Watching the World Go By: Representation Learning from Unlabeled Videos

Daniel Gordon[1] Kiana Ehsani[1] Dieter Fox[1,2] Ali Farhadi[1]
[1]University of Washington [2]Nvidia

danielgordon@cs.washington.edu

Abstract. Recent single image unsupervised representation learning techniques show remarkable success on a variety of tasks. The basic principle in these works is instance discrimination: learning to differentiate between two augmented versions of the same image and a large batch of unrelated images. Networks learn to ignore the augmentation noise and extract semantically meaningful representations. Prior work uses artificial data augmentation techniques such as cropping, and color jitter which can only affect the image in superficial ways and are not aligned with how objects actually change e.g. occlusion, deformation, viewpoint change. In this paper, we argue that videos offer this natural augmentation for free. Videos can provide entirely new views of objects, show deformation, and even connect semantically similar but visually distinct concepts. We propose Video Noise Contrastive Estimation, a method for using unlabeled video to learn strong, transferable single image representations. We demonstrate improvements over recent unsupervised single image techniques, as well as over fully supervised ImageNet pretraining, across a variety of temporal and non-temporal tasks.

1 Introduction

The world seen through our eyes is constantly changing. As we move through and interact with the world, we see much more than a single static image: objects rotate revealing occluded regions, deform, the surroundings change, and we ourselves move. Our internal visual systems are constantly seeing temporally coherent images. Yet many popular computer vision models learn representations which are limited to inference on single images, lacking temporal context. Visual representations learned from static images will be inherently limited to an understanding of the world as many unrelated static snapshots.

This is especially true of recent unsupervised learning techniques [2, 6, 11, 13, 14, 25, 33, 38], all of which train on a set of highly-curated, well-balanced data: ImageNet [7]. Scaling up single-image techniques to larger, less-curated datasets like Instagram-1B [22] has not provided large improvements in performance [11]. There is only so much that can be learned from a single image: no amount of artificial augmentation can show a new view of an object or what might happen next in a scene. This dichotomy can be seen in Figure 1.

In order to move beyond this limitation, we argue that video supplies significantly more semantically meaningful content than a single image. With video, we can see how the world changes, find connections between images, and more directly observe the

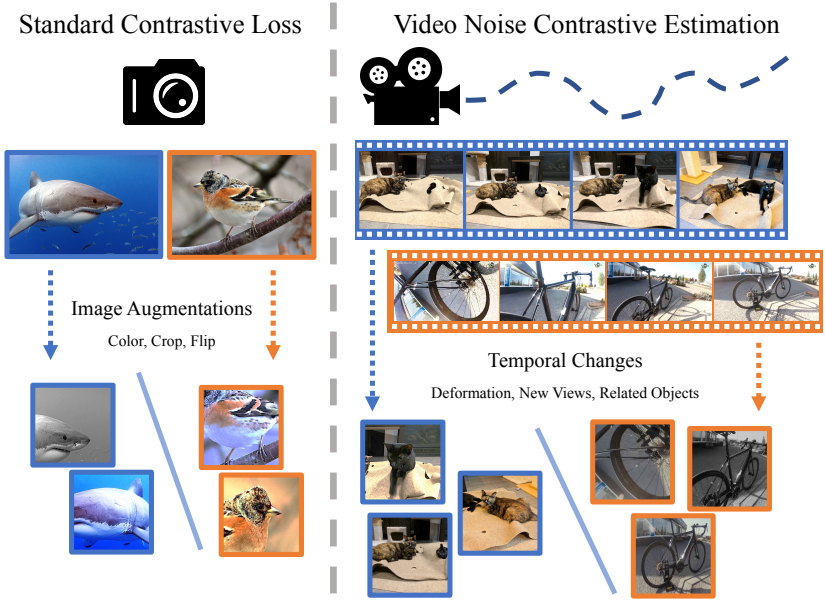


Fig. 1: The standard unsupervised learning setup learns to separate multiple augmentations of the same image. Our method uses truly novel views and temporal consistency which single images cannot provide.

underlying scene. Prior work using temporal cues has shown success in learning from unlabeled videos [26, 36, 32], but has not been able to surpass supervised pretraining. On the other hand, single image techniques have shown improvements over state-of-the-art by using Noise Contrastive Estimation [10] (NCE). In this work, we merge the two concepts with Video Noise Contrastive Estimation (VINCE), a method for using unlabeled videos as a basis for learning visual representations. Instead of predicting whether two feature vectors come from the same underlying image, we task our network with predicting whether two images originate from the same video. Not only does this allow our method to learn how a single object might change, it also enables learning which things might be in a scene together, e.g. cats are more likely to be in videos with dogs than with sharks. Additionally, we generalize the NCE technique to operate on multiple positive pairs from a single source. To facilitate this learning, we construct Random Related Video Views (R2V2), a set 960,000 frames from 240,000 uncurated videos. Using our learning technique, we achieve across-the-board improvements over the recent Momentum Contrast method [11] as well as over a network pretrained on supervised ImageNet on diverse tasks such as scene classification, activity recognition, and object tracking.[1]

[1] Code and the Random Related Video Views dataset will be released soon.

2 Related Work

2.1 Noise Contrastive Estimation (NCE)

The NCE loss [10] is at the center of many recent representation learning methods [2, 6, 11, 13, 14, 25, 33, 38]. Similar to the triplet loss [5], the basic principle behind NCE is to maximize the similarity between an anchor data point and a positive data point while minimizing similarity to all other (negative) points.

A challenge for using NCE in an unsupervised fashion is devising a way to construct positive pairs. Pairs should be different enough that a network learns a non-trivial representation, but structured enough that the learned representation is useful for downstream tasks. A standard approach used by [2, 6, 11] is to generate the pairs via artificial data augmentation techniques such as color jitter, cropping, and flipping. Contrastive Multiview Coding [33] uses multiple "views" of a single source image such as intensity (L), color (ab), depth, or segmentation, training separate encoders for each view. PIRL [25] uses the jigsaw technique [27] to break the image into non-overlapping regions and learns a shared representation for the full image and the shuffled image patches. Similarly, CPC [13] uses crops of an image as "context" and predicts features for the unseen portions of the image. We provide a more natural data augmentation by using multiple frames from a single video. As a video progresses, the objects in the scene, the background, and the camera itself may move, providing new views. Whereas augmentations on an image are constrained by a single snapshot in time, using different frames from a single video gives entirely new information about the scene. Additionally, rather than restricting our method to only use two frames from a video, we generalize the NCE technique to use many images from a single video, resulting in more computational reuse and a better final representation (AMDIM [2] similarly makes multiple comparisons per pair, but each anchor has only one positive).

2.2 Unsupervised Learning Using Video Cues

In contrast with supervised learning which requires hand-labeling, self-supervised and unsupervised learning acquire their labels for free. These techniques can create datasets which are orders of magnitude larger than comparable fully-supervised datasets. Whereas self-supervised learning requires extra setup during data generation [9, 28, 31], unsupervised learning can use existing data without the need for any specific generation constraints. Unsupervised single image methods such as auto-encoders [19], colorization [40], GANs [29], jigsaw [27], and NCE [38] rely on properties of the images themselves and can be applied to arbitrary image datasets. However these image datasets cannot represent temporal information, nor can they show novel object views or occlusions.

Video data automatically provides temporal cohesion which can be used as additional supervisory signal to learn these phenomena. There is a long history of using videos for low level [20, 23, 32] and high-level tasks [26, 36]. One of the most common unsupervised setups is using the present to predict the future. The Natural Language Processing community has embraced language modeling as an unsupervised task which has resulted in numerous breakthroughs [8, 24, 30]. However, similar systems applied

to unlabeled videos have not revolutionized computer vision. These representations still underperform supervised methods due to several issues. Primarily, neighboring video frames do not change nearly as much as neighboring words in a sentence, so a network which learns the identity function would perform well at next frame prediction. Additionally, words are reused and can thus be tokenized in an effective way whereas images never repeat, especially between two disparate video sources.

To avoid these issues, many have opted for other methods. Misra *et al*. [26] shuffle the frames of a video and train a network to predict whether they are correctly temporally ordered. Wang *et al*. [36] and Vondrick *et al*. [34] use cycle consistency and color as a form of tracking points from one frame onto another. Wang *et al*. [35] use hand-crafted features to track patches of a video and learn a correspondence between the patches. Our approach is inspired by these works but focuses on learning a semantic representation of the entire scene. If a network can consistently represent visually dissimilar images from the same video with similar vectors, then not only has it learned how to recognize what is in each image, but it can also represent what might happen in the past or future of that scene.

3 Methods

In order to learn a semantically meaningful representation, we exploit the natural augmentations provided by unlabeled videos. In this section, we first outline the dataset generation process. We then describe the learning algorithm used to train our representation.

3.1 Dataset

Using ImageNet as a basis for representation learning has shown remarkable success both with supervised pretraining as well as unsupervised learning. However, even without labels, the images of ImageNet have been hand selected and are unnaturally balanced. To improve learned representations using existing techniques may require significantly larger datasets [11], but obtaining data with similar properties automatically and at scale is not practical. Instead, we turn to unlabeled videos as a source of additional supervision.

In order to train on a diverse set of realistic video frames, we collect a new dataset which we call Random Related Video Views (R2V2). We use the following fast and automated procedure to generate the images in our dataset. Using this procedure, we are able to construct R2V2 in under a day on a single machine.

1. Use YouTube Search to find videos for a set of queries, and download the top K videos licensed under the Creative Commons. In practice we use the ImageNet 1K classes.
2. Filter out videos with static images using a simple threshold over the percent of pixels which change between two frames. This removes videos of static images, which is common for music uploads.
3. Pick a random point in the video and extract T images with a gap of G seconds between each image. In practice $T = 4$ and $G = 5$.

Dataset	Number of Images (Train)	Number of Videos (Train)	Number of Categories	Mean Image Size
ImageNet 1K [7]	1.3 M	0	1000	(428, 406)
YouTube 8M [1]	0	3.7 M	3862	-
Kinetics 400 [17]	0	0.22 M	400	-
GOT-10k [16]	1.4 M	>9 K	563	(1600, 912)
R2V2 (Ours)	0.96 M	0.24 M	-	(467, 280)

Table 1: A comparison of various image and video datasets. While we have neither the most images nor the most videos, we provide good diversity between videos which is crucial for learning a strong, generic image representation. GOT-10k [16] training set contains 9,000 video clips, but multiple clips may originate from a single source video.

Using ImageNet synsets for search queries provides reasonable visual diversity, but could be substituted with another set of queries. While we acknowledge that using YouTube's Search feature is not truly random, this procedure resulted in significantly more diverse samples than using existing datasets like YouTube8M [1] which is heavily unbalanced with unnatural videos like "Video Games" and "Cartoons." We do no additional data cleaning to ensure that the videos or extracted images actually contain the search term (many do not), nor do we search for "high interest" video segments as in Misra *et al*. [26]. We also discard the search term itself as a form of supervision. We find that a gap of 5 seconds between each saved image typically results in visually distinct but semantically related images. Too much shorter results in images which are less individually distinct, and too much longer may result in large and unpredictable changes. A sample from each dataset can be seen in the supplemental material.

We compare R2V2 with other popular datasets in Table 1. Because our dataset is constructed automatically, we can easily gather more data (more frames per video, more videos overall). In this work we limit the scale to roughly that of comparable datasets.

3.2 Noise Contrastive Estimation (NCE) Learning

Given a dataset of diverse video frames, we learn a representation which takes advantage of the structure of the data. We choose the Noise Contrastive Estimation technique [10] which has been popular in many recent works [2, 6, 11, 13, 14, 25, 33, 38], augmented with temporal supervision.

The standard NCE implementation (used in [6, 2, 14]) uses the following procedure. First, a batch of anchor images A are selected. Second, a batch of positive images P are selected, one for each anchor. Positive matches for one example are reused as negatives for the other samples without the need to recompute the features. The NCE loss for a single batch is shown in equation 1 where $sim(X, Y)$ is any similarity metric between the two inputs. Gradients flow through the positive pairs (pulling the vectors together) as well as the negative pairs (pushing the vectors away from each other).

$$\mathcal{L}_{\text{NCE}} = -\frac{1}{n}\sum_{i=1}^{n}\log\frac{e^{sim(A_i,P_i)}}{\sum_{j=1}^{n}e^{sim(A_i,P_j)}} \tag{1}$$

As in other works, we use the cosine similarity of the feature embeddings of the data points (as seen in equation 2) as the similarity metric due to its computational efficiency [2, 6, 11, 13, 25, 38]. The similarity is rescaled by a temperature vector τ to create peaked softmax distributions. f and g are neural networks.

$$sim(X, Y) = \tau * \frac{f(X)}{\|f(X)\|} \cdot \frac{g(Y)}{\|g(Y)\|} \tag{2}$$

3.3 Multi-Frame NCE

All recent works perform some sort of transformation on a single image to create Anchor-Positive pairs for the NCE loss [2, 6, 11, 13, 25, 38]. We refer to this as "Same Frame." We differ from these works by using multiple images from a single video to form our pairs. This allows our network to see truly different views, deformations, similar objects, and larger scene changes. Additionally, this encodes temporal consistency as the semantic contents of a video are unlikely to change suddenly. For example in a video of two cats playing, the camera may focus on one cat, or may even pan to a previously unseen dog, but it is unlikely to pan to a shark. Note that in practice, we select frames with replacement, so it is still possible to pair an image with itself, making our potential pairs a strict superset of those in prior works.

3.4 Memory Banks and Momentum Contrast

NCE-based methods benefit greatly from large pools of negatives because this increases the likelihood of finding at least one hard negative for each positive example. In some works [25, 38], negatives are sampled from a large memory bank which was filled with earlier outputs of the network. The NCE loss can be modified to use negatives from prior batches as shown in equation 3 for a memory bank of negatives $N_{1...m}$.

$$\mathcal{L}_{\text{NCE}} = -\frac{1}{n}\sum_{i=1}^{n} \log \frac{e^{sim(A_i, P_i)}}{e^{sim(A_i, P_i)} + \sum_{j=1}^{m} e^{sim(A_i, N_j)}} \tag{3}$$

As the network trains, its output distribution will shift. A potential issue when using a memory bank is the network learns a simple classifier between the current distribution and an old one. Momentum Contrast (MoCo) [11] alleviates this issue by using a quickly updating primary network (f in equation 2) and a slowly updating secondary network (g). f is updated based on the NCE loss in equation 3 and g is updated using a momentum rule $g \leftarrow \alpha g + (1 - \alpha) f$. The memory bank is filled with previous outputs from the slowly changing network g, reducing the likelihood that f will be able to learn a simple recent batch/old batch classifier. For more details, see [11].

3.5 Multi-Pair NCE

By using MoCo, we increase the number of useful negatives without a large computational cost. Yet MoCo only uses n positive pairs per batch of size n. We can further

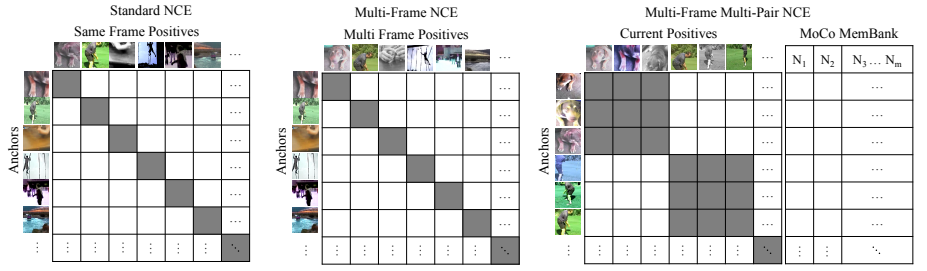


Fig. 2: Left: Standard NCE using "Same Frame" where all correct pairs come from the same image. Middle: Standard NCE using "Multi-Frame" where correct pairs come from the same video. Right: Multi-Frame Multi-Pair NCE which uses more than one positive pair per video, resulting in more positives per batch. The gray boxes indicate the true match pairs. The MoCo Memory Bank adds more negatives for each anchor.

increase the number of positives per batch (while holding batch size constant) by simply selecting v videos and k samples from each video where $k = \frac{n}{v}$. By computing the pairwise similarity between each pair, we reuse each positive sample k times, resulting in $k^2 v = \frac{n^2}{v}$ positives per batch. In the extreme, every sample from a batch could belong to the same class, resulting in n^2 positives, however this causes noisier, more extreme gradients which makes training unstable. Using a simple block-diagonal mask as shown in Figure 2 and Algorithm 1, we can efficiently compute the similarities and NCE loss both between elements of the batch and across a memory bank, achieving a large number of positive comparisons per batch while retaining a large negative size. In practice, we notice no meaningful computational cost to this approach. We refer to this full method using Multi-Pair on video data and the Multi-Frame learning procedure as Video Noise Contrastive Estimation (VINCE).

Using clusters of positives has the additional benefit of forcing each feature to match with multiple other features at once. For videos, this means a representation for a single image will be pulled towards some global video feature, resulting in a more consistent representation over a video. For single images, this more strongly enforces invariance to data augmentation.

3.6 Implementation Details

For training our representation we use ResNet18 [12] up to the global average pooling followed by a fully connected layer (512 x 512), Leaky-ReLU and a final embedding layer (512 x 64). The features are L2-normalized and multiplied by $\tau = \frac{1}{0.07}$ as in MoCo [11]. We use 8 GPUs, a training batch size of 256 and a MoCo Memory Bank of size 65536 and a g-network momentum of 0.999. For multi-GPU training we employ the Shuffle-BN technique shuffling both the anchors and the positives to reduce the correlation between batches filled with multiple images from the same video. We use SGD with a learning rate of 0.03, momentum=0.9, and weight decay=0.0001. All of these hyperparameters are shared with MoCo [11]. All methods are trained for approximately 450k iterations. When selecting frames from a video, we pick with replacement. In addition to the natural augmentation, we perform standard data augmentation (color jitter,

Algorithm 1 Python-style pseudo code for Multi-Pair NCE.

```
def multi_pair_nce(
        f_output,                                          # [v, k, d] output of f
      encoder
        g_output,                                          # [v, k, d] output of g
      encoder
        moco_mem,                                          # [m, d]
        mask,                                              # [n, (n + m)] block diagonal
      boolean matrix
        temperature):                                      # [1]

    f_output = f_output.reshape(v * k, d)                  # [n, d]
    g_output = g_output.reshape(v * k, d)                  # [n, d]
    compare = concatenate((g_output, moco_mem), axis=0)    # [(n + m), d]
    similarities = matmul(f_output, compare.T)             # [n, (n + m)]
    similarities /= temperature
    pos_similarities = similarities[mask]                  # [n, k]
    neg similarities = similarities[!mask]                 # [n, (n + m - k)]
    exp_pos_sim = exp(pos_similarities)                    # [n, k]
    normalizing_constant = broadcast(                      # [n, k]
        reduce_sum(exp(neg_similarities), axis=1),
                  shape(numerator))))
    score = exp_pos_sim / (exp_pos_sim + normalizing_constant)
    loss = -mean(log(score))
    return loss
```

In practice we use the Log-Sum-Exp trick for numerical stability but omit here for clarity. `broadcast` repeats the input until it matches the provided dimensions. `!mask` flips the booleans of each point in the mask.

cropping, flipping) on the inputs. This prevents the network from relying too heavily on shared video statistics like mean frame color. After cropping, all images are resized to 224×224. It is worth noting that both VINCE and our data (R2V2) can be used with other network architectures or learning algorithms such as AMDIM [2], PIRL [25], or SimCLR [6]. We choose ResNet18 and MoCo due to implementation simplicity and relatively low computational constraints.

4 Experiments

We evaluate our method (**VINCE**) on both single-image and temporal tasks by freezing our learned representation and adding a small network (in most cases a single linear layer) for adaptation to new end-tasks. Our learned representation transfers well to a variety of visual tasks, especially tasks which require temporal reasoning. To show this, we compare with multiple strong baselines:

- **MoCo-IN**: Network pretrained on ImageNet [7] using the MoCo algorithm.
- **MoCo-R2V2**: Network pretrained on R2V2 using the MoCo algorithm. This uses exactly the same data as VINCE but the Same Frame technique described in 3.3. We also prevent multiple images from the same video being in the MoCo Memory Bank at the same time.
- **Sup-IN**: Network pretrained on fully supervised ImageNet.

To validate the benefits of unsupervised, uncleaned video, we additionally compare against an unsupervised, uncleaned image dataset. Since ImageNet itself required time, effort, and money to create, we construct a new static image dataset analogous to our video dataset. Specifically, we search Google Images for the ImageNet synsets and download the top K results for each category. We refer to this as **MoCo-G**).

	Image Classification ImageNet[7]	**Scene Classification SUN Scenes[39]**	**Action Recognition Kinetics 400[17]**	**Tracking OTB 2015[37]**	
Trained Layer(s)	Linear	Linear	LSTM → Linear	1x1 Conv	
Metric	Accuracy (Top 1)	Accuracy (Top 1)	Accuracy (Top 1)	Precision	Success
Sup-IN	**0.696**	0.491	0.207	0.557	0.396
MoCo-IN	0.447	0.487	0.336	0.583	0.429
MoCo-G	0.393	0.444	0.313	0.551	0.413
MoCo-R2V2	0.358	0.450	0.318	0.555	0.403
VINCE (Ours)	0.400	**0.495**	**0.362**	**0.629**	**0.465**
Relative Gain over MoCo-R2V2	11.91%	9.93%	13.85%	13.33%	15.38%

Table 2: Comparison of representation performance across a variety of end tasks. We show improvements over MoCo trained on the same data on all tasks, and outperform MoCo trained on ImageNet as well as supervised pretraining on ImageNet on all tasks but ImageNet itself. Each representation uses the ResNet18 convolutional architecture, sharing weights across all tasks. Linear (for Kinetics LSTM → Linear) classifiers are the only learned weights for each end task.

4.1 Target Tasks

We compare each method on several diverse end-tasks. Results for these tasks are shown in Table 2. We train all end-task models using Adam [18] and a shared learning rate schedule per task. For each dataset, we use standard data augmentation approaches (crop, flip, color jitter except for tracking). One overall trend to note is the relative gain over MoCo-R2V2. If the single-frame algorithm performed as well as our multi-frame method on temporal tasks, it would indicate minimal temporal understanding. However, the relative gain of VINCE over MoCo-R2V2 on Kinetics (13.85%) and tracking (13.33% and 15.38%) are higher than those on single-frame tasks, showing that our method can incorporate temporal cues.

ImageNet: For this task, we use our frozen learned representations, adding a single linear layer after the global average pool. Although none of the methods match the fully supervised performance of ResNet18 on ImageNet, they do achieve reasonable performance given only single linear layer. It is unsurprising that MoCo pretrained on ImageNet images (MoCo-IN) outperforms our method (0.447 vs 0.400) due to the domain shift between pretrain and end-task. However MoCo pretrained on R2V2 (MoCo-R2V2) suffers nearly a $2\times$ drop in accuracy (8.9%) compared to our method (4.7%), indicating that pretraining on multi-frame matching provides a clear benefit over single frame pairs. Our method, which has never seen an image from ImageNet before, still learns a representation which generalizes well to this new type of data. Even drawing images from a similar class distribution (MoCo-G) does not outperform our method.

SUN Scenes: SUN Scenes is a classification dataset in which each image is categorized into one of 397 possible scene types such as airplane cabin, bedroom, and coffee shop. Again we train a single linear layer on top of each pretrained network. This data is quite similar to ImageNet in that each image is well-curated and contains single, unambiguous subjects. As such, the ImageNet fully supervised baseline transfers quite well

to SUN Scenes. However VINCE outperforms Sup-IN by a small margin. Again we note a large improvement of VINCE over MoCo-R2V2 (0.495 vs. 0.450). This shows that our method learns to recognize not just the main subject of an image but also the surrounding scene, which requires a richer understanding of the world.

Kinetics 400: This dataset consists of 10 second clips from YouTube videos and action labels for each segment. We first download each video and subsample each clip to one frame per second (10 frames per clip). We train a single layer LSTM [15] followed by a single linear layer to predict the action category for each segment. Kinetics acts as a crucial test to evaluate whether our model learns temporal cues. VINCE greatly outperform all other methods, whereas traditional baselines such as fine-tuning supervised ImageNet do not adapt well at all. This shows that contrary to popular belief, representations pretrained on ImageNet many not be a good fit for other visual domains, especially on temporal tasks.

Object Tracking Benchmark (OTB) 2015: OTB 2015 is a popular tracking dataset. Given an initial bounding box around an arbitrary object in the first image, a model must locate the object in the following frames. We use the SiamFC [3] tracking algorithm on top of our learned representation. SiamFC first crops the initial bounding box and extracts spatial features using a CNN. For each frame, it localizes the object by extracting spatial features on the full frame and convolving the template features with the full image features. This process is similar to template matching [4] but in deep feature space. For a more complete explanation, see [3].

To extract these features, we use the outputs of each model from before the average pooling. We additionally use dilated convolutions rather than strides for the second and third ResNet18 block to preserve spatial information even though the initial representation was pretrained using strides. We add a single 1x1 convolution layer to each representation. As OTB 2015 is only a set of test videos, we train on the GOT-10k dataset [16], a dataset of 9000 training clips and 1.4 million images.

OTB is evaluated using two metrics – precision and success. Precision measures the percentage of frames where the (normalized) center error is less than a certain threshold, using an area-under-the-curve evaluation. Similarly, success measures the percentage of frames where the Intersection over Union is more than a certain threshold, again using the area-under-the-curve.

A representation which works well for SiamFC would have the property that the cross correlation of two images of the same object is high, but the cross correlation of two different objects, or a poorly cropped image of the same object, is low. Pretraining our representations on multiple frames from the same video coincides quite well with the first objective, however since we use cropped data augmentations, the representations tend to be somewhat invariant to poorly-cropped candidates. Still, the models perform quite well across a variety of difficult tracking instances. Our model transfers significantly better than all other methods indicating a clear benefit to using temporal cues during pretraining.

Images Per Video	Test Task ImageNet	SUN Scene	Kinetics 400	OTB 2015 Precision	OTB 2015 Success
1: Same Frame	0.358	0.450	0.318	0.555	0.403
2: Multi-Frame	0.381	0.478	**0.361**	0.622	**0.464**
8: Multi-Frame Multi-Pair	**0.400**	**0.495**	**0.362**	**0.629**	**0.465**

Table 3: Method ablation for VINCE. We compare using one source image with two augmentations (the standard approach), two different images, or a set of different images. Using Multi-Frame results in a large boost across the board. Multi-Frame Multi-Pair further increases the power of the representation. Note that all methods use the entire dataset, but only Multi-Frame methods use multiple images from a video within one batch.

4.2 Method Ablation

We validate the effectiveness of Multi-Frame (Sec. 3.3) and Multi-Pair (Sec. 3.5) learning by ablating the number of images from each video used in a batch of comparisons and show the results in Table 3. The first row is equivalent to the procedure done in MoCo [11] i.e. the anchor and positive pairs are two data augmentations of the same image. The second row uses the MoCo procedure as well, however the anchors and positives may be from different images from the same video. The third row uses our Multi-Pair NCE method taking 4 positives and 4 anchors from each video, resulting in 16 positive pairs. A pictorial representation can be seen in Figure 2. Note that when selecting images for row 2 and 3, we use sampling with replacement, making our method a strict super-set of MoCo.

We observe across-the-board improvements from both modifications to the MoCo approach. The majority of the improvement comes from using two non-identical frames for matching, but we still gain an additional improvement from using Multi-Pair NCE. Our intuition is that using the Multi-Pair NCE creates gradients that pull each feature towards a global video representation whereas the standard NCE remains more instance-based, only moving a representation in one direction at a time. Thus, we would expect the Multi-Pair NCE features to be more holistically semantic whereas the standard NCE may retain more uniquely identifying features. In fact, we observe a larger performance gap on the more semantic ImageNet and SUN Scene tasks. In contrast, because the Kinetics model uses an LSTM to reason over all input images at once, instance-level features are equally useful as global video features for overall accuracy.

4.3 Pretraining Data Ablation

In Table 4 we explore the effect of different pretraining datasets on end-task performance. For each experiment, we use VINCE but use video data from three different sources: our method of searching ImageNet synset queries, using the URLs from YouTube 8M [1], and using the URLs from Kinetics 400 [17]. Our YouTube8M(YT8M) pretraining data uses the same filtering procedure as R2V2 and contains 5.8 million images from 1.4 million videos. As noted in Table 2 MoCo-IN results, using the same dataset for pretraining as the end-task results in a boost in performance on that specific task but does not indicate that the representation will be better on all tasks. We see this

Pretraining Data	Test Task ImageNet	SUN Scene	Kinetics 400	OTB 2015 Precision	OTB 2015 Success
R2V2 IN-Queries	**0.400**	**0.495**	0.362	0.629	0.465
R2V2 YT8M URLs	0.367	0.478	0.343	**0.667**	**0.492**
Kinetics 400 URLs	0.368	**0.494**	**0.390**	0.612	0.456

Table 4: Pretraining data ablation for VINCE. Each method uses exactly the same training setup, only substituting one data source for another. Since R2V2 uses ImageNet search queries, it outperforms the others on ImageNet. Similarly, pretraining on Kinetics 400 videos results in better end performance on Kinetics.

trend is true again when pretraining on Kinetics data. Similarly, since R2V2 uses ImageNet synset for search queries, pretraining on it performs better on ImageNet than the other less-aligned datasets.

In general, this would indicate that given a large enough set of diverse videos, pretraining directly on the unlabeled source data would result in the best performing representation on that data. If this is not possible, then pretraining on a large external source of data may still result in a useful representation. It also indicates that the VINCE method works well on a variety of different pretraining datasets. The increased performance on tracking when using YT8M data could be explained by it simply having access to a larger number of video sources and frames. For generic object tracking, class diversity may be less important than number of samples because the class identity is ignored.

4.4 Qualitative Results

We additionally provide two qualitative analyses to better understand the success and failure cases of VINCE: Nearest Neighbors, and t-SNE.

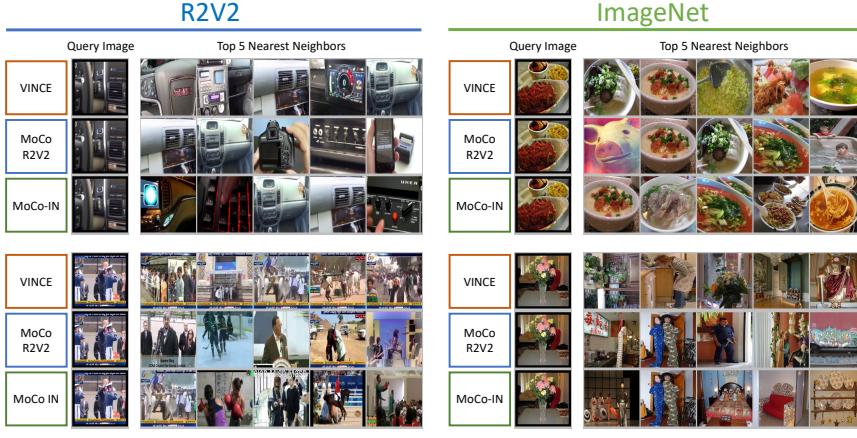


Fig. 3: Nearest neighbor results for a sampling of query images from R2V2 and ImageNet using various models. VINCE shows a clear understanding of each image and finds highly relevant neighbors.

Fig. 4: t-SNE embedding of images from R2V2 test set. A High-Res version of the t-SNE embedding will be included in the supplementary material.

Nearest Neighbors: We additionally query ImageNet Val and a set of test videos for nearest neighbor matches, taking at most one neighbor per video. We visualize the top 5 neighbors for VINCE, MoCo-R2V2, and MoCo-IN in Figure 3. We observe that VINCE seems to understand the semantics of an image more than MoCo-R2V2 and MoCo-IN. For instance, although MoCo-R2V2 and MoCo-IN find other control panels and buttons in query 1, they do not make the scene-level connection to car interiors as well as VINCE does. Query 2 shows an interesting quirk case of our method. Rather than matching the semantics of the image, VINCE relies on the news logo as a differentiating feature due to its discriminative nature. Each image in VINCE's query 2 results is from a separate video, but from the same news source. For the ImageNet queries, despite never seeing ImageNet inputs during pretraining, VINCE is able to find good matches as well as MoCo-IN which was trained using only ImageNet inputs.

t-SNE: Using a set of held-out video frames, we project the 64-D embedding space from VINCE to 2D using t-SNE [21] and visualize the formed clusters in Figure 4. Not only does this assist in verifying the quality of the embedding, it also serves as a visual method for evaluating the diversity of the dataset itself. The largest of the clusters seems to be the face cluster. YouTube is full of videos of people looking and talking directly to

a camera, and our random subset reflects this pattern. Other interesting, yet unexpected clusters emerge as well such as cats (YouTube loves cats), hands (demo videos), and food (cooking videos).

5 Conclusions

In this work we introduced Video Noise Contrastive Estimation, a process for using unlabeled videos to learn an unsupervised image representation. By training on multiple images from the same video instance, we learn from more natural changes such as deformation and viewpoint change rather than 2D artificial augmentations. To learn from a large variety of diverse video clips, we collect Random Related Video Views in a completely automated fashion. Using geometric video cues like structure from motion and optical flow could provide an even richer dataset, but we leave this as a promising future direction. We show across-the-board improvements over the recently proposed MoCo [11] technique on a wide variety of tasks, and we believe Video Noise Contrastive Estimation will extend to other unsupervised methods such as SimCLR [6] and PIRL [25] as well as other end-tasks. As representation learning techniques improve, we believe that videos – rather than images – will prove an invaluable resource for pushing the state-of-the-art forward.

6 Acknowledgements

This work is in part supported by the Nvidia Graduate Research Fellowship, NSF-NRI-1637479, NSF IIS 1652052, IIS 17303166, DARPA N66001-19-2-4031, 67102239, and gifts from Allen Institute for Artificial Intelligence.

References

1. Abu-El-Haija, S., Kothari, N., Lee, J., Natsev, P., Toderici, G., Varadarajan, B., Vijayanarasimhan, S.: Youtube-8m: A large-scale video classification benchmark. arXiv preprint arXiv:1609.08675 (2016)
2. Bachman, P., Hjelm, R.D., Buchwalter, W.: Learning representations by maximizing mutual information across views. In: Advances in Neural Information Processing Systems. pp. 15509–15519 (2019)
3. Bertinetto, L., Valmadre, J., Henriques, J.F., Vedaldi, A., Torr, P.H.: Fully-convolutional siamese networks for object tracking. In: European conference on computer vision. pp. 850–865. Springer (2016)
4. Briechle, K., Hanebeck, U.D.: Template matching using fast normalized cross correlation. In: SPIE Defense + Commercial Sensing (2001)
5. Chechik, G., Sharma, V., Shalit, U., Bengio, S.: Large scale online learning of image similarity through ranking. Journal of Machine Learning Research **11**(Mar), 1109–1135 (2010)
6. Chen, T., Kornblith, S., Norouzi, M., Hinton, G.: A simple framework for contrastive learning of visual representations. arXiv preprint arXiv:2002.05709 (2020)
7. Deng, J., Dong, W., Socher, R., Li, L.J., Li, K., Fei-Fei, L.: Imagenet: A large-scale hierarchical image database. In: IEEE conference on computer vision and pattern recognition. pp. 248–255. Ieee (2009)

8. Devlin, J., Chang, M.W., Lee, K., Toutanova, K.: Bert: Pre-training of deep bidirectional transformers for language understanding. In: NAACL-HLT (2019)
9. Godard, C., Mac Aodha, O., Brostow, G.J.: Unsupervised monocular depth estimation with left-right consistency. In: IEEE Conference on Computer Vision and Pattern Recognition (July 2017)
10. Gutmann, M., Hyvärinen, A.: Noise-contrastive estimation: A new estimation principle for unnormalized statistical models. In: Thirteenth International Conference on Artificial Intelligence and Statistics. pp. 297–304 (2010)
11. He, K., Fan, H., Wu, Y., Xie, S., Girshick, R.: Momentum contrast for unsupervised visual representation learning. arXiv preprint arXiv:1911.05722 (2019)
12. He, K., Zhang, X., Ren, S., Sun, J.: Deep residual learning for image recognition. In: IEEE conference on computer vision and pattern recognition. pp. 770–778 (2016)
13. Hénaff, O.J., Razavi, A., Doersch, C., Eslami, S., Oord, A.v.d.: Data-efficient image recognition with contrastive predictive coding. arXiv preprint arXiv:1905.09272 (2019)
14. 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: International Conference on Learning Representations (2019), https://openreview.net/forum?id=Bklr3j0cKX
15. Hochreiter, S., Schmidhuber, J.: Long short-term memory. Neural Computation **9**, 1735–1780 (1997)
16. Huang, L., Zhao, X., Huang, K.: Got-10k: A large high-diversity benchmark for generic object tracking in the wild. IEEE Transactions on Pattern Analysis and Machine Intelligence p. 1–1 (2019). https://doi.org/10.1109/tpami.2019.2957464, http://dx.doi.org/10.1109/TPAMI.2019.2957464
17. Kay, W., Carreira, J., Simonyan, K., Zhang, B., Hillier, C., Vijayanarasimhan, S., Viola, F., Green, T., Back, T., Natsev, P., et al.: The kinetics human action video dataset. arXiv preprint arXiv:1705.06950 (2017)
18. Kingma, D.P., Ba, J.: Adam: A method for stochastic optimization. arXiv preprint arXiv:1412.6980 (2014)
19. Kramer, M.A.: Nonlinear principal component analysis using autoassociative neural networks. AIChE journal **37**(2), 233–243 (1991)
20. Li, Z., Dekel, T., Cole, F., Tucker, R., Snavely, N., Liu, C., Freeman, W.T.: Learning the depths of moving people by watching frozen people. In: IEEE Conference on Computer Vision and Pattern Recognition. pp. 4521–4530 (2019)
21. Maaten, L.v.d., Hinton, G.: Visualizing data using t-sne. Journal of machine learning research **9**(Nov), 2579–2605 (2008)
22. Mahajan, D., Girshick, R., Ramanathan, V., He, K., Paluri, M., Li, Y., Bharambe, A., van der Maaten, L.: Exploring the limits of weakly supervised pretraining. In: European Conference on Computer Vision. pp. 181–196 (2018)
23. Meister, S., Hur, J., Roth, S.: Unflow: Unsupervised learning of optical flow with a bidirectional census loss. In: Thirty-Second AAAI Conference on Artificial Intelligence (2018)
24. Mikolov, T., Sutskever, I., Chen, K., Corrado, G.S., Dean, J.: Distributed representations of words and phrases and their compositionality. In: Advances in neural information processing systems. pp. 3111–3119 (2013)
25. Misra, I., van der Maaten, L.: Self-supervised learning of pretext-invariant representations. arXiv preprint arXiv:1912.01991 (2019)
26. Misra, I., Zitnick, C.L., Hebert, M.: Shuffle and learn: unsupervised learning using temporal order verification. In: European Conference on Computer Vision. pp. 527–544. Springer (2016)
27. Noroozi, M., Favaro, P.: Unsupervised learning of visual representations by solving jigsaw puzzles. In: European Conference on Computer Vision. pp. 69–84. Springer (2016)

28. Pinto, L., Gandhi, D., Han, Y., Park, Y.L., Gupta, A.: The curious robot: Learning visual representations via physical interactions. In: European conference on computer vision (2016)
29. Radford, A., Metz, L., Chintala, S.: Unsupervised representation learning with deep convolutional generative adversarial networks. arXiv preprint arXiv:1511.06434 (2015)
30. Radford, A., Wu, J., Child, R., Luan, D., Amodei, D., Sutskever, I.: Language models are unsupervised multitask learners (2019)
31. Schmidt, T., Newcombe, R., Fox, D.: Self-supervised visual descriptor learning for dense correspondence. IEEE Robotics and Automation Letters **2**(2), 420–427 (2016)
32. Srivastava, N., Mansimov, E., Salakhudinov, R.: Unsupervised learning of video representations using lstms. In: International conference on machine learning. pp. 843–852 (2015)
33. Tian, Y., Krishnan, D., Isola, P.: Contrastive multiview coding. arXiv preprint arXiv:1906.05849 (2019)
34. Vondrick, C., Shrivastava, A., Fathi, A., Guadarrama, S., Murphy, K.: Tracking emerges by colorizing videos. In: European Conference on Computer Vision. pp. 391–408 (2018)
35. Wang, X., Gupta, A.: Unsupervised learning of visual representations using videos. In: IEEE International Conference on Computer Vision. pp. 2794–2802 (2015)
36. Wang, X., Jabri, A., Efros, A.A.: Learning correspondence from the cycle-consistency of time. In: IEEE Conference on Computer Vision and Pattern Recognition. pp. 2566–2576 (2019)
37. Wu, Y., Lim, J., Yang, M.H.: Object tracking benchmark. IEEE Transactions on Pattern Analysis and Machine Intelligence **37**(9), 1834–1848 (2015)
38. Wu, Z., Xiong, Y., Yu, S.X., Lin, D.: Unsupervised feature learning via non-parametric instance discrimination. In: IEEE Conference on Computer Vision and Pattern Recognition. pp. 3733–3742 (2018)
39. Xiao, J., Hays, J., Ehinger, K.A., Oliva, A., Torralba, A.: Sun database: Large-scale scene recognition from abbey to zoo. In: IEEE Conference on Computer Vision and Pattern Recognition. pp. 3485–3492. IEEE (2010)
40. Zhang, R., Isola, P., Efros, A.A.: Colorful image colorization. In: European conference on computer vision. pp. 649–666. Springer (2016)

Appendix 1 t-SNE

We provide the full resolution t-SNE [21] image for further inspection. Best viewed on a screen.

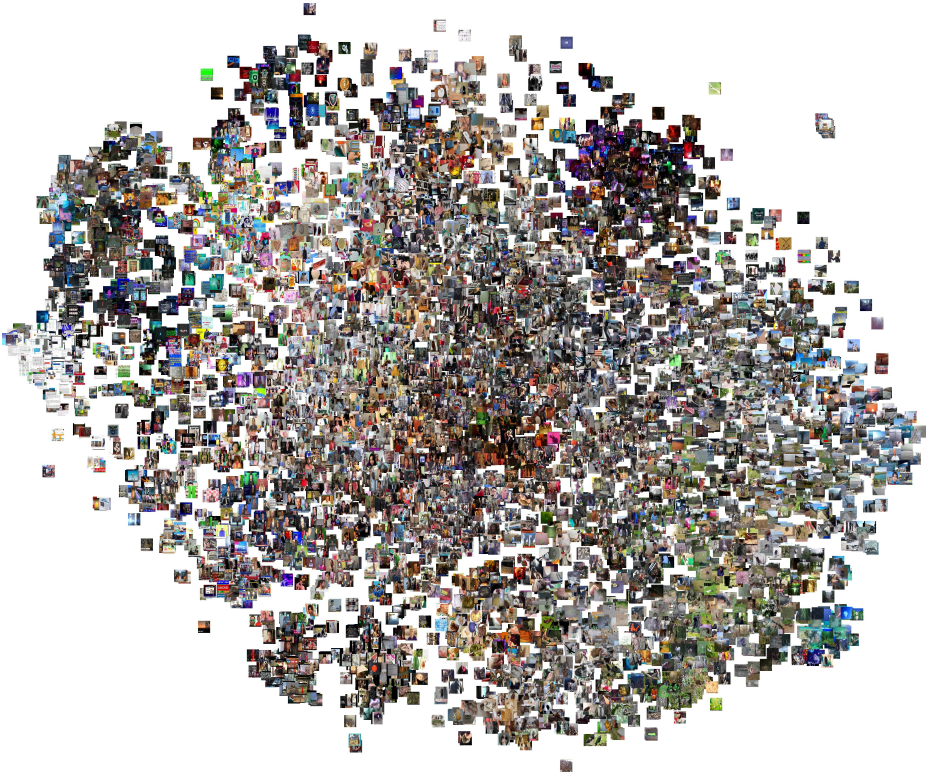


Fig. F1: Full Resolution t-SNE embedding of images from R2V2 test set.

Appendix 2 Dataset Samples

Existing datasets such as YouTube8M [1] and Kinetics400 [17] provide a large number of YouTube links over a diverse set of videos. However, these dataset are highly unbalanced and contain many videos undesirable for learning strong visual representations. For example, the second most common category in YouTube8M, comprising 540k of the 3.7 million training videos is “Video Game,” and the fifth is “Cartoon” with 240k. The category “Minecraft” itself has over 57,000 videos in the dataset whereas the category “Pear” has only 138. Kinetics contains only videos of humans performing actions.

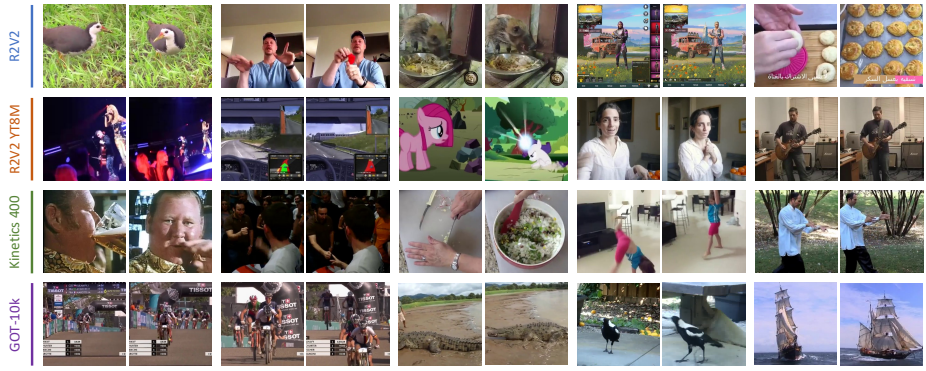


Fig. F2: Random sampling of pairs of images from videos in each dataset. In GOT-10k, sometimes different video clips are segments from the same original video as seen in the first and second sample. Images are square cropped for visualization purposes only.

Alternative datasets such as GOT-10k [16] provide a comparatively small number of videos but with dense annotations (in GOT-10k's case for object tracking).

Appendix 2.1 Random Related Video Views Samples

We show more samples from R2V2 in Figure F3. Videos each have four images 150 frames apart. Each separate video (outlined in blue) lists its corresponding YouTube link.

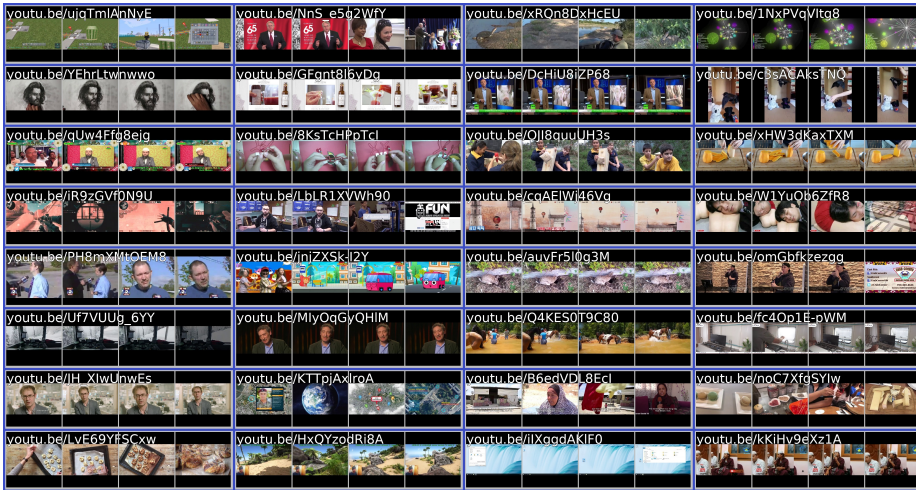
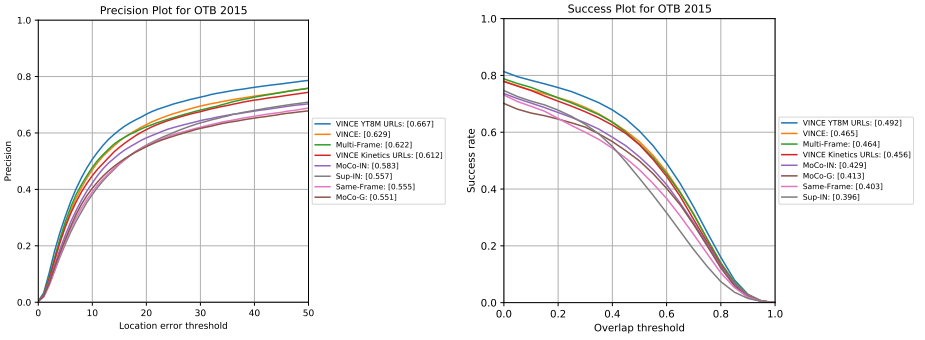


Fig. F3: Sample from Random Related Video Views (train set).



(a) (b)

Fig. F4: Precision (a) and Success (b) plots for OTB 2015 for various backbones.

Appendix 3 Precision and Success Plots for OTB 2015

We provide full breakdowns of the Precision and Success of each method on OTB 2015 [37]. The values in the legend correspond to the (mean) area under the curve for each method.