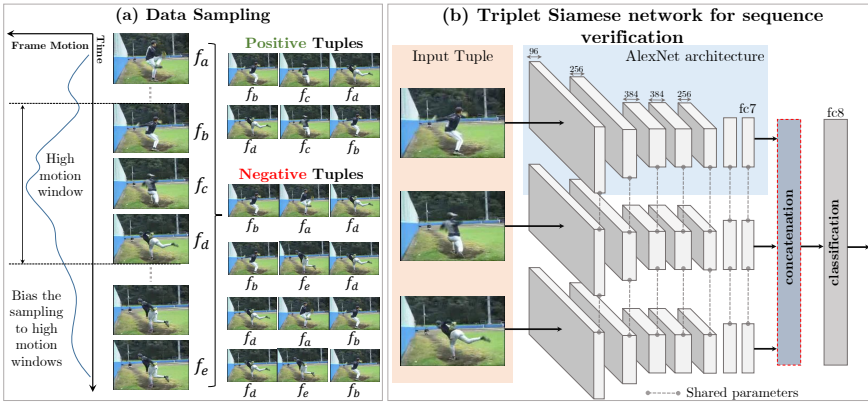

Fig. 2: **(a)** We sample tuples of frames from high motion windows in a video. We form positive and negative tuples based on whether the three input frames are in the correct temporal order. **(b)** Our triplet Siamese network architecture has three parallel network stacks with shared weights upto the `fc7` layer. Each stack takes a frame as input, and produces a representation at the `fc7` layer. The concatenated `fc7` representations are used to predict whether the input tuple is in the correct temporal order.

LSTMs [46]. Unlike our method, these works [38, 43, 45, 47] explicitly predict individual frames, but do not explore large image sizes or datasets. [48, 49] also consider the task of predicting the future from videos, but consider it as their end task and do not use it for unsupervised pre-training.

Several recent papers [36, 48, 50] use egomotion constraints from video to further constrain the learning. Jayaraman *et al.* [36] show how they can learn equivariant transforms from such constraints. Similar to our work, they use full video frames for learning with little pre-processing. Owens *et al.* [51] use audio signals from videos to learn visual representations. Another line of work [52] uses video data to mine patches which belong to the same object to learn representations useful for distinguishing objects. Typically, these approaches require significant pre-processing to create this task. While our work also uses videos, we explore them in the spirit of sequence verification for action recognition which learns from the raw video with very little pre-processing.

We demonstrate the effectiveness of our unsupervised pre-training using two extensively studied vision tasks - action recognition and pose estimation. These tasks have well established benchmark datasets [12–15]. As it is beyond the scope of this paper, we refer the reader to [53] for a survey on action recognition, and [54] for a survey on pose estimation.

3 Our Approach

Our goal is to learn a feature representation using only the raw spatiotemporal signal naturally available in videos. We learn this representation using a sequential verification task and focus on videos with human actions. Specifically, as

shown in Figure 1, we extract a tuple of frames from a video, and ask whether the frames are in the correct temporal order. In this section, we begin by motivating our use of sequential tasks and how they use the temporal structure of videos. We then describe how positive and negative tuples are sampled from videos, and describe our model.

3.1 Task motivation

When using only raw videos as input, sequential verification tasks offer a promising approach to unsupervised learning. In addition to our approach described below, several alternative tasks are explored in Section 5.2. The goal of these tasks is to encourage the model to reason about the motion and appearance of the objects, and thus learn the temporal structure of videos. Example tasks may include reasoning about the ordering of frames, or determining the relative temporal proximity of frames. For tasks that ask for the verification of temporal order, how many frames are needed to determine a correct answer? If we want to determine the correct order from just two frames, the question may be ambiguous in cases where cyclical motion is present. For example, consider a short video sequence of a person picking up a coffee cup. Given two frames the temporal order is ambiguous; the person may be picking the coffee cup up, or placing it down.

To reduce such ambiguity, we propose sampling a three frame tuple, and ask whether the tuple's frames are correctly ordered. While theoretically, three frames are not sufficient to resolve cyclical ambiguity [55], we found that combining this with smart sampling (Section 3.2) removes a significant portion of ambiguous cases. We now formalize this problem into a classification task. Consider the set of frames $\{f_1, \ldots, f_n\}$ from an unlabeled video $\mathcal{V}$. We consider the tuple (f_b, f_c, f_d) to be in the correct temporal order (class 1, positive tuple) if the frames obey either ordering $b < c < d$ or $d < c < b$, to account for the directional ambiguity in video clips. Otherwise, if $b < d < c$ or $c < b < d$, we say that the frames are not in the correct temporal order (class 0, negative tuple).

3.2 Tuple sampling

A critical challenge when training a network on the three-tuple ordering task is how to sample positive and negative training instances. A naive method may sample the tuples uniformly from a video. However, in temporal windows with very little motion it is hard to distinguish between a positive and a negative tuple, resulting in many ambiguous training examples. Instead, we only sample tuples from temporal windows with high motion. As Figure 2 shows, we use coarse frame level optical flow [56] as a proxy to measure the motion between frames. We treat the average flow magnitude per-frame as a weight for that frame, and use it to bias our sampling towards high motion windows. This ensures that the classification of the tuples is not ambiguous. Figure 1 (b) shows examples of such tuples.

To create positive and negative tuples, we sample five frames $(f_a, f_b, f_c, f_d, f_e)$ from a temporal window such that $a < b < c < d < e$ (see Figure 2 (a)). Positive instances are created using (f_b, f_c, f_d), while negative instances are created using (f_b, f_a, f_d) and (f_b, f_e, f_d). Additional training examples are also created by inverting the order of all training instances, *eg.*, (f_d, f_c, f_b) is positive. During training it is critical to use the same beginning frame f_b and ending frame f_d while only changing the middle frame for both positive and negative examples. Since only the middle frame changes between training examples, the network is encouraged to focus on this signal to learn the subtle difference between positives and negatives, rather than irrelevant features.

To avoid sampling ambiguous negative frames f_a and f_e, we enforce that the appearance of the positive f_c frame is not too similar (measured by SSD on RGB pixel values) to f_a or f_e. These simple conditions eliminated most ambiguous examples. We provide further analysis of sampling data in Section 4.1.

3.3 Model Parametrization and Learning

To learn a feature representation from the tuple ordering task, we use a simple triplet Siamese network. This network has three parallel stacks of layers with shared parameters (Figure 2). Every network stack follows the standard CaffeNet [57] (a slight modification of AlexNet [58]) architecture from the `conv1` to the `fc7` layer. Each stack takes as input one of the frames from the tuple and produces a representation at the `fc7` layer. The three `fc7` outputs are concatenated as input to a linear classification layer. The classification layer can reason about all three frames at once and predict whether they are in order or not (two class classification). Since the layers from `conv1` to `fc7` are shared across the network stacks, the Siamese architecture has the same number of parameters as AlexNet barring the final `fc8` layer. We update the parameters of the network by minimizing the regularized cross-entropy loss of the predictions on each tuple. While this network takes three inputs at training time, during testing we can obtain the `conv1` to `fc7` representations of a single input frame by using just one stack, as the parameters across the three stacks are shared.

4 Empirical ablation analysis

In this section (and in the Appendix), we present experiments to analyze the various design decisions for training our network. In Sections 5 and 6, we provide results on both action recognition and pose estimation.

Dataset: We report all our results using split 1 of the benchmark UCF101 [12] dataset. This dataset contains videos for 101 action categories with $\sim$ 9.5k videos for training and $\sim$ 3.5k videos for testing. Each video has an associated action category label. The standard performance metric for action recognition on this dataset is classification accuracy.

Details for unsupervised pre-training: For unsupervised pre-training, we do not use the semantic action labels. We sample about 900k tuples from the

Table 1: We study the effect of our design choices such as temporal sampling parameters, and varying class ratios for unsupervised pre-training. We measure the tuple prediction accuracy on a held out set from UCF101. We also show action classification results after finetuning the models on the UCF101 action recognition task (split 1).

(a) Varying temporal sampling

$\tau_{\max}$	$\tau_{\min}$	Tuple Pred.	Action Recog.
30	15	60.2	47.2
60	15	**72.1**	**50.9**
60	60	64.3	49.1

(b) Varying class ratios

Class Ratio		Tuple Pred.	Action Recog.
Neg	Pos		
0.5	0.5	52.1	38.1
0.65	0.35	68.5	45.5
0.75	0.25	**72.1**	**50.9**
0.85	0.15	67.7	48.6

UCF101 training videos. We randomly initialize our network, and train for 100k iterations with a fixed learning rate of 10^{-3} and mini-batch size of 128 tuples. Each tuple consists of 3 frames. Using more (4, 5) frames per tuple did not show significant improvement. We use batch normalization [59].

Details for Action Recognition: The spatial network from [60] is a well-established method of action recognition that uses only RGB appearance information. The parameters of the spatial network are initialized with our unsupervised pre-trained network. We use the provided action labels per video and follow the training and testing protocol as suggested in[60, 61]. Briefly, for training we form mini-batches by sampling random frames from videos. At test time, 25 frames are uniformly sampled from each video. Each frame is used to generate 10 inputs after fixed cropping and flipping (5 crops $\times$ 2 flips), and the prediction for the video is an average of the predictions across these 25×10 inputs. We use the CaffeNet architecture for its speed and efficiency. We initialize the network parameters up to the `fc7` layer using the parameters from the unsupervised pre-trained network, and initialize a new `fc8` layer for the action recognition task. We finetune the network following [60] for 20k iterations with a batch size of 256, and learning rate of 10^{-2} decaying by 10 after 14k iterations, using SGD with momentum of 0.9, and dropout of 0.5. While [60] used the wider VGG-M-2048 [62] architecture, we found that their parameters transfer to CaffeNet because of the similarities in their architectures.

4.1 Sampling of data

In this section we study the impact of sampling parameters described in Section 3.2 on the unsupervised pre-training task. We denote the maximum distance between frames of positive tuples by $\tau_{\max} = |b - d|$. This parameter controls the 'difficulty' of positives: a very high value makes it difficult to see correspondence across the positive tuple, and a very low value gives almost identical frames and thus very easy positives. Similarly, we compute the minimum distance between the frames f_a and f_e used for negative tuples to the other frames by

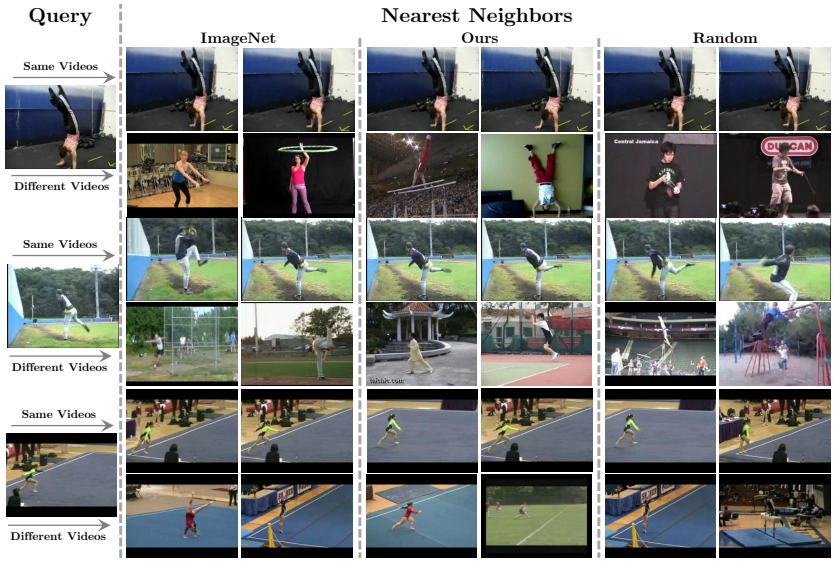

Fig. 3: We compute nearest neighbors using `fc7` features on the UCF101 dataset. We compare these results across three networks: pre-trained on ImageNet, pre-trained on our unsupervised task and a randomly initialized network. We choose a input query frame from a clip and retrieve results from other clips in the dataset. Since the dataset contains multiple clips from the same video we get near duplicate retrievals (first row). We remove these duplicates, and display results in the second row. While ImageNet focuses on the high level semantics, our network captures the human pose.

$\tau_{\text{min}} = \min(|a-b|, |d-e|)$. This parameter controls the difficulty of negatives with a low value making them harder, and a high value making them easier.

We compute the training and testing accuracy of these networks on the tuple prediction task on held out videos. This held out set is a union of samples using all the temporal sampling parameters. We show results in Table 1 (a). We also use these networks for finetuning on the UCF101 action recognition task. Our results show that the tuple prediction accuracy and the performance on the action recognition task are correlated. A large temporal window for positive sampling improves over a smaller temporal window (Rows 1 and 2), while a large window for negative sampling hurts performance (Rows 2 and 3).

4.2 Class ratios in mini-batch

Another important factor when training the model is the class ratios in each mini-batch. As has been observed empirically [63, 64], a good class ratio per mini-batch ensures that the model does not overfit to one particular class, and helps the learning process. For these experiments, we choose a single temporal window for sampling and vary only the ratio of positive and negative tuples per mini-batch. We compare the accuracy of these networks on the tuple prediction

task on held out videos in Table 1 (b). Additionally, we report the accuracy of these networks after finetuning on the action recognition task. These results show that the class ratio used for unsupervised pre-training can significantly impact learning. It is important to have a larger percentage of negative examples.

4.3 What does the temporal ordering task capture?

Nearest Neighbor retrieval We retrieve nearest neighbors using our unsupervised features on the UCF101 dataset and compare them in Figure 3 to retrievals by the pre-trained ImageNet features, and a randomly initialized network. Additional examples are shown in the supplementary materials. We pick an input query frame from a clip and retrieve neighbors from other clips in the UCF101 dataset. Since the UCF101 dataset has clips from the same video, the first set of retrievals (after removing frames from the same input clip) are near duplicates which are not very informative (notice the random network's results). We remove these near-duplicates by computing the sum of squared distances (SSD) between the frames, and display the top results in the second row of each query. These results make two things clear: 1) the ImageNet pre-trained network focuses on scene semantics 2) Our unsupervised pre-trained network focuses on the pose of the person. This would seem to indicate that the information captured by our unsupervised pre-training is complementary to that of ImageNet. Such behavior is not surprising, if we consider our network was trained without semantic labels, and must reason about spatiotemporal signals for the tuple verification task.

Visualizing `pool5` unit responses We analyze the feature representation of the unsupervised network trained using the tuple prediction task on UCF101. Following the procedure of [65] we show the top regions for `pool5` units alongwith their receptive field in Figure 4. This gives us insight into the network's internal feature representation and shows that many units show preference for human body parts and pose. This is not surprising given that our network is trained on videos of human action recognition, and must reason about human movements for the tuple ordering task.

5 Additional Experiments on Action Recognition

The previous experiments show that the unsupervised task learns a meaningful representation. In this section we compare our unsupervised method against existing baseline methods and present more quantitative results. We organize our experiments as follows: 1) Comparing our unsupervised method to learning from random initialization. 2) Exploring other unsupervised baselines and comparing our method with them. 3) Combining our unsupervised representation learning method with a supervised image representation. Additional experiments are in the supplementary material. We now describe the common experimental setup. **Datasets and Evaluation:** We use the UCF101 [12] dataset which was also used for our ablation analysis in Section 4 and measure accuracy on the 101

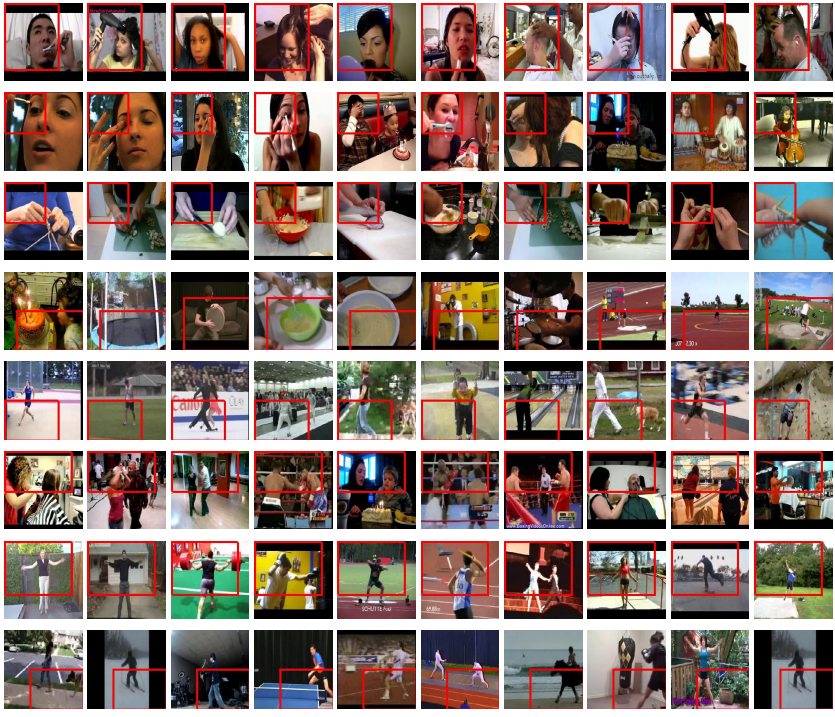

Fig. 4: In each row we display the top image regions for a unit from the `pool5` layer. We follow the method in [65] and display the receptive fields (marked in red boxes) for these units. As our network is trained on human action recognition videos, many units show preference for human body parts and pose.

action classification task. Additionally, we use the HMDB51 [13] dataset for action recognition. This dataset contains 3 splits for train/test, each with about 3.4k videos for train and 1.4k videos for testing. Each video belongs to one of 51 action categories, and performance is evaluated by measuring classification accuracy. We follow the same train/test protocols for both UCF101 and HMDB51 as described in Section 4. Note that the UCF101 dataset is about 2.5× larger than the HMDB51 dataset.

Implementation details for pre-training: We use tuples sampled using $\tau_{\max} = 60$ and $\tau_{\min} = 15$ as described in Section 4. The class ratio of positive examples per mini-batch is 25%. The other parameters for training/finetuning are kept unchanged from Section 4.

Action recognition details: As in Section 4, we use the CaffeNet architecture and the parameters from [60] for both training from scratch and finetuning. We described the finetuning parameters in Section 4. For training from random initialization (or 'scratch'), we train for 80k iterations with an initial learning rate of 10^{-2}, decaying by a factor of 10 at steps 50k and 70k. The other training

Table 2: Mean classification accuracies over the 3 splits of UCF101 and HMDB51 datasets. We compare different initializations and finetune them for action recognition.

Dataset	Initialization	Mean Accuracy
UCF101	Random	38.6
	(Ours) Tuple verification	**50.2**
HMDB51	Random	13.3
	UCF Supervised	15.2
	(Ours) Tuple verification	**18.1**

parameters (momentum, batch size etc.) are kept the same as in finetuning. We use the improved data augmentation scheme (different aspect-ratio, fixed crops) from [61] for all our methods and baselines. Note that we train or finetune all the layers of the network for all methods, including ours.

5.1 Unsupervised pre-training or random initialization?

In these experiments we study the advantage of unsupervised pre-training for action recognition in comparison to learning without any pre-training. We use our tuple prediction task to train a network starting from random initialization on the train split of UCF101. The unsupervised pre-trained network is fine-tuned on both the UCF101 and HMDB51 datasets for action recognition and compared against learning from scratch (without pre-training). We report the performance in Table 2. Our unsupervised pre-training shows a dramatic **improvement of +12.4**% over training from scratch in UCF101 and a significant gain of +4.7% in HMDB51. This impressive gain demonstrates the informativeness of the unsupervised tuple verification task. On HMDB51, we additionally finetune a network which was trained from scratch on UCF101 and report its performance in Table 2 indicated by 'UCF supervised'. We see that this network performs worse than our unsupervised pre-trained network. The UCF101 and HMDB51 have only 23 action classes in common [60] and we hypothesize that the poor performance is due to the scratch UCF101 network being unable to generalize to actions from HMDB51. For reference, a model pre-trained on the supervised ImageNet dataset [16, 66] and finetuned on UCF101 gives 67.1% accuracy, and ImageNet finetuned on HMDB51 gives an accuracy of 28.5%.

5.2 Unsupervised Baselines

In this section, we enumerate a variety of alternative verification tasks that use only video frames and their temporal ordering. For each task, we use a similar frame sampling procedure to the one described in Section 4.1. We compare their performance after finetuning them on the task of action recognition. A more informative task should serve as a better task for pre-training.

Two Close: In this task two frames (f_b, f_d) (with high motion) are considered to be temporally close if $|b - d| < \tau$ for a fixed temporal window $\tau = 30$.

Table 3: We compare the unsupervised methods defined in Section 5.2 by finetuning on the UCF101 and HMDB51 Action recognition (split 1 for both). Method with * was not pre-trained on action data.

Unsup Method →	Two Close	Two Order	DrLim [40]	TempCoh [38]	Three Order (Ours)	Obj. Patch* [52]
Acc. UCF101	42.3	44.1	45.7	45.4	**50.9**	40.7
Acc. HMDB51	15.0	16.4	16.3	15.9	**19.8**	15.6

Two Order: Two frames (f_b, f_d) are considered to be correct if $b < d$. Otherwise they are considered incorrect. $|b - d| < 30$.
Three Order: This is the original temporal ordering task we proposed in Section 3.1. We consider the 3-tuple (f_b, f_c, f_d) to be correct only if the frames obey either ordering $b < c < d$ or $b > c > d$.

We also compare against standard baselines for unsupervised learning from video.
DrLim [40]: As Equation 1 shows, this method enforces temporal smoothness over the learned features by minimizing the l_2 distance d between representations (`fc7`) of nearby frames f_b, f_d (positive class or $c = 1$), while requiring frames that are not close (negative class or $c = 0$) to be separated by a margin δ. We use the same samples as in the 'Two Close' baseline, and set $\delta = 1.0$ [38].

$$L(f_b, f_d) = \mathbb{1}(c = 1)d(f_b, f_d) + \mathbb{1}(c = 0)\max(\delta - d(f_b, f_d), 0) \quad (1)$$

TempCoh [38]: Similar to the DrLim method, temporal coherence learns representations from video by using the l_1 distance for pairs of frames rather than the l_2 distance of DrLim.
Obj. Patch [52]: We use their publicly available model which was unsupervised pre-trained on videos of objects. As their patch-mining code is not available, we do not do unsupervised pre-training on UCF101 for their model.

All these methods (except [52]) are pre-trained on training split 1 of UCF101 without action labels, and then finetuned on test split 1 of UCF101 actions and HMDB51 actions. We compare them in Table 3. Scratch performance for test split 1 of UCF101 and HMDB51 is 39.1% and 14.8% respectively. The tuple verification task outperforms other sequential ordering tasks, and the standard baselines by a significant margin. We attribute the low number of [52] to the fact that they focus on object detection on a very different set of videos, and thus do not perform well on action recognition.

5.3 Combining unsupervised and supervised pre-training

We have thus far seen that unsupervised pre-training gives a significant performance boost over training from random initialization. We now see if our pre-training can help improve existing image representations. Specifically, we initialize our model using the weights from the ImageNet pre-trained model and

Table 4: Results of using our unsupervised pre-training to adapt existing image representations trained on ImageNet. We use unsupervised data from training split 1 of UCF101, and show the mean accuracy (3 splits) by finetuning on HMDB51.

Initialization	Mean Accuracy
Random	13.3
(Ours) Tuple verification	18.1
UCF sup.	15.2
ImageNet	28.5
(Ours) ImageNet + Tuple verification	**29.9**
ImageNet + UCF sup.	30.6

use it for the tuple-prediction task on UCF101 by finetuning for 10k iterations. We hypothesize this may add complementary information to the ImageNet representation. To test this, we finetune this model on the HMDB51 [13] action recognition task. We compare this performance to finetuning on HMDB51 without the tuple-prediction task. Table 4 shows these results.

Our results show that combining our pre-training with ImageNet helps improve the accuracy of the model (rows 3, 4). Finally, we compare against using multiple sources of supervised data: initialized using the ImageNet weights, finetuned on UCF101 action recognition and then finetuned on HMDB51 (row 5). The accuracy using all sources of supervised data is only slightly better than the performance of our model (rows 4, 5). This demonstrates the effectiveness of our simple yet powerful unsupervised pre-training.

6 Pose Estimation Experiments

The qualitative results from Sec 4.3 suggest that our network captures information about human pose. To evaluate this quantitatively, we conduct experiments on the task of pose estimation using keypoint prediction.

Datasets and Metrics: We use the FLIC (full) [14] and the MPII [15] datasets. For FLIC, we consider 7 keypoints on the torso: head, 2× (shoulders, elbows, wrists). We compute the keypoint for the head as an average of the keypoints for the eyes and nose. We evaluate the Probability of Correct Keypoints (PCK) measure [67] for the keypoints. For MPII, we use all the keypoints on the full body and report the PCKh@0.5 metric as is standard for this dataset.

Model training: We use the CaffeNet architecture to regress to the keypoints. We follow the training procedure in [68][3]. For FLIC, we use a train/test split of 17k and 3k images respectively and finetune models for 100k iterations. For MPII, we use a train/test split of 18k and 2k images. We use a batch size of 32, learning rate of 5×10^{-4} with AdaGrad [69] and minimize the Euclidean loss

[3] Public re-implementation from https://github.com/mitmul/deeppose

Table 5: Pose estimation results on the FLIC and MPII datasets.

Init.	PCK for FLIC						PCKh@0.5 for MPII		
	wri	elb	sho	head	Mean	AUC	Upper	Full	AUC
Random Init.	53.0	75.2	86.7	91.7	74.5	36.1	76.1	72.9	34.0
Tuple Verif.	**69.6**	**85.5**	**92.8**	**97.4**	**84.7**	**49.6**	**87.7**	**85.8**	**47.6**
Obj. Patch[52]	58.2	77.8	88.4	94.8	77.1	42.1	84.3	82.8	43.8
DrLim[40]	37.8	68.4	80.4	83.4	65.2	27.9	84.3	81.5	41.5
UCF Sup.	61.0	78.8	89.1	93.8	78.8	42.0	86.9	84.6	45.5
ImageNet	69.6	86.7	93.6	97.9	85.8	51.3	85.1	83.5	47.2
ImageNet + Tuple	**69.7**	**87.1**	**93.8**	**98.1**	**86.2**	**52.5**	**87.6**	**86.0**	**49.5**

(l_2 distance between ground truth and predicted keypoints). For training from scratch (Random Init.), we use a learning rate of 5×10^{-4} for 1.3M iterations.
Methods: Following the setup in Sec 5.1, we compare against various initializations of the network. We consider two supervised initalizations - from pre-training on ImageNet and UCF101. We consider three unsupervised initializations - our tuple based method, DrLim [40] on UCF101, and the method of [52]. We also combine our unsupervised initialization with ImageNet pre-training.

Our results for pose estimation are summarized in Table 5. Our unsupervised pre-training method outperforms the fully supervised UCF network (Sec 5.1) by +7.6% on FLIC and +2.1% on MPII. Our method is also competitive with ImageNet pre-training on both these datasets. Our unsupervised pre-training is complementary to ImageNet pre-training, and can improve results after being combined with it. This supports the qualitative results from Sec 4.3 that show our method can learn human pose information from unsupervised videos.

7 Discussion

In this paper, we studied unsupervised learning from the raw spatiotemporal signal in videos. Our proposed method outperforms other existing unsupervised methods and is competitive with supervised methods. A next step to our work is to explore different types of videos and use other 'free' signals such as optical flow. Another direction is to use a combination of CNNs and RNNs, and to extend our tuple verification task to much longer sequences. We believe combining this with semi-supervised methods [70, 71] is a promising future direction.

Acknowledgments: The authors thank Pushmeet Kohli, Ross Girshick, Abhinav Shrivastava and Saurabh Gupta for helpful discussions. Ed Walter for his timely help with the systems. This work was supported in part by ONR MURI N000141612007 and the US Army Research Laboratory (ARL) under the CTA program (Agreement W911NF-10-2-0016). We gratefully acknowledge the hardware donation by NVIDIA.

References

[1] Cleeremans, A., McClelland, J.L.: Learning the structure of event sequences. Journal of Experimental Psychology: General **120**(3) (1991) 235 1

[2] Reber, A.S.: Implicit learning and tacit knowledge. Journal of experimental psychology: General **118**(3) (1989) 219

[3] Cleeremans, A.: Mechanisms of implicit learning: Connectionist models of sequence processing. MIT press (1993) 1

[4] Sun, R., Merrill, E., Peterson, T.: From implicit skills to explicit knowledge: A bottom-up model of skill learning. Cognitive science **25**(2) (2001) 1

[5] Baker, R., Dexter, M., Hardwicke, T.E., Goldstone, A., Kourtzi, Z.: Learning to predict: exposure to temporal sequences facilitates prediction of future events. Vision research **99** (2014) 124–133 1

[6] Sun, R., Giles, C.L.: Sequence learning: from recognition and prediction to sequential decision making. IEEE Intelligent Systems **16**(4) (2001) 2

[7] Mikolov, T., Chen, K., Corrado, G., Dean, J.: Efficient estimation of word representations in vector space. arXiv preprint arXiv:1301.3781 (2013) 2

[8] Mikolov, T., Sutskever, I., Chen, K., Corrado, G.S., Dean, J.: Distributed representations of words and phrases and their compositionality. In: NIPS. (2013) 2

[9] Firth, J.R.: A synopsis of linguistic theory 1930-1955. (1957) 2

[10] Doersch, C., Gupta, A., Efros, A.A.: Unsupervised visual representation learning by context prediction. In: ICCV. (2015) 2, 3, 20

[11] LeCun, Y., Boser, B., Denker, J.S., Henderson, D., Howard, R.E., Hubbard, W., Jackel, L.D.: Backpropagation applied to handwritten zip code recognition. Neural computation **1** (1989) 2

[12] Soomro, K., Zamir, A.R., Shah, M.: UCF101: A dataset of 101 human actions classes from videos in the wild. arXiv preprint arXiv:1212.0402 (2012) 2, 4, 6, 9

[13] Kuehne, H., Jhuang, H., Garrote, E., Poggio, T., Serre, T.: HMDB: a large video database for human motion recognition. In: ICCV. (2011) 2, 10, 13

[14] Sapp, B., Taskar, B.: Modec: Multimodal decomposable models for human pose estimation. In: CVPR. (2013) 2, 13

[15] Andriluka, M., Pishchulin, L., Gehler, P., Schiele, B.: 2d human pose estimation: New benchmark and state of the art analysis. In: CVPR. (June 2014) 2, 4, 13

[16] Deng, J., Dong, W., Socher, R., jia Li, L., Li, K., Fei-fei, L.: Imagenet: A large-scale hierarchical image database. In: CVPR. (2009) 2, 11, 20

[17] Faktor, A., Irani, M.: clustering by composition–unsupervised discovery of image categories. In: ECCV. (2012) 3

[18] Sivic, J., Russell, B.C., Efros, A.A., Zisserman, A., Freeman, W.T.: Discovering objects and their location in images. In: ICCV. (2005)

[19] Russell, B.C., Freeman, W.T., Efros, A.A., Sivic, J., Zisserman, A.: Using multiple segmentations to discover objects and their extent in image collections. In: CVPR. (2006) 3

[20] Singh, S., Gupta, A., Efros, A.: Unsupervised discovery of mid-level discriminative patches. ECCV (2012) 3
[21] Juneja, M., Vedaldi, A., Jawahar, C., Zisserman, A.: Blocks that shout: Distinctive parts for scene classification. In: CVPR. (2013)
[22] Doersch, C., Gupta, A., Efros, A.A.: Mid-level visual element discovery as discriminative mode seeking. In: NIPS. (2013)
[23] Li, Q., Wu, J., Tu, Z.: Harvesting mid-level visual concepts from large-scale internet images. In: CVPR. (2013)
[24] Sun, J., Ponce, J.: Learning discriminative part detectors for image classification and cosegmentation. In: ICCV. (2013) 3
[25] Olshausen, B.A., et al.: Emergence of simple-cell receptive field properties by learning a sparse code for natural images. Nature **381**(6583) (1996) 3
[26] Bengio, Y., Thibodeau-Laufer, E., Alain, G., Yosinski, J.: Deep generative stochastic networks trainable by backprop. arXiv preprint arXiv:1306.1091 (2013)
[27] Vincent, P., Larochelle, H., Bengio, Y., Manzagol, P.A.: Extracting and composing robust features with denoising autoencoders. In: ICML. (2008) 3
[28] Salakhutdinov, R., Hinton, G.E.: Deep boltzmann machines. In: ICAIS. (2009) 3
[29] Kingma, D.P., Welling, M.: Auto-encoding variational bayes. arXiv preprint arXiv:1312.6114 (2013) 3
[30] Rezende, D.J., Mohamed, S., Wierstra, D.: Stochastic backpropagation and approximate inference in deep generative models. arXiv preprint arXiv:1401.4082 (2014) 3
[31] Lee, H., Battle, A., Raina, R., Ng, A.Y.: Efficient sparse coding algorithms. In: NIPS. (2006) 3
[32] Bengio, Y., Lamblin, P., Popovici, D., Larochelle, H., et al.: Greedy layer-wise training of deep networks. NIPS (2007) 3
[33] Le, Q.V.: Building high-level features using large scale unsupervised learning. In: ICASSP. (2013) 3
[34] Wang, X., Gupta, A.: Generative image modeling using style and structure adversarial networks. In: ECCV. (2016) 3
[35] Sermanet, P., Kavukcuoglu, K., Chintala, S., LeCun, Y.: Pedestrian detection with unsupervised multi-stage feature learning. In: CVPR. (2013) 3
[36] Jayaraman, D., Grauman, K.: Learning image representations equivariant to ego-motion. ICCV (2015) 3, 4
[37] Jayaraman, D., Grauman, K.: Slow and steady feature analysis: Higher order temporal coherence in video. arXiv preprint arXiv:1506.04714 (2015) 3
[38] Mobahi, H., Collobert, R., Weston, J.: Deep learning from temporal coherence in video. In: ICML. (2009) 3, 4, 12
[39] Isola, P., Zoran, D., Krishnan, D., Adelson, E.H.: Learning visual groups from co-occurrences in space and time. arXiv preprint arXiv:1511.06811 (2015)

[40] Hadsell, R., Chopra, S., LeCun, Y.: Dimensionality reduction by learning an invariant mapping. In: CVPR, IEEE (2006) 3, 12, 14
[41] Földiák, P.: Learning invariance from transformation sequences. Neural Computation **3**(2) (1991)
[42] Wiskott, L., Sejnowski, T.J.: Slow feature analysis: Unsupervised learning of invariances. Neural computation **14**(4) (2002)
[43] Goroshin, R., Bruna, J., Tompson, J., Eigen, D., LeCun, Y.: Unsupervised learning of spatiotemporally coherent metrics. In: ICCV. (2015) 3, 4
[44] Zhang, Z., Tao, D.: Slow feature analysis for human action recognition. TPAMI **34**(3) (2012) 3
[45] Srivastava, N., Mansimov, E., Salakhutdinov, R.: Unsupervised learning of video representations using lstms. arXiv preprint arXiv:1502.04681 (2015) 3, 4
[46] Hochreiter, S., Schmidhuber, J.: Long short-term memory. Neural computation **9** (1997) 4
[47] Taylor, G.W., Fergus, R., LeCun, Y., Bregler, C.: Convolutional learning of spatio-temporal features. In: ECCV. (2010) 4
[48] Zhou, Y., Berg, T.L.: Temporal perception and prediction in ego-centric video. In: ICCV. (2015) 4
[49] Vondrick, C., Pirsiavash, H., Torralba, A.: Anticipating the future by watching unlabeled video. arXiv preprint arXiv:1504.08023 (2015) 4
[50] Agrawal, P., Carreira, J., Malik, J.: Learning to see by moving. In: ICCV. (2015) 4
[51] Owens, A., Isola, P., McDermott, J., Torralba, A., Adelson, E.H., Freeman, W.T.: Visually indicated sounds. In: CVPR. (2016) 4
[52] Wang, X., Gupta, A.: Unsupervised learning of visual representations using videos. In: ICCV. (2015) 4, 12, 14
[53] Poppe, R.: A survey on vision-based human action recognition. Image and vision computing **28**(6) (2010) 4
[54] Perez-Sala, X., Escalera, S., Angulo, C., Gonzalez, J.: A survey on model based approaches for 2d and 3d visual human pose recovery. Sensors **14** (2014) 4
[55] Shannon, C.E.: Communication in the presence of noise. Proceedings of the IRE **37** (1949) 5
[56] Farnebäck, G.: Two-frame motion estimation based on polynomial expansion. In: Image analysis. Springer (2003) 5
[57] Jia, Y., Shelhamer, E., Donahue, J., Karayev, S., Long, J., Girshick, R., Guadarrama, S., Darrell, T.: Caffe: Convolutional architecture for fast feature embedding. In: ACMM. (2014) 6
[58] Krizhevsky, A., Sutskever, I., Hinton, G.E.: Imagenet classification with deep convolutional neural networks. In: NIPS. (2012) 6
[59] Ioffe, S., Szegedy, C.: Batch normalization: Accelerating deep network training by reducing internal covariate shift. arXiv preprint arXiv:1502.03167 (2015) 7, 20
[60] Simonyan, K., Zisserman, A.: Two-stream convolutional networks for action recognition in videos. In: NIPS. (2014) 7, 10, 11, 20

[61] Wang, L., Xiong, Y., Wang, Z., Qiao, Y.: Towards good practices for very deep two-stream convnets. arXiv preprint arXiv:1507.02159 (2015) 7, 11
[62] Chatfield, K., Simonyan, K., Vedaldi, A., Zisserman, A.: Return of the devil in the details: Delving deep into convolutional nets. In: BMVC. (2014) 7
[63] Girshick, R.: Fast R-CNN. In: ICCV. (2015) 8
[64] Ren, S., He, K., Girshick, R., Sun, J.: Faster r-cnn: Towards real-time object detection with region proposal networks. In: NIPS. (2015) 8
[65] Girshick, R., Donahue, J., Darrell, T., Malik, J.: Rich feature hierarchies for accurate object detection and semantic segmentation. In: CVPR. (2014) 9, 10
[66] Russakovsky, O., Deng, J., Su, H., Krause, J., Satheesh, S., Ma, S., Huang, Z., Karpathy, A., Khosla, A., Bernstein, M., Berg, A.C., Fei-Fei, L.: ImageNet Large Scale Visual Recognition Challenge. IJCV **115** (2015) 11, 20
[67] Yang, Y., Ramanan, D.: Articulated human detection with flexible mixtures of parts. TPAMI **35** (2013) 13, 20
[68] Toshev, A., Szegedy, C.: Deeppose: Human pose estimation via deep neural networks. In: CVPR. (2014) 13
[69] Duchi, J., Hazan, E., Singer, Y.: Adaptive subgradient methods for online learning and stochastic optimization. JMLR **12** (2011) 13
[70] Misra, I., Shrivastava, A., Hebert, M.: Watch and learn: Semi-supervised learning of object detectors from videos. In: CVPR. (2015) 14
[71] Liang, X., Liu, S., Wei, Y., Liu, L., Lin, L., Yan, S.: Towards computational baby learning: A weakly-supervised approach for object detection. In: ICCV. (2015) 14
[72] Pfister, T., Charles, J., Zisserman, A.: Flowing convnets for human pose estimation in videos. In: ICCV. (2015) 20

8 Appendix

We present extra results and analysis of our method.

8.1 More qualitative results

Nearest Neighbors: We provide more visualizations of nearest neighbors in Figure 5, similar to Section 4.3 of the main paper.

Fill in the blanks: Given a start and an end frame, we use our unsupervised network to find a middle frame from the input video. We compute such results on held out videos and show them in Figure 6. Our network is able to correctly predict frames that should temporally lie between a given start and end frame. For cyclical actions with large motion, *eg.*, a child on a swing (row 2), our network resolves directional ambiguity (is the swing going up or down). The last row shows failure cases which lack motion (applying makeup) or have only small moving objects (soccer ball).

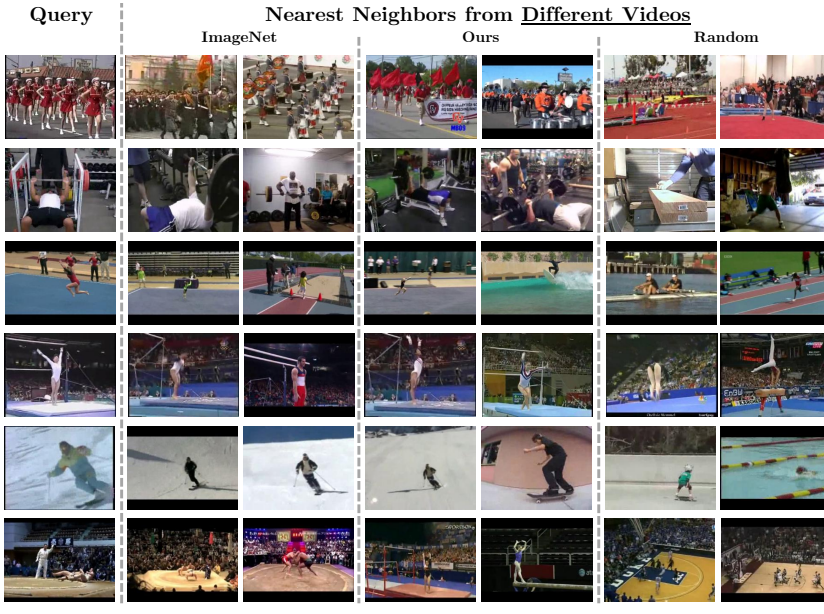

Fig. 5: We compute nearest neighbors using `fc7` features on the UCF101 dataset. We compare these results across three networks: pre-trained on ImageNet, pre-trained on our unsupervised task and a randomly initialized network. We choose a input query frame from a clip and retrieve results from other clips in the dataset. Since the dataset contains multiple clips from the same video we get near duplicate retrievals. We remove these duplicates, and display results. While ImageNet focuses on the high level semantics, our network captures the human pose.

8.2 Control experiments

In these experiments we control for certain variations like batch normalization, number of iterations etc. in our method and the baseline methods. These experiments will help us tease apart gains over the baselines that are due to these variations vs. gains due to our unsupervised pre-training.

Control for more iterations: Our unsupervised training method is trained for a larger number of iterations (without action labels). This gives it an advantage over the baselines that are trained from scratch or finetuned from other datasets (like ImageNet [16, 66]). To control for this, we run the baseline methods, both scratch and finetuning, for a higher number of iterations (200k vs. 80k for scratch, 40k vs. 20k for finetune) than in [60], and report the highest accuracy.

Control for batch normalization: We use batch normalization [59] for training our triple-Siamese network. Since the other baseline methods from [60] predate the batch normalization method, we first re-trained the baseline models using batch normalization. For lack of space, these numbers are reported in the supplementary material. As also reported by [10], we observed that using batch normalization consistently gave about 1% worse accuracy. Thus, we report baseline numbers without batch normalization.

8.3 Pose Estimation

We evaluate the Probability of Correct Keypoints (PCK) measure [67] on the wrist and elbow keypoints for the FLIC-full dataset, as used in [72] (Figure 7). PCK measures the correctness of predicted keypoints by varying the distance threshold at which they are considered correct.

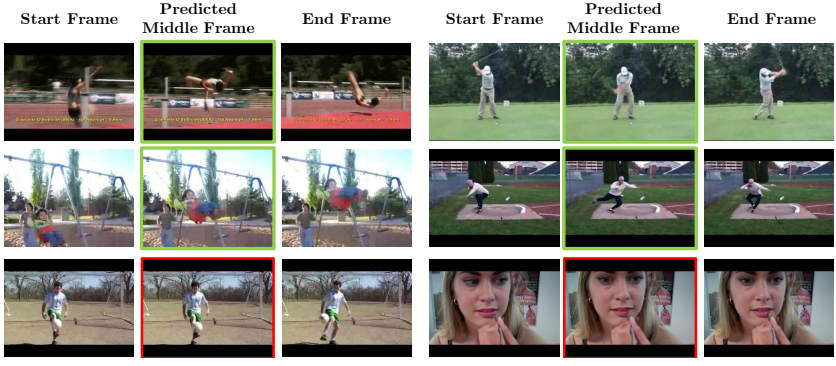

Fig. 6: Given a start and an end frame on a held out video, we use our network to find a middle frame. The first two rows demonstrate correct predictions, with the network correctly predicting motion for cyclical cases, *eg.*, a child on a swing. The last row depicts wrong predictions.

Fig. 7: PCK values for the wrist and elbow keypoints on the FLIC dataset