Memory-augmented Dense Predictive Coding for Video Representation Learning

Tengda Han[0000−0002−1874−9664], Weidi Xie[0000−0003−3804−2639], and Andrew Zisserman[0000−0002−8945−8573]

Visual Geometry Group, Department of Engineering Science, University of Oxford
{htd,weidi,az}@robots.ox.ac.uk

Abstract. The objective of this paper is self-supervised learning from video, in particular for representations for action recognition. We make the following contributions: (i) We propose a new architecture and learning framework *Memory-augmented Dense Predictive Coding* (`MemDPC`) for the task. It is trained with a *predictive attention mechanism* over the set of *compressed memories*, such that any future states can always be constructed by a convex combination of the condensed representations, allowing to make multiple hypotheses efficiently. (ii) We investigate visual-only self-supervised video representation learning from RGB frames, or from unsupervised optical flow, or both. (iii) We thoroughly evaluate the quality of the learnt representation on four different downstream tasks: action recognition, video retrieval, learning with scarce annotations, and unintentional action classification. In all cases, we demonstrate state-of-the-art or comparable performance over other approaches with orders of magnitude fewer training data.

1 Introduction

Recent advances in self-supervised representation learning for images have yielded impressive results, *e.g.* [11, 26, 27, 28, 34, 50, 57, 70], with performance matching or exceeding that of supervised representation learning on downstream tasks. However, in the case of videos, although there have been similar gains for *multi-modal* self-supervised representation learning, *e.g.* [2, 4, 39, 47, 52, 56], progress on learning *only* from the video stream (without additional audio or text streams) is lagging behind. The objective of this paper is to improve the performance of video only self-supervised learning.

Compared to still images, videos should be a more suitable source for self-supervised representation learning as they naturally provide various augmentation, such as object out of plane rotations and deformations. In addition, videos contain additional temporal information that can be used to disambiguate actions *e.g.* open vs. close. The temporal information can also act as a free supervisory signal to train a model to predict the future states from the past either

[1] Code is available at `http://www.robots.ox.ac.uk/~vgg/research/DPC`

passively by watching videos [24, 45, 59] or actively in an interactive environment [16], and thereby learn a video representation.

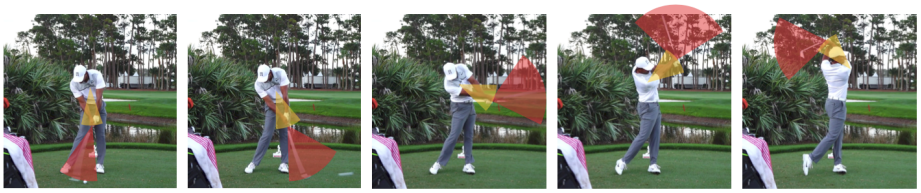

Fig. 1. Can you predict the next frame? Future prediction naturally involves challenges from multiple hypotheses, *e.g.* the motion of each leaf, reflections on the water, hands and the golf club can be in many possible positions

Unfortunately, the exact future is indeterministic (a problem long discussed in the history of science, and known as "Laplace's Demon"). As shown in Figure 1, this problem is directly apparent in the stochastic variability of scenes, *e.g.* trying to predict the exact motion of each leaf on a tree when the wind blows, or the changing reflections on the water. More concretely, consider the action of 'playing golf' – once the action starts, a future frame could have the hands and golf club in many possible positions, depending on the person who is playing. Learning visual representation by predicting the future therefore requires designing specific training schemes that simultaneously circumvents the unpredictable details in exact frames, and also handles multiple hypotheses and incomplete data – in particular only one possible future is exposed by the frames of one video.

Various approaches have been developed to deal with the multiple possible futures for an action. Vondrick *et al.* [59] explicitly generates multiple hypotheses, and only the hypothesis that is closest to the true observation is chosen during optimization, however, this approach limits the number of possible future states. Another line of work [24, 50] circumvents this difficulty by using contrastive learning – the model is only asked to predict *one* future state that assigns higher similarity to the true observation than to any distractor observation (from different videos or from elsewhere in the same video). Recalling the 'playing golf' example, the embedding must capture the hand movement for this action, but not necessarily the precise position and velocity, only sufficiently to disambiguate future frames.

In this paper, we continue the idea of contrastive learning, but improve it by the addition of a *Compressive Memory*, which maps "lifelong" experience to a set of compressed memories and helps to better anticipate the future. We make the following four contributions: *First*, we propose a novel Memory-augmented Dense Predictive Coding (`MemDPC`) architecture. It is trained with a *predictive attention mechanism* over the set of *compressed memories*, such that any future states can always be constructed by a convex combination of the condensed representations,

allowing it to make multiple hypotheses efficiently. *Second*, we investigate visual only self-supervised video representation learning from RGB frames, or from unsupervised optical flow, or both. *Third*, we argue that, in addition to the standard linear probes and fine-tuning [56,69], that have been used for evaluating representation from self-supervised learning, a non-linear probe should also be used, and demonstrate the difference that this probe makes. *Finally*, we evaluate the quality of learnt feature representation on four different downstream tasks: action recognition, learning under low-data regime (scarce annotations), video retrieval, and unintentional action classification; and demonstrate state-of-the-art performance over other approaches with similar settings on *all* tasks.

2 Related Work

Self-supervised learning for images has undergone rapid progress in visual representation learning recently [11, 26, 27, 28, 34, 50, 57, 70]. Generally speaking, the success can be attributed to one specific training paradigm, namely contrastive learning [12, 23], *i.e.* contrast the positive and negative sample pairs.

Self-supervised learning for videos has explored various ideas to learn representations by exploiting spatio-temporal information [1,2,8,15,20,21,30,31,32, 33, 35, 37, 42, 43, 44, 45, 48, 59, 60, 62, 63, 64, 66]. Of more relevance here is the line of research using contrastive learning, *e.g.* [2, 4, 5, 39, 51, 52] learn from visual-audio correspondence, [47] learns from video and narrations, and our previous work [24] learns video representations by predicting future states.

Memory models have been considered as one of the fundamental building blocks towards intelligence. In the deep learning literature, two different lines of research have received extensive attention, one is to build networks that involve an internal memory which can be implicitly updated in a recurrent manner, *e.g.* LSTM [29] and GRU [13]. The other line of research focuses on augmenting feed-forward models with an explicit memory that can be read or written to with an attention-based procedure [6, 14, 22, 41, 55, 58, 61, 67]. In this work, our compressive memory falls in the latter line, *i.e.* an external memory module.

3 Methodology

The proposed Memory-augmented Dense Predictive Coding (`MemDPC`), is a conceptually simple model for learning a video representation with contrastive predictive coding. The key novelty is to augment the previous DPC model with a *Compressive Memory*. This provides a mechanism for handling the multiple future hypotheses required in learning due to the problem that only one possible future is exposed by a particular video.

The architecture is shown in Figure 2. As in the case of DPC, the video is partitioned into 8 blocks with 5 frames each, and an encoder network f generates an embedding z for each block. For inference, these embeddings are aggregated

over time by a function g into a video level embedding c. During training, the future block embeddings $\hat{z}$ are predicted and used to select the true embedding in the dense predictive coding manner. In `MemDPC`, the prediction of $\hat{z}$ is from a convex combination of memory elements (rather from c directly as in DPC), and it is this restriction that also enables the network to handle multiple hypotheses, as will be explained below.

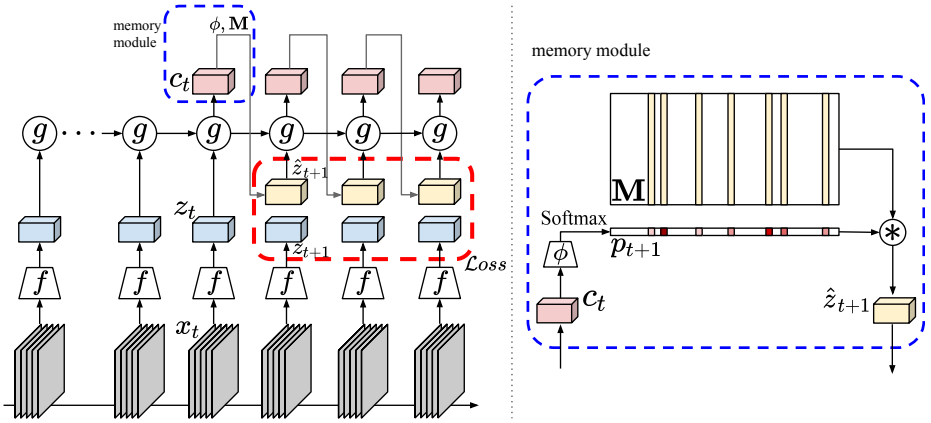

Fig. 2. Architecture of the Memory-augmented Dense Predictive Coding (`MemDPC`). Note, the memory module is only used during the self-supervised training. The c_t embedding is used for downstream tasks

3.1 Memory-augmented Dense Predictive Coding (`MemDPC`)

Video Block Encoder. As shown in Figure 2, we partition the input video sequence into multiple blocks $x_1, ..., x_t, x_{t+1}, ...$, where each block is composed of multiple video frames. Then a shared feature extractor $f(.)$ (architecture details are given in the supplementary material) extracts the video features z_i from each video block x_i:

$$z_i = f(x_i) \tag{1}$$

Temporal Aggregation. After acquiring block representations, multiple block embeddings are aggregated to obtain a context feature c_t, summarizing the information over a longer temporal window. Specifically,

$$c_t = g(z_1, z_2, ..., z_t) \tag{2}$$

in our case, we simply adopt Recurrent Neural Networks (RNNs) for $g(.)$, but other auto-regressive model should also be feasible for temporal aggregation.

Compressive Memory. In order to enable efficient multi-hypotheses estimation, we augment the predictive models with an external common compressive

memory. This external memory bank is shared for the entire dataset during training, and is accessed by a *predictive addressing mechanism* that infers a probability distribution over the memory entries, where each memory entry acts as a potential hypothesis.

In detail, the compressed memory bank is written $\mathbf{M} = [m_1, m_2, ..., m_k]^\top \in \mathbb{R}^{k \times C}$, where k is the memory size and C is the dimension of each compressed memory. During training, a predictive memory addressing mechanism (Eq. 3) is used to draw a hypothesis from the compressed memory, and the predicted future states $\hat{z}_{t+1}$ is then computed as the expectation of sampled hypotheses (Eq. 4):

$$p_{t+1} = \text{Softmax}\big(\phi(c_t)\big) \tag{3}$$

$$\hat{z}_{t+1} = \sum_{i=1}^{k} p_{(i,t+1)} \cdot m_i = p_{t+1}\mathbf{M} \tag{4}$$

where $p_{(i,t+1)} \in \mathbb{R}^k$ refers to the contribution of i-th memory slot for the future representation at time step t. A future prediction function $\phi(.)$ projects the context representation to $p_{(i,t+1)}$, in practice, $\phi(.)$ is learned with a multilayer perceptron. The softmax operation is applied on the k dimension.

Multiple Hypotheses. The dot product of the predicted and desired future pairs can be rewritten as:

$$\psi(\hat{z}^\top, z) = \left(\sum_{i=1}^{k} p_i \cdot m_i^\top \right) z = \sum_{i=1}^{k} p_i \cdot \left(m_i^\top z \right) \tag{5}$$

where $m_i^\top z$ refers to the dot product (*i.e.* similarity) between a single memory slot and the feature states from the observation. The objective of $\phi(.)$ is to predict a probability distribution over k hypotheses in the memory bank, such that the expectation of $m_i^\top z$ for a positive pair is larger than that of negative pairs. Since the future is uncertain, the desired future feature z might be similar to one of the multiple hypotheses in the memory bank, say either m_p or m_q, for instance. To handle this uncertainty, the future prediction function $\phi(.)$ just needs to put a higher probability on both the p and q slots, such that Equation 5 is always large no matter which state the future is. In this way, the burden of modelling the future uncertainly is allocated to the memory bank $\mathbf{M}$ and future prediction function $\phi(.)$, thus the backbone encoder $f(.)$ and $g(.)$ can save capacity and capture the high-level action trajectory.

Memory Mechanism Discussion. Note, in contrast to the memory mechanism in Wu *et al.* [65] and MoCo [26], which has the goal of storing more data samples to increase the number of negative samples during contrastive learning, our Compressive Memory has the goal of aiding learning by compressing all the potential hypotheses within the entire dataset, and allowing access through the predictive addressing mechanism. The memory mechanism shares similarity with NetVLAD [3], which represents a feature distribution with "trainable

centroids". However, in NetVLAD the goal is for compact and discriminative feature aggregation, and it encodes a weighted sum of *residuals* between feature vectors and the centroids. In contrast, our goal with $\phi(.)$ is to predict attention weights for the entries in the memory bank $\mathbf{M}$, in order to construct the future state as a convex combination these entries. The model can also sequentially predict further into the future with the *same* memory bank.

3.2 Contrastive Learning

Contrastive Learning generally refers to the training paradigm that forces the similarity scores of positive pairs to be higher than those of negatives.

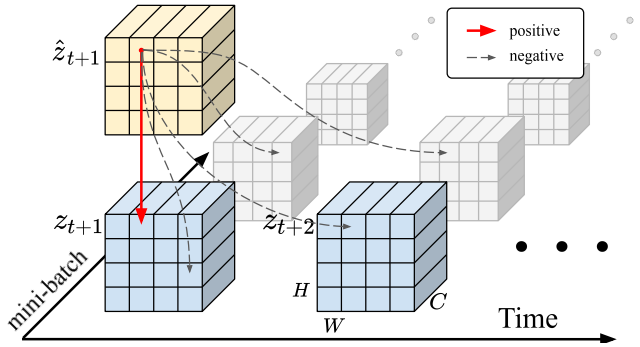

Fig. 3. Details of the contrastive loss. Contrastive loss is computed densely, *i.e.* over both spatial and temporal dimension of the feature

Specifically, in `MemDPC`, we predict the future states recursively, ending up with a sequence of predicted features $\hat{z}_{t+1}, \hat{z}_{t+2}, \ldots, \hat{z}_{end}$ and the video feature from the true observations $z_{t+1}, z_{t+2}, \ldots, z_{end}$. As shown in Figure 3, each predicted $\hat{z}$ in practise is a dense feature map. To simplify the notation, we denote temporal index with i and denote other indexes including spatial index and batch-wise index as k, where batch-wise index means the index in the current mini-batch, $k \in \{(1,1,1),(1,1,2),...,(B,H,W)\}$. The objective function to minimize becomes:

$$\mathcal{L} = -\mathbb{E}\left[\sum_{i,k} \log \frac{e^{\psi(\hat{z}_{i,k}^{\top}, z_{i,k})}}{e^{\psi(\hat{z}_{i,k}^{\top}, z_{i,k})} + \sum_{(j,m)\neq(i,k)} e^{\psi(\hat{z}_{i,k}^{\top}, z_{j,m})}}\right] \tag{6}$$

where $\psi(\cdot)$ is acting as a *critic* function, in our case, we simply use dot product between the two vectors (we also experiment with L2-normalization, and find it gives similar performance on downstream tasks). Essentially, the objective function acts as a multi-way classifier, and the goal of optimization is to learn the video block encoder that assigns the highest values for $(\hat{z}_{i,k}, z_{i,k})$ *i.e.* higher

similarity between the predicted future states and that from true observations originating from the same video and spatial-temporal aligned position.

3.3 Improving Performance by Extensions

As `MemDPC` is a general self-supervised learning framework, it can be combined with other 'modules' like two-stream networks and bi-directional RNN to improve the quality of the visual representations.

Two-stream Architecture. We represent dense optical flow as images by stacking the x and y displacement fields and another zero-valued array to make them 3-channel images. There is no need to modify the `MemDPC` framework, and it can be directly applied to optical flow inputs by simply replacing the input x_t from RGB frames to optical flow frames. We use late fusion like [19, 53] to combine both streams.

Bi-directional `MemDPC`. From the perspective of human perception, where only the future is actively predicted, `MemDPC` is initially designed to be single-directional. However, when passively taking the videos as input, predicting backwards becomes feasible. Bi-directional `MemDPC` has a shared feature extractor $f(.)$ to extract the features $z_1, z_2, ..., z_t$, but has two identical aggregators $g^f(.)$ and $g^b(.)$ denoting forward and backward aggregation. They aggregate the bi-directional context features c_t^f and c_t^b. Then `MemDPC` predicts the past and the future features with the shared $\phi(.)$ and shared memory bank $\mathbf{M}$, and constructs contrastive losses for both directions, namely $\mathcal{L}^f$ and $\mathcal{L}^b$. The final loss is the average of the losses from both directions.

4 How to Evaluate Self-Supervised Learning?

The standard way to evaluate the quality of the learned representation is to assess the performance on downstream tasks using two protocols: (i) a linear probe – freezing the network and only train a linear head for the downstream task; or (ii) fine-tuning the entire network for the downstream task. For example, in (i) if the downstream task is classification, *e.g.* of UCF101, then a linear classifier is trained on top of the frozen base network. In (ii) the self-supervised training of the base network only provides the initialization. However, there is no particular reason why self-supervision should lead to features that are linearly separable, even if the representation has encoded semantic information. Consequently, in addition to the two protocols mentioned above, we also evaluate the frozen features with non-linear probing, *e.g.* in the case of a classification downstream task, a non-linear MLP head is trained as the final classifier. In the experiments we evaluate the representation on four different downstream tasks.

Action Classification is a common evaluation task for self-supervised learning on videos and it allows us to compare against other methods. After self-supervised training, our `MemDPC` can be evaluated on this task under two settings:

(i) linear and non-linear probing with a fixed network (here the entire backbone network, namely $f(.)$, $g(.)$); and (ii) fine-tuning the entire network end-to-end with supervised learning. For the embedding, as shown in Figure 4, we take the input video blocks $x_1, x_2, ..., x_t$ in the same way as `MemDPC` and extract the context feature c_t using the feature extractor $f(.)$ for each block and temporal aggregator $g(.)$; then we spatially pool the context feature c_t to obtain the embedding. We describe the training details in Section 5.3. The detailed experiment can be found in Section 5.4.

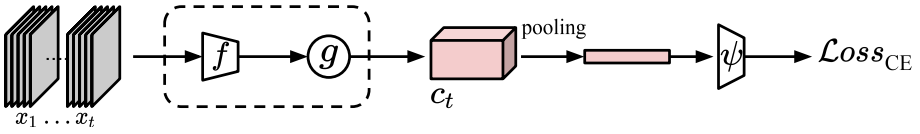

Fig. 4. Architecture of the action classification framework

Data Efficiency and Generalizability are reflected by the effectiveness of the representation under a scarce-annotation regime. For this task, we take the `MemDPC` representation and finetune it for action classification task, but limit the model to only use 10%, 20% and 50% of the labelled training samples, then we report the accuracy on the same testing set. The classifier has the identical training pipeline as shown in Figure 4, and training details are explained in Section 5.3. The detailed experiment can be found in Section 5.5.

Video Action Retrieval directly evaluates the quality of the representation without any further training, aiming to provide a straightforward understanding on the quality of the learnt representation. Here, we use the simplest non-parametric classifier, *i.e.* k-nearest neighbours, to determine whether semantically similar actions are close in the high-dimensional space. Referring to Figure 4, for each video, we truncate it into blocks $x_1, x_2, ..., x_t$ and extract the context feature c_t with the $f(.)$ and $g(.)$ trained with `MemDPC`. We spatially pool c_t to get a context feature vector, which is directly used as a query vector for measuring the similarity with other videos in the dataset. The detailed experiment can be found in Section 5.6.

Unintentional Actions is a straightforward application for a predictive framework like `MemDPC`. We evaluate our representation on the task of unintentional event classification that is proposed in the recent Oops dataset [17]. The core of unintentional events detection in video is a problem of anomaly detection. Usually, one of the predicted hypotheses tends to match true future relatively well for most of the videos. The discrepancy between them yields a measurement of future predictability, or 'surprise' level. A big surprise or a mismatched prediction can be used to locate the failing moment. In detail, we design the model

as follows: first, we compute both the predicted feature $\hat{z}_i$ and the corresponding true feature z_i, and let a function $\xi(.)$ to measure their discrepancy. We train the model with two settings: (i) freezing the representation and only train the classifier $\xi(.)$; (ii) finetuning the entire network. The detailed structure for the classification task can be found in Section 5.7.

5 Experiments

5.1 Datasets

For the self-supervised training, two video action recognition datasets are used, but labels are dropped during training: *UCF101* [54], containing 13k videos spanning over 101 human actions; and Kinetics400 (*K400*) [36] with 306k 10-second video clips covering 400 human actions. For the downstream tasks we also use UCF101, and additionally we use: *HMDB51* [40] containing 7k videos spanning over 51 human actions; and *Oops* [17] containing 20k videos of daily human activities with unexpected failed moments, among them 14k videos have the time stamps of the failed moments manually labelled.

5.2 Self-Supervised Training

In our experiment, we use a (2+3D)-ResNet, following [18, 24], as the encoder $f(.)$, where the first two residual blocks res2 and res3 have 2D convolutional kernels, and only res4 and res5 have 3D kernels. Specifically, (2+3D)-ResNet18 and (2+3D)-ResNet34 are used in our experiments, denoted as R18 and R34 below. For the temporal aggregation, $g(\cdot)$, we use an one-layer GRU with kernel size 1×1, with the weights shared among all spatial positions on the feature map. The future prediction function, $\phi(.)$, is a two-layer convolutional network. We choose the size of the memory bank $\mathbf{M}$ to be 1024 based on experiments in Table 1. Network architecture are given in the supplementary material.

For the data, raw videos are decoded at a frame rate 24-30 fps, and each data sample consists of 40 consecutive frames, sampled with a temporal stride of 3 from the raw video. As input to `MemDPC`, they are divided into 8 video clips – so that each encoder $f(.)$ inputs 5 frames, covering around 0.5 seconds, and the 40 frames around 4 seconds. For optical flow, in order to eliminate extra supervisory signals in the self-supervised training stage, we use the **un-supervised** TV-L1 algorithm [68], and follow the same pre-processing procedures as [10], *i.e.* truncating large motions with more than ± 20 in both channels, appending a third 0s channel, and transforming the values from $[-20, 20]$ to $[0, 255]$. For data augmentation, we apply clip-wise random crop and horizontal flip, and frame-wise color jittering and random greyscale, for both the RGB and optical flow streams. We experiment with both 128×128 and 224×224 input resolution. The original video resolution is 256×256 and it is firstly cropped to 224×224 then rescaled if needed. Self-supervised training uses the Adam [38] optimizer with initial learning rate 10^{-3}. The learning rate is decayed once to 10^{-4} when the validation loss plateaus. We use a batch size of 16 samples per GPU.

5.3 Supervised Classification

For all action classification downstream tasks, the input follows the same frame sampling procedure as when the model is trained with self-supervised learning, and then we train the classifier with cross-entropy loss as shown in Figure 4. A dropout of 0.9 is applied on the final layer. For data augmentation, we use clip-wise random crop, random horizontal flip, and random color jittering. The classifier is trained with Adam with a 10^{-3} initial learning rate, and decayed once to 10^{-4} when the validation loss plateaus. During testing, we follow the standard pipeline, *i.e.* ten-crop (center and four corner crops, w/o horizontal flip), take the same sequence length as training from the video, and average the prediction from the sampling temporal moving window.

5.4 Evaluation: Action Classification

We conduct two sets of experiments: (i) ablation studies on the effectiveness of the different modules in the `MemDPC`, by self-supervised learning on UCF101, (ii) to compare with other state-of-the-art approaches, we run `MemDPC` on K400 with self-supervised learning. For both settings, the representation quality is evaluated on UCF101 and HMDB51 with linear probing, non-linear probing, and end-to-end finetuning.

Ablations on UCF101. In this section, we conduct extensive experiments to validate the effectiveness of compressive memory, bidirectional aggregation, and self-supervised learning on optical flow. Note that, in each experiment, we keep the settings identical, and only vary one variable at a time.

As shown in Table 1, the following phenomena can be observed: *First*, comparing experiment $\mathcal{C}2$ against $\mathcal{B}1$ (68.2 vs. 61.8), networks initialized with self-supervised `MemDPC` clearly present better generalization than a randomly initialized network; *Second*, comparing with a strong baseline ($\mathcal{A}$), the proposed compressive memory boost the learnt representation by around 5% (68.2 vs. 63.6), and the optimal memory size for UCF101 is 1024; *Third*, `MemDPC` acts as a general learning framework that can also help to boost the generalizability of motion representations, a 7.3% boost can be seen from $\mathcal{D}1$ vs. $\mathcal{B}2$ (81.9 vs. 74.6); *Fourth*, the bidirectional aggregation provides a small boost to the accuracy by about 1% ($\mathcal{E}1$ vs. $\mathcal{C}2$, $\mathcal{E}2$ vs. $\mathcal{D}1$, $\mathcal{E}3$ vs. $\mathcal{D}2$). *Lastly*, after fusing both streams, $\mathcal{D}2$ achieves 84% classification accuracy, confirming our claim that self-supervised learning with only the video stream (without additional audio or text streams) can also end up with strong action recognition models.

Comparison with others. In this section, we train `MemDPC` on K400 and evaluate the action classification performance on UCF101 and HMDB51. Specifically, we evaluate three settings: (1) finetuning the entire network (denoted as Freeze=✗); (2) freeze the backbone and only train a linear classifier, *i.e.* linear

Table 1. Ablation studies. We train `MemDPC` on UCF101 and evaluate on UCF101 action classification by finetuning the entire network. We group rows for clarity: $\mathcal{A}$ is the reimplementation of DPC, $\mathcal{B}$ are random initialization baselines, $\mathcal{C}$ for different memory size, $\mathcal{D}$ incorporates optical flow, $\mathcal{E}$ incorporates a bi-directional RNN

#	Network	Self-Sup. Dataset	Input	Resolution	Memory size	Sup. (top1) UCF101(ft)
$\mathcal{A}$	R18	UCF101	RGB	128 × 128	- (DPC [24])	63.6
$\mathcal{B}1$	R18	-(rand. init.)	RGB	128 × 128	-	61.8
$\mathcal{B}2$	R18	-(rand. init.)	Flow	128 × 128	-	74.6
$\mathcal{B}3$	R18×2	-(rand. init.)	RGB+F	128 × 128	-	78.7
$\mathcal{C}1$	R18	UCF101	RGB	128 × 128	512	65.3
$\mathcal{C}2$	R18	UCF101	RGB	128 × 128	1024	68.2
$\mathcal{C}3$	R18	UCF101	RGB	128 × 128	2048	68.0
$\mathcal{D}1$	R18	UCF101	Flow	128 × 128	1024	81.9
$\mathcal{D}2$	R18×2	UCF101	RGB+F	128 × 128	1024	84.0
$\mathcal{E}1$	R18-bd	UCF101	RGB	128 × 128	1024	69.2
$\mathcal{E}2$	R18-bd	UCF101	Flow	128 × 128	1024	82.3
$\mathcal{E}3$	R18-bd×2	UCF101	RGB+F	128 × 128	1024	84.3

Table 2. Comparison with state-of-the-art approaches. In the left columns, we show the pre-training setting, *e.g.* dataset, resolution, architectures with encoder depth, modality. In the right columns, the top-1 accuracy is reported on the downstream action classification task for different datasets, *e.g.* UCF, HMDB, K400. The dataset parenthesis shows the total video duration in time (**d** for day, **y** for year). 'Frozen ✗' means the network is end-to-end finetuned from the pretrained representation, shown in the top half of the table; 'Frozen ✓' means the pretrained representation is fixed and classified with a linear layer, 'n.l.' denotes a non-linear classifier. For input, 'V' refers to visual only (colored with blue), 'A' is audio, 'T' is text narration. MemDPC models with † refer to the two-stream networks, where the predictions from RGB and Flow networks are averaged

Method	Date	Dataset (duration)	Res.	Arch.	Depth	Modality	Frozen	UCF	HMDB
CBT [56]	2019	K600+ (273d)	112	S3D	23	V	✓	54.0	29.5
MIL-NCE [47]	2020	HTM (15y)	224	S3D	23	V+T	✓	82.7	53.1
MIL-NCE [47]	2020	HTM (15y)	224	I3D	22	V+T	✓	83.4	54.8
XDC [2]	2019	IG65M (21y)	224	R(2+1)D	26	V+A	✓	85.3	56.0
ELO [52]	2020	Youtube8M- (8y)	224	R(2+1)D	65	V+A	✓	–	64.5
MemDPC†		K400 (28d)	224	R-2D3D	33	V	✓	54.1	30.5
MemDPC†		K400 (28d)	224	R-2D3D	33	V	✓n.l.	58.5	33.6
OPN [44]	2017	UCF (1d)	227	VGG	14	V	✗	59.6	23.8
3D-RotNet [35]	2018	K400 (28d)	112	R3D	17	V	✗	62.9	33.7
ST-Puzzle [37]	2019	K400 (28d)	224	R3D	17	V	✗	63.9	33.7
VCOP [66]	2019	UCF (1d)	112	R(2+1)D	26	V	✗	72.4	30.9
DPC [24]	2019	K400 (28d)	224	R-2D3D	33	V	✗	75.7	35.7
CBT [56]	2019	K600+ (273d)	112	S3D	23	V	✗	79.5	44.6
DynamoNet [15]	2019	Youtube8M-1 (1.9y)	112	STCNet	133	V	✗	88.1	59.9
SpeedNet [7]	2020	K400 (28d)	224	S3D-G	23	V	✗	81.1	48.8
AVTS [39]	2018	K400 (28d)	224	I3D	22	V+A	✗	83.7	53.0
AVTS [39]	2018	AudioSet (240d)	224	MC3	17	V+A	✗	89.0	61.6
XDC [2]	2019	K400 (28d)	224	R(2+1)D	26	V+A	✗	84.2	47.1
XDC [2]	2019	IG65M (21y)	224	R(2+1)D	26	V+A	✗	94.2	67.4
GDT [51]	2020	K400 (28d)	112	R(2+1)D	26	V+A	✗	88.7	57.8
MIL-NCE [47]	2020	HTM (15y)	224	S3D-G	23	V+T	✗	91.3	61.0
ELO [52]	2020	Youtube8M-2 (13y)	224	R(2+1)D	65	V+A	✗	93.8	67.4
MemDPC		K400 (28d)	224	R-2D3D	33	V	✗	78.1	41.2
MemDPC†		K400 (28d)	224	R-2D3D	33	V	✗	86.1	54.5
Supervised [25]		K400 (28d)	224	R3D	33	V	✗	87.7	59.1

probe (denoted as Freeze=✓); (3) freeze the backbone and only train a **non-linear** classifier, *i.e.* non-linear probe (denoted as 'n.l.').

As shown in Table 2, for the same amount of data (K400) and visual-only input, `MemDPC` surpasses all previous state-of-the-art self-supervised methods on

both UCF101 and HMDB51 (although there exist small differences in architecture, *e.g.* for 3DRotNet, ST-Puzzle, DPC, SpeedNet). When freezing the representation, it can be seen that a non-linear probe gives better results than a linear probe, and in practice a non-linear classifier is still very cheap to train.

Other self-supervised training methods on the same benchmarks are not directly comparable, even ignoring the architecture differences, due to the duration of videos used or to the number of modalities used. For example, CBT [56] uses a longer version of K600 (referred to as K600+ in the table), the size is about 9 times that of the standard K400 that we use, and CBT requires RotNet [35] initialization while `MemDPC` can be trained from scratch. Nevertheless, our performance exceeds that of CBT. Other works use additional modalities for pre-text tasks like audio [2,39,51,52], or narrations [47], and train on larger datasets. Despite these disadvantages, we demonstrate that `MemDPC` trained with only visual inputs, can achieve competitive results on the finetuning protocol.

5.5 Evaluation: Data Efficiency

In Figure 5, we show the data efficiency of `MemDPC` on both RGB input and optical flow with action recognition on the UCF101 dataset. As we reduce the labelled training samples, action classifier trained on `MemDPC` representation generalize significantly better than the classifier trained from scratch. Also, to match the performance of a random initialized classifier trained on 100% labelled data, a classifier trained on `MemDPC` initialization only requires less than 50% labelled data for both RGB and optical flow input.

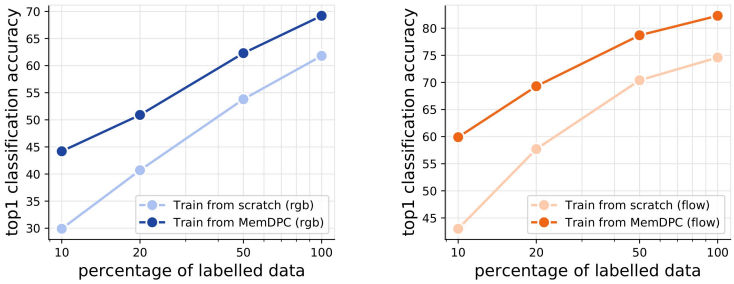

Fig. 5. Data efficiency of `MemDPC` representations. Left is RGB input and right is optical flow input. The `MemDPC` is trained on UCF101 and it is evaluated on action classification (finetuning protocol) on UCF101 with a reduced number of labels

5.6 Evaluation: Video Retrieval

In this protocol, we evaluate our representation with nearest-neighbour video retrieval, features are extracted from the model, which is only trained with self-supervised learning, no further finetuning is allowed.

Experiments are shown on two datasets: UCF101 and HMDB51. For both datasets, within the training set or within the testing set, multiple clips could be from the same source video, hence they are visually similar and make the retrieval task trivial. We follow the practice of [46, 66], and use each clip in the test set to query the k nearest clips in the training set.

For each clip, we sample multiple 8 video blocks with a sliding window, and extract the context representation c_t for each window. We spatial-pool each c_t and take the average over all the windows. For distance measurement, we use cosine distance. We report Recall at k (R@k) as the evaluation metric. That is, as long as one clip of the same class is retrieved in the top k nearest neighbours, a correct retrieval is counted.

Table 3. Comparison with others on Nearest-Neighbour video retrieval on UCF101 and HMDB51. Testing set clips are used to retrieve training set videos and R@k is reported, where $k \in [1, 5, 10, 20]$. Note that all the models reported were only pretrained on UCF101 with self-supervised learning except SpeedNet

Method	Date	Dataset	UCF				HMDB			
			R@1	R@5	R@10	R@20	R@1	R@5	R@10	R@20
Jigsaw [49]	2016	UCF	19.7	28.5	33.5	40.0	-	-	-	-
OPN [44]	2017	UCF	19.9	28.7	34.0	40.6	-	-	-	-
Buchler [9]	2018	UCF	25.7	36.2	42.2	49.2	-	-	-	-
VCOP [66]	2019	UCF	14.1	30.3	40.4	51.1	7.6	22.9	34.4	48.8
VCP [46]	2020	UCF	18.6	33.6	42.5	53.5	7.6	24.4	36.3	53.6
SpeedNet [7]	2020	K400	13.0	28.1	37.5	49.5	-	-	-	-
MemDPC-RGB		UCF	20.2	**40.4**	**52.4**	**64.7**	**7.7**	**25.7**	**40.6**	**57.7**
MemDPC-Flow		UCF	**40.2**	**63.2**	**71.9**	**78.6**	**15.6**	**37.6**	**52.0**	**65.3**

In Table 3, we show the retrieval performance on UCF101 and HMDB51. Note that the MemDPC benchmarked here is only trained on UCF101, the same as [46, 66]. For fair comparison, MemDPC in this experiment uses a R18 backbone, which has the same depth but less parameters than the 3D-ResNet used in [46, 66]. With RGB inputs, our MemDPC gets state-of-the-art performance on all the metrics except R@1 in UCF101, where the method from Buchler *et al.* [9] specializes well on R@1. While for Flow inputs, MemDPC significantly outperforms all previous methods by a large margin. We also qualitatively show video retrieval results in the supplementary material.

5.7 Evaluation: Unintentional Actions

We evaluate MemDPC on the Oops dataset on unintentional action classification. In Oops, there is one failure moment in the middle of each video. When cutting the video into short clips, the clip overlapping the failure moment is defined as a 'transitioning' action, the clips before are 'intentional' actions, and the clips afterwards are 'unintentional' actions. The core task is therefore to classify each short video clip into one of three categories,

In this experiment, we use a R18 based MemDPC model that takes 128×128 resolution video frames as input. After MemDPC is trained on K400 and the Oops

training set videos with self-supervised learning, we further train it for unintentional action classification with a linear probe, and end-to-end finetuning (as shown in Table 4). The training details are given in the supplementary material. State-of-the-art performance is demonstrated by our MemDPC on this unintentional action classification task, even outperforming the model pretrained on K700 with full supervision with finetuning.

Table 4. MemDPC on unintentional action classification tasks. Note that our backbone 2+3D-ResNet18 has the same depth as 3D-ResNet18 used in [17] but with less parameters. MemDPC model is trained on K400 and the OOPS training set without using labels, and the network is then finetuned with supervision from the OOPS training set

Task	Method	Backbone	Freeze	Finetune
Classification	K700 Supervision	3D-ResNet18	**53.6**	64.0
	Video Speed [17]	3D-ResNet18	53.4	61.6
	MemDPC	R18	53.0	**64.4**

6 Conclusion

In this paper, we propose a new architecture and learning framework (MemDPC) for self-supervised learning from video, in particular for representations for action recognition. With the novel compressive memory, the model can efficiently handle the nature of multiple hypotheses in the self-supervised predictive learning procedure. In order to thoroughly evaluate the quality of the learnt representation, we conduct experiments on four different downstream tasks, namely action recognition, video retrieval, learning with scarce annotations, and unintentional action classification. In all cases, we demonstrate state-of-the-art or competitive performance over other approaches that use orders of magnitude more training data. Above all, for the first time, we show that it is possible to learn high-quality video representations with self-supervised learning, from the visual stream alone (without additional audio or text streams).

Acknowledgements.

Funding for this research is provided by a Google-DeepMind Graduate Scholarship, and by the EPSRC Programme Grant Seebibyte EP/M013774/1. We would like to thank João F. Henriques, Samuel Albanie and Triantafyllos Afouras for helpful discussions.

References

1. Agrawal, P., Carreira, J., Malik, J.: Learning to see by moving. In: Proc. ICCV. pp. 37–45. IEEE (2015)
2. Alwassel, H., Mahajan, D., Torresani, L., Ghanem, B., Tran, D.: Self-supervised learning by cross-modal audio-video clustering. arXiv preprint arXiv:1911.12667 (2019)
3. Arandjelović, R., Gronat, P., Torii, A., Pajdla, T., Sivic, J.: NetVLAD: CNN architecture for weakly supervised place recognition. In: Proc. CVPR (2016)
4. Arandjelović, R., Zisserman, A.: Look, listen and learn. In: Proc. ICCV (2017)
5. Arandjelović, R., Zisserman, A.: Objects that sound. In: Proc. ECCV (2018)
6. Bahdanau, D., Cho, K., Bengio, Y.: Neural machine translation by jointly learning to align and translate. In: Proc. ICLR (2015)
7. Benaim, S., Ephrat, A., Lang, O., Mosseri, I., Freeman, W.T., Rubinstein, M., Irani, M., Dekel, T.: SpeedNet: Learning the Speediness in Videos. In: Proc. CVPR (2020)
8. Brabandere, B.D., Jia, X., Tuytelaars, T., Gool, L.V.: Dynamic filter networks. In: NeurIPS (2016)
9. Büchler, U., Brattoli, B., Ommer, B.: Improving spatiotemporal self-supervision by deep reinforcement learning. In: Proc. ECCV (2018)
10. Carreira, J., Zisserman, A.: Quo vadis, action recognition? A new model and the kinetics dataset. In: Proc. CVPR (2017)
11. Chen, T., Kornblith, S., Norouzi, M., Hinton, G.: A simple framework for contrastive learning of visual representations. In: Proc. ICML (2020)
12. Chopra, S., Hadsell, R., LeCun, Y.: Learning a similarity metric discriminatively, with application to face verification. In: Proc. CVPR (2005)
13. Chung, J., Gulcehre, C., Cho, K., Bengio, Y.: Empirical evaluation of gated recurrent neural networks on sequence modeling. arXiv preprint arXiv:1412.3555 (2014)
14. Devlin, J., Chang, M., Lee, K., Toutanova, K.: Bert: Pre-training of deep bidirectional transformers for language understanding. arXiv preprint arXiv:1810.04805 (2018)
15. Diba, A., Sharma, V., Gool, L.V., Stiefelhagen, R.: DynamoNet: Dynamic Action and Motion Network. In: Proc. ICCV (2019)
16. Dosovitskiy, A., Koltun, V.: Learning to act by predicting the future. In: Proc. ICLR (2017)
17. Epstein, D., Chen, B., Vondrick, C.: Oops! predicting unintentional action in video. In: Proc. CVPR (2020)
18. Feichtenhofer, C., Pinz, A., Wildes, R.P., Zisserman, A.: What have we learned from deep representations for action recognition? In: Proc. CVPR (2018)
19. Feichtenhofer, C., Pinz, A., Zisserman, A.: Convolutional two-stream network fusion for video action recognition. In: Proc. CVPR (2016)
20. Fernando, B., Bilen, H., Gavves, E., Gould, S.: Self-supervised video representation learning with odd-one-out networks. In: Proc. ICCV (2017)
21. Gan, C., Gong, B., Liu, K., Su, H., Guibas, L.J.: Geometry guided convolutional neural networks for self-supervised video representation learning. In: Proc. CVPR (2018)
22. Graves, A., Wayne, G., Danihelka, I.: Neural turing machines. arXiv preprint arXiv:1410.5401 (2014)
23. Gutmann, M.U., Hyvärinen, A.: Noise-contrastive estimation: A new estimation principle for unnormalized statistical models. In: AISTATS (2010)

24. Han, T., Xie, W., Zisserman, A.: Video representation learning by dense predictive coding. In: Workshop on Large Scale Holistic Video Understanding, ICCV (2019)
25. Hara, K., Kataoka, H., Satoh, Y.: Can spatiotemporal 3d cnns retrace the history of 2d cnns and imagenet? In: CVPR (2018)
26. He, K., Fan, H., Wu, A., Xie, S., Girshick, R.: Momentum contrast for unsupervised visual representation learning. In: Proc. CVPR (2020)
27. Hénaff, O.J., Razavi, A., Doersch, C., Eslami, S.M.A., van den Oord, A.: Data-efficient image recognition with contrastive predictive coding. arXiv preprint arXiv:1905.09272 (2019)
28. Hjelm, R.D., Fedorov, A., Lavoie-Marchildon, S., Grewal, K., Bachman, P., Trischler, A., Bengio, Y.: Learning deep representations by mutual information estimation and maximization. In: Proc. ICLR (2019)
29. Hochreiter, S., Schmidhuber, J.: Long short-term memory. Neural computation **9**(8), 1735–1780 (1997)
30. Isola, P., Zoran, D., Krishnan, D., Adelson, E.H.: Learning visual groups from co-occurrences in space and time. In: Proc. ICLR (2015)
31. Jakab, T., Gupta, A., Bilen, H., Vedaldi, A.: Unsupervised learning of object landmarks through conditional image generation. In: NeurIPS (2018)
32. Jayaraman, D., Grauman, K.: Learning image representations tied to ego-motion. In: Proc. ICCV (2015)
33. Jayaraman, D., Grauman, K.: Slow and steady feature analysis: higher order temporal coherence in video. In: Proc. CVPR (2016)
34. Ji, X., Henriques, J.F., Vedaldi, A.: Invariant information clustering for unsupervised image classification and segmentation. In: Proc. ICCV (2019)
35. Jing, L., Tian, Y.: Self-supervised spatiotemporal feature learning by video geometric transformations. arXiv preprint arXiv:1811.11387 (2018)
36. Kay, W., Carreira, J., Simonyan, K., Zhang, B., Hillier, C., Vijayanarasimhan, S., Viola, F., Green, T., Back, T., Natsev, P., Suleyman, M., Zisserman, A.: The kinetics human action video dataset. arXiv preprint arXiv:1705.06950 (2017)
37. Kim, D., Cho, D., Kweon, I.S.: Self-supervised video representation learning with space-time cubic puzzles. In: AAAI (2019)
38. Kingma, D.P., Ba, J.: Adam: A method for stochastic optimization. arXiv preprint arXiv:1412.6980 (2014)
39. Korbar, B., Tran, D., Torresani, L.: Cooperative learning of audio and video models from self-supervised synchronization. In: NeurIPS (2018)
40. Kuehne, H., Jhuang, H., Garrote, E., Poggio, T., Serre, T.: HMDB: A large video database for human motion recognition. In: Proc. ICCV. pp. 2556–2563 (2011)
41. Kumar, A., Irsoy, O., Ondruska, P., Iyyer, M., Bradbury, J., Gulrajani, I., Zhong, V., Paulus, R., Socher, R.: Ask me anything: Dynamic memory networks for natural language processing. In: Proc. ICML (2016)
42. Lai, Z., Lu, E., Xie, W.: MAST: A memory-augmented self-supervised tracker. In: Proc. CVPR (2020)
43. Lai, Z., Xie, W.: Self-supervised learning for video correspondence flow. In: Proc. BMVC (2019)
44. Lee, H., Huang, J., Singh, M., Yang, M.: Unsupervised representation learning by sorting sequence. In: Proc. ICCV (2017)
45. Lotter, W., Kreiman, G., Cox, D.: Deep predictive coding networks for video prediction and unsupervised learning. In: Proc. ICLR (2017)
46. Luo, D., Liu, C., Zhou, Y., Yang, D., Ma, C., Ye, Q., Wang, W.: Video cloze procedure for self-supervised spatio-temporal learning. In: AAAI (2020)

47. Miech, A., Alayrac, J.B., Smaira, L., Laptev, I., Sivic, J., Zisserman, A.: End-to-end learning of visual representations from uncurated instructional videos. In: Proc. CVPR (2020)
48. Misra, I., Zitnick, C.L., Hebert, M.: Shuffle and learn: Unsupervised learning using temporal order verification. In: Proc. ECCV (2016)
49. Noroozi, M., Favaro, P.: Unsupervised learning of visual representations by solving jigsaw puzzles. In: Proc. ECCV (2016)
50. van den Oord, A., Li, Y., Vinyals, O.: Representation learning with contrastive predictive coding. arXiv preprint arXiv:1807.03748 (2018)
51. Patrick, M., Asano, Y.M., Fong, R., Henriques, J.F., Zweig, G., Vedaldi, A.: Multi-modal self-supervision from generalized data transformations. arXiv preprint arXiv:2003.04298 (2020)
52. Piergiovanni, A., Angelova, A., Ryoo, M.S.: Evolving losses for unsupervised video representation learning. In: Proc. CVPR (2020)
53. Simonyan, K., Zisserman, A.: Two-stream convolutional networks for action recognition in videos. In: NeurIPS (2014)
54. 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)
55. Sukhbaatar, S., Szlam, A., Weston, J., Fergus, R.: End-to-end memory networks. In: NeurIPS (2015)
56. Sun, C., Baradel, F., Murphy, K., Schmid, C.: Contrastive bidirectional transformer for temporal representation learning. arXiv preprint arXiv:1906.05743 (2019)
57. Tian, Y., Krishnan, D., Isola, P.: Contrastive multiview coding. arXiv preprint arXiv:1906.05849 (2019)
58. Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A.N., Kaiser, L., Polosukhin, I.: Attention is all you need. In: NIPS (2017)
59. Vondrick, C., Pirsiavash, H., Torralba, A.: Anticipating visual representations from unlabelled video. In: CVPR (2016)
60. Vondrick, C., Shrivastava, A., Fathi, A., Guadarrama, S., Murphy, K.: Tracking emerges by colorizing videos. In: ECCV (2018)
61. Wang, X., Girshick, R., Gupta, A., He, K.: Non-local neural networks. In: Proc. CVPR (2018)
62. Wang, X., Gupta, A.: Unsupervised learning of visual representations using videos. In: Proc. ICCV (2015)
63. Wang, X., Jabri, A., Efros, A.A.: Learning correspondence from the cycle-consistency of time. In: CVPR (2019)
64. Wiles, O., Koepke, A.S., Zisserman, A.: Self-supervised learning of a facial attribute embedding from video. In: Proc. BMVC (2018)
65. Wu, Z., Xiong, Y., Yu, S., Lin, D.: Unsupervised feature learning via non-parametric instance-level discrimination. In: Proc. CVPR. vol. abs/1805.01978 (2018)
66. Xu, D., Xiao, J., Zhao, Z., Shao, J., Xie, D., Zhuang, Y.: Self-supervised spatiotemporal learning via video clip order prediction. In: CVPR (2019)
67. Xu, K., Ba, J., Kiros, R., Cho, K., Courville, A., Salakhudinov, R., Zemel, R., Bengio, Y.: Show, attend and tell: Neural image caption generation with visual attention. In: Proc. ICML (2015)
68. Zach, C., Pock, T., Bischof, H.: A duality based approach for realtime TV-L1 optical flow. In: Pattern Recognition (2007)
69. Zhang, R., Isola, P., Efros, A.A.: Colorful image colorization. In: Proc. ECCV (2016)

70. Zhuang, C., Zhai, A.L., Yamins, D.: Local aggregation for unsupervised learning of visual embeddings. In: ICCV (2019)