Evolving Losses for Unsupervised Video Representation Learning

AJ Piergiovanni, Anelia Angelova, Michael S. Ryoo
Research at Google
{ajpiergi, anelia, mryoo}@google.com

Abstract

We present a new method to learn video representations from large-scale unlabeled video data. Ideally, this representation will be generic and transferable, directly usable for new tasks such as action recognition and zero or few-shot learning. We formulate unsupervised representation learning as a multi-modal, multi-task learning problem, where the representations are shared across different modalities via distillation. Further, we introduce the concept of loss function evolution by using an evolutionary search algorithm to automatically find optimal combination of loss functions capturing many (self-supervised) tasks and modalities. Thirdly, we propose an unsupervised representation evaluation metric using distribution matching to a large unlabeled dataset as a prior constraint, based on Zipf’s law. This unsupervised constraint, which is not guided by any labeling, produces similar results to weakly-supervised, task-specific ones. The proposed unsupervised representation learning results in a single RGB network and outperforms previous methods. Notably, it is also more effective than several label-based methods (e.g., ImageNet), with the exception of large, fully labeled video datasets.

1. Introduction

Video representation learning is an important problem which benefits high-level perception tasks including action recognition and video object detection [40, 6, 41]. It has many key applications, such as web-video retrieval, robot perception, and smart homes and cities. However, learning visual representations generally requires a large number of labeled training examples. This is even more so for videos, as videos are higher-dimensional input than images and video CNNs have more learnable parameters than 2D ones. Simultaneously, videos are more expensive to collect and annotate than images, as they require additional, and often ambiguous, temporal annotations [34]. Additionally, in rare event detection, very few examples may be available to begin with. Thus, obtaining a good video representation without relying on domain specific, annotated video samples, has significant impact on real-world scenarios where large-scale video data curation and labeling is prohibitive.

In this paper, we present a new, principled method for unsupervised learning of video representations from unlabeled video data. It is based on the observation that the optimized combination of multiple self-supervised tasks\(^1\), which are additionally encouraged by multi-modal distillation, is often sufficient to learn good feature representations. Importantly, we demonstrate that such combination could be found without per-class or per-video labeling by instead matching the representation statistics to a general power distribution of video classes, e.g., Zipf’s law [50].

Our approach is to train the network so that its intermediate representations reflect not just information directly obtained from its own input modality (e.g., RGB image) but also information from different modalities (e.g.,

\(^1\)In this work we use unsupervised and self-supervised interchangeably.
Figure 2: The multi-task, multi-modal, unsupervised representation learning framework. Each modality is trained to optimize a set of tasks. Distillation regularization loss terms ‘infuse’ each modality’s information into the main RGB network (drawn center). We evolve the loss function to automatically find the weights for each task and distillation location, via an unsupervised objective. The goal is to obtain representation from a single RGB network that transfers to recognition tasks.

Our goal is to find video feature representations, based on a single RGB network, that can seamlessly improve supervised or unsupervised tasks without additional annotations. The main contributions are:

- Formulation of unsupervised learning as multi-modal, multi-task learning, including distillation tasks to transfer features across modalities into a single-stream network. Once learned, it allows for faster representation computation while still capturing multi-modal features.
- Evolutionary search for a loss function that automatically combines self-supervised and distillation tasks that are beneficial for unsupervised representation learning.
- Introduction of an unsupervised representation evaluation metric based on power law distribution matching, which requires no labels and performs similarly to the label-guided one.

This work makes the surprising finding that large amounts of unlabeled data, combined with self-supervised tasks and the power law distribution matching, produces very powerful feature representations which are only rivaled by large datasets with very extensive data labeling: Our feature representations (obtained with zero labels) outperform ImageNet pre-training, and pre-training on small and medium-size labeled video datasets; it is only outperformed by Kinetics pre-training with full annotations based on human labeling of more than 200,000 videos. Further, the proposed representations outperform Kinetics training, when fine-tuned with Kinetics labels. We refer to the model as ‘ELo’, as it is based on evolving unsupervised losses.
2. Related Work

Unsupervised Video Representation Learning: Obtaining labeled video data is expensive, and unlabeled video data is plentiful, and there have been many methods proposed for self-supervised learning of video representations. Some tasks take advantage of the temporal structure in videos, such as predicting if frames appear in order, reverse order, shuffled, color-consistency across frames, etc. [12, 26, 30, 24, 31, 47, 19, 21, 45, 46, 42]. Other work has explored using the spatial structure present in images, such as predicting relative spatial location of image patches [28] or tracking patches over time [44], showing promising results. Reconstruction or prediction of future frames [37], or time-contrastive learning [18, 33] to obtain representations has also been successful. Constrastive losses are also common [39]. Learning representations taking advantage of audio and video features has been explored by predicting if an audio clip is from a video [2] or if audio and video are temporally aligned [29, 7, 22, 3].

Multi-task self-supervised learning has also shown promising results [10, 32, 48], where tasks are assumed to have equal weights and are not multi-modal. Generating weak labels using k-means clustering on CNN features [4, 5] or using clustering with meta-learning [17] has also been explored. In this paper, we propose a generalized approach to unsupervised representation learning, allowing for multi-modal inputs and automatic discovery of the tasks that benefit recognition performance.

Activity Recognition: Activity recognition is a active area of vision research, with a variety of methods proposed [43, 40, 35, 11]. With the introduction of large activity recognition datasets (e.g., Kinetics and Moments in Time [20, 27]), much more accurate deep video CNNs are possible [6]. We here show that they can be further improved by unsupervised representation learning.

3. Method

We formulate unsupervised video representation learning as a combination of multi-task, multi-modal learning. The objective is not only to take advantage of multiple self-supervised tasks for the learning of a (good) representation space, but also to do so across multiple modalities. The idea is that models from synchronized multi-modal data, sharing the same semantic content, will benefit the representation learning of each other. We encourage that via introducing ‘distillation’ losses. At the same time, each modality may have multiple self-supervised tasks with their corresponding losses. Fig. 2 illustrates the multi-task, multi-modal formulation with multiple losses and distillation, Section 3.1 has the details.

To facilitate multi-task, multi-modal learning, importantly, we introduce the new concept of automatically evolving the main loss function. Certain tasks and modalities are more relevant to the final task so the representation needs to focus on those more than the others. The idea is to computationally search for how different multi-task and distillation losses should be combined, instead of constructing a loss function by trial-and-error. We discuss this more in Section 3.2.

One key technical question is how one can guide the evolution without a pre-defined task or a fitness function. We propose an unsupervised method to evaluate each loss function, based on matching of the power law distribution of activity classes (Section 3.2.2).

3.1. Unsupervised multi-modal learning

We construct a CNN for each modality. Each network is trained using several tasks not requiring labeled video data, and the information from each modality is combined using distillation [16] (Fig. 2). More specifically, we take advantage of multiple self-supervised tasks, such as frame reconstruction, future frame prediction, and frame temporal ordering (Section 3.3 discusses them in detail). Each of these tasks will yield an unsupervised loss for training. Learning with multiple self-supervised tasks makes our representations more generic, as they need to generalize to many tasks and are more transferable to unseen tasks.

For each modality, m and its input I_m, we build an embedding network, E_m, which generates an embedded representation of the input: x_m = E_m(I_m). x_m is the feature representation for modality m. Our embedding networks are (2+1)D ResNet-50 models, which take advantage of both 2D spatial convolutions and 1D temporal convolutions to represent videos; they provide state-of-the-art performance on video understanding tasks. As mentioned, for each modality, we consider several learning tasks, for example, frame reconstruction. Each of the tasks per modality has its own loss function. L_{m,t} is the loss from task t and modality m and \{t_1, t_2, ..., t_{N_m}\} is the set of tasks for the modality.

Further, to better take advantage of the multi-modal representations, we use distillation to ‘infuse’ the other modalities into the RGB network at different locations. Our final objective is to train a single RGB network that provides a strong representation for video understanding. Our formulation allows the RGB network to learn representations from various tasks and modalities.

We combine the multi-task losses for unsupervised training, and per each modality, by a weighted sum, and we further combine it with a number of distillation losses \mathcal{L}_d which fuse or synchronize the multiple modalities:

\[
\mathcal{L} = \sum_m \sum_t \lambda_{m,t} \mathcal{L}_{m,t} + \sum_d \lambda_d \mathcal{L}_d
\]

(1)

where \lambda_{m,t} and \lambda_d are the weights of the losses. The weighted sum \mathcal{L} is the loss we use to train the entire model.
3.1.1 Distillation

Distillation was introduced to train smaller networks by matching representation of deeper ones [16], or for generally transferring knowledge from pre-trained networks. Here, we use distillation to ‘infuse’ representations of different modalities into the main, RGB network. Note that we distill representations jointly while training. The distillation losses learn features by transferring information across modalities. More specifically, our formulation allows for the distillation of audio, optical flow and temporal information losses learn features by transferring information across modalities. The weighting supposedly reflects the importance or relevance of each task and modality to the main task; for example the optical flow modality may be important for tracking, whereas audio may give more information for temporal segmentation of videos in certain settings.

3.2.1 Unsupervised loss construction

Instead of manually constructing a loss function, we evolve the loss function by taking advantage of well-established evolutionary algorithms, e.g., [13]. More specifically, our search space consists of all the weights of the loss function, both task weights and distillation weights. Each $\lambda_{m,t}$ or $\lambda_t$ is constrained to be in $[0, 1]$. Our evolutionary algorithm maintains a pool (i.e., population) of individuals where each individual is a set of weight values that compose the final loss function.

3.2.2 Unsupervised Zipf distribution matching

The evolutionary algorithm requires evaluation of the loss function (i.e., fitness measure) at each round to optimize the loss weight coefficients. We propose a new, unsupervised method for this. In order to measure the fitness of each individual (i.e., the set of weights to combine the tasks and modalities to form the final loss), we apply a $k$-means clustering on the representation learned with the corresponding loss function, and analyze the cluster distributions. We first train the network using a smaller subset (100k) of unlabeled, random YouTube videos for 10,000 iterations (using the corresponding loss function). Then we use a subset of random YouTube videos and similarly extract representations $x_{RGB} = E_{RGB}(I)$. Given these representations, we can then cluster them into $k$ clusters.

$k$-means clustering can be viewed as a Gaussian Mixture Model with fixed variances and we calculate probabilities of each feature vector belonging to a cluster, which reduces to computing the distance. Specifically, for the cluster centroids $\{c_1, c_2, \ldots, c_k\}$ where $c_i \in \mathbb{R}^D$ (a $D$-dimensional vector), we can compute the probability of a feature vector $x \in \mathbb{R}^D$ belonging to a cluster $c_i$ as:

$$p(x|c_i) = \frac{1}{\sqrt{2\pi \sigma^2}} \exp\left(-\frac{(x - c_i)^2}{2\sigma^2}\right)$$ (3)

Since we are (naively) assuming all clusters have the same variance (for simplicity, let $2\sigma^2 = 1$) and an equal prior
over all clusters, we can use Bayes rules to rewrite as:

$$
p(c_i|x) = \frac{p(c_i)p(x|c_i)}{\sum_j p(c_j)p(x|c_j)} = \frac{\exp\left(-\frac{(x-c_i)^2}{2\sigma^2}ight)}{\sum_j \exp\left(-\frac{(x-c_j)^2}{2\sigma^2}\right)} \quad (4)$$

which we note is the standard softmax function applied to the squared distances from a feature $x$ to a cluster center $c_i$.

As observed in many large activity recognition datasets, like AVA [14] and Kinetics [20], the activity classes of videos follow a Zipf distribution. We can use this as a prior constraint on the distribution of the videos in these clusters. Specifically, given the above probability of each video belonging to each cluster, and the Zipf distribution, we compute the prior probability of each class as $q(c_i) = \frac{1}{i^{H/s}}$, where $H$ is the $k$th harmonic number and $s$ is some real constant. We then let $p(c_i|x) = \frac{1}{N} \sum_{v \in V} p(c_i|x_v)$, the average over all videos in the set. Using these two probability functions representing the classes/clusters, we can minimize the KL divergence:

$$KL(p||q) = \sum_{i=1}^{k} p(c_i) \log \left( \frac{p(c_i)}{q(c_i)} \right) \quad (5)$$

By using this as a fitness metric, it poses a prior constraint over the distribution of (learned) video representations in clusters to follow the Zipf distribution. Note that this method requires no labeled data and is fully unsupervised. We refer to this entirely unsupervised method as ‘ELo’.

**Weakly-supervised baseline:** As an upper-bound and an alternative to our approach, we use a handful of class labels to evaluate the clustering (which is referred to as ELo-weak). This is done for the sake of comparison and is also a good alternative to align the final loss to a downstream video classification task. A subset of HMDB is used, and $k$-means clustering is applied to the output representation of the RGB stream. The clusters are used for nearest-neighbors classification and the accuracy is the fitness of the individual. Due to randomness in $k$-means clustering, in both settings, we run this processes 20 times and average fitness across all trials.

3.2.3 Loss evolution

As is typical with evolution approaches, the evolution of the loss is driven by mutations. Since our search space consists of continuous values in $[0, 1]$, we compare two different evolutionary strategies: tournament selection [13] and CMA-ES [15]. For the tournament selection search, we mutate an individual loss function by randomly picking one weight and assigning new value in uniformly sampled from $[0, 1]$. For CMA-ES, at each iteration, all the components are changed based on the fitness of all the individuals in the evolution pool. For tournament selection, we evolve the loss function for 2000 rounds, generating and evaluating 2000 different loss functions and we use 250 rounds with CMA-ES, finding much faster convergence. Fig. 3 shows an example how our weights evolve over the rounds and Table 4 compares the performance of different search methods. Since everything is differentiable, we could learn these weights also with gradient descent, however, we leave this for future exploration as taking the derivative of the entire network w.r.t. to task weights is non-trivial.

3.3. Self-supervised tasks

Many tasks have been designed for unsupervised learning. We describe briefly the tasks we employ for our representation learning. Importantly, we allow many possible tasks and let the evolved loss function automatically discover which tasks are important and the optimal relative weightings. We also use tasks like DeepCluster [5] and local aggregation [49].

**Reconstruction and prediction tasks:** Given the representation for a modality, $x_m$, we use a decoder CNN to generate the output. As the reconstruction is only used as a supervision signal, we do not need to generate extremely high quality reconstructions. Thus, we use a small, cheap decoder with only 6 convolutional layers, saving both memory and training time. Once the unsupervised learning is complete, we discard the decoders. Further, following previous work [38], our decoders have no temporal convolution, forcing the representations to contain all needed temporal information, which is desirable for video understanding. Each modality (e.g., RGB, optical flow, and audio) will be reconstructed. We also use several cross-modality transfer tasks: RGB to Flow, Flow to RGB, etc.
Another way to learn video temporal structure is to train a decoder to predict the next $N$ frames given $T$ frames. Future prediction of frames has previously been used for representation learning [37] and we perform this task for each modality. For these tasks, we minimize the $L_2$ distance between the ground truth ($I$) and predicted output ($\hat{I}$):

$$L_R(\hat{I}, I) = ||\hat{I} - I||_2$$  \hspace{1cm} (6)

**Temporal ordering:** We use two tasks to learn representations that take advantage of temporal structure: binary classification of ordered frames and shuffled frames [26] and binary classification of forward and backward videos [31]. We use a single, fully-connected layer to make a binary classification of the representation: $p = Wx_m$ ($x_m$ is the representation for a modality). These are trained to minimize binary cross-entropy:

$$L_B(p, y) = -(y \log(p) + (1-y) \log(1-p))$$  \hspace{1cm} (7)

where $p$ is the output of the binary classifier and $y$ is the ground truth.

**Multi-modal contrastive loss:** As videos contain multiple modalities, we want to take advantage of these different representations to learn an generic representation by using a multi-modal embedding space. Given the representations for each modality, $x_m$, we minimize a contrastive loss between various embedding spaces:

$$L_c(x_1, x_2, x_n) = ||x_1 - x_2||_2 + \max(0, \alpha - ||x_1 - x_n||_2)$$  \hspace{1cm} (8)

where $x_m1$ and $x_m2$ are representations from the same video but different modalities and $x_n$ is a representation from a different video. This task encourages the representations from the same video, but different modalities, to be close in the representation space, while representations from different videos are further apart.

**Multi-modal alignment:** We can further take advantage of both the temporal information and the multi-modal data by performing a multi-modal alignment task, illustrated in Fig. 4. The networks take input of temporally aligned samples from two modalities, and a sample from one modality from a temporally different region. The model is trained to make a binary prediction if the two samples are temporally aligned.

### 4. Experiments

**Unsupervised data source.** We use two million random, unlabeled YouTube video clips sampled randomly from the Youtube-8M dataset [1] (limiting the size for computational reasons). Previous works on self-supervised learning used videos from datasets (e.g., Kinetics or AudioSet in [22]) while ignoring the labels, leading a bias in the dataset, as those videos are trimmed to intervals with specific activities. Using a random sample from Youtube is less prone to bias as the videos are user generated, the labels are automatically tagged (no human verification), and potentially offers a very large set for training (up to 8M). We have verified there is no overlap between those datasets and the ones the models are evaluated on (e.g., HMDB).

**Evaluation datasets and protocols.** We used the following widely used datasets for evaluation: HMDB [23], UCF101 [36], Kinetics [20]. We also used Imagenet [9] and Kinetics for reporting results with pre-training, as is customary in previous work. We use the standard protocols when evaluating video classification results on the labeled datasets, adopted by prior work as well. Please see the sup. material for dataset details.

**Implementation details.** We use a (2+1)D ResNet-50 as our backbone network. Given a loss function, we train the network for 100 epochs on 2 million unlabeled videos. The learning rate is set to 0.1 (during both the evolution and the final training).

During search, we used smaller networks, similar to a ResNet-18 for faster learning. For search, the fitness of each model can be found in 4 hours using 8 GPUs. The final model uses 64 GPUs for 3 days to train (equivalent time to training I3D/(2+1)D ResNet-50 on Kinetics).

#### 4.1. Comparison to previous methods

We evaluate our method in comparison to prior unsupervised and supervised representation learning. Specifically,
Table 2: Comparison to the state-of-art on HMDB51 and UCF101. Note that previous approaches train on activity recognition datasets (e.g., Kinetics) are much more aligned to the final task, whereas we use random video clips. Even using more difficult data, we outperform the previous methods. (The top portion shows results for (2+1)D ResNet-50 with supervised pretraining as in Table 1).

![Figure 5: How much labeled, supervised data is needed.](image)

Figure 5: How much labeled, supervised data is needed once the unsupervised representation is learned. We achieve comparable performance with roughly half the data and outperform the supervised baselines using the entire dataset.

![Figure 6: Comparisons of different amounts of unsupervised data.](image)

Figure 6: Comparisons of different amounts of unsupervised data. **Left:** Total number of training iterations fixed (i.e., less epochs as data is added). **Right:** Total number of epochs fixed (i.e., more iterations as more data is added). We observe that adding more data without increasing training time improves performance, while training longer on more data is better. On HMDB.

![Figure 7: The values of the loss function for the various tasks throughout evolution.](image)

Figure 7: The values of the loss function for the various tasks throughout evolution. Higher weight values indicate the task is more important. The learned loss functions automatically finds the tasks that most benefit recognition.

We evaluate the representations in 3 settings: (1) k-means clustering of the representations (2) fixing the weights of the network and training a single, fully-connected layer for classification and (3) fine-tuning the entire network. These three evaluations are done by directly evaluating the representation as well as fine-tuning the entire network.

We find that while all approaches outperform the randomly initialized networks, only our evolved loss function outperforms ImageNet pretraining and performs comparably to the pretrained network with labeled Kinetics data (Table 1). Furthermore, we outperform all prior unsupervised methods. Our approach performs similarly to the weakly supervised version of our evolution method, despite being unsupervised. We also compare to a loss function randomly sampled from our search space, which performs poorly. We find that some tasks are not beneficial to representation learning, thus the evolution is quite important as this allows automatically finding the best tasks and weightings.

In Table 2, we compare our approach to previous reported methods. We find that even though our approach is using more difficult unlabeled data, we still outperform the exiting methods by a significant margin. We also find that distillation is extremely important. Without it, the RGB network can only take advantage of the other modalities through multi-modal tasks.

### 4.2. Improving supervised learning

Once we have learned a representation space using large amounts of unlabeled data, we want to determine how much labeled data is needed to achieve competitive performance. In Fig. 5 and Table 3, we compare various approaches trained using our unlabeled videos then fine-tuned on Ki-
Table 3: Using different amounts of labeled samples on the currently available (March 2019) Kinetics-400 dataset using a (2+1)D ResNet-50. We achieve similar performance with only \( \sim 50\% \) of the data. Using the entire dataset, we outperform the randomly initialized network.

<table>
<thead>
<tr>
<th>Method</th>
<th>400</th>
<th>2k</th>
<th>4k</th>
<th>8k</th>
<th>20k</th>
<th>40k</th>
<th>80k</th>
<th>120k</th>
<th>160k</th>
<th>225k (all samples)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Random Init</td>
<td>0.93</td>
<td>2.1</td>
<td>2.8</td>
<td>4.4</td>
<td>6.2</td>
<td>12.5</td>
<td>26.4</td>
<td>52.5</td>
<td>64.3</td>
<td>71.2</td>
</tr>
<tr>
<td>Frame Shuffle</td>
<td>1.5</td>
<td>5.3</td>
<td>12.4</td>
<td>18.4</td>
<td>28.4</td>
<td>32.5</td>
<td>38.2</td>
<td>57.4</td>
<td>66.8</td>
<td>70.9</td>
</tr>
<tr>
<td>Audio Align</td>
<td>2.5</td>
<td>9.8</td>
<td>17.2</td>
<td>28.1</td>
<td>36.0</td>
<td>46.0</td>
<td>54.1</td>
<td>64.3</td>
<td>69.5</td>
<td>71.5</td>
</tr>
<tr>
<td>ELo (unsupervised)</td>
<td>3.6</td>
<td>15.8</td>
<td>24.8</td>
<td>47.0</td>
<td>58.3</td>
<td>67.5</td>
<td>69.2</td>
<td>70.2</td>
<td>72.2</td>
<td>74.4</td>
</tr>
</tbody>
</table>

Table 4: Comparison of best loss found with different evolutionary strategies evaluated on HMDB.

<table>
<thead>
<tr>
<th>Method</th>
<th>Num iter.</th>
<th>Acc</th>
</tr>
</thead>
<tbody>
<tr>
<td>Random Search</td>
<td>2000</td>
<td>52.4</td>
</tr>
<tr>
<td>Grid Search</td>
<td>2000</td>
<td>57.3</td>
</tr>
<tr>
<td>Tournament Selection</td>
<td>2000</td>
<td>61.4</td>
</tr>
<tr>
<td>CMA-ES</td>
<td>250</td>
<td>67.4</td>
</tr>
</tbody>
</table>

4.3. Benefit of additional unlabeled data

We explore the effect of using different amounts of unlabeled data. Given a loss function, we train a network using \( N \) unlabeled samples. We compare the training both keeping the iterations fixed and the number of epochs fixed.

The results on HMDB are shown in Fig. 6. When fixing the number of iterations to 100k, the performance increases as we add more data, even though the number of epochs (e.g., number of times each sample is seen) decreases. This suggests that during unsupervised training, the use of more, diverse data is beneficial, even when samples are seen fewer times. When fixing the number of epochs to 100, we find that adding more data further improves performance, suggesting that more training plus more data is best.

4.4. Additional Analysis

Examining the weights of the evolved loss function, \( \lambda_{m,t} \) and \( \lambda_d \), allows us to check which tasks are more important for the target task. Fig. 7 illustrates the weights for several tasks (\( \lambda_{m,t} \)) over the 250 evolution rounds. We observe tasks such as RGB frame shuffle get very low weights, suggesting they are not very useful for the action recognition task. Tasks such as audio alignment are quite important. The final fully-evolved loss is shown in Fig. 8.

Table 4 compares different search methods. As seen CMA-ES converges the most quickly and to the best fitness. In Fig. 9, we compare the two different fitness measures, finding strong correlation. This suggests that Zipf matching is suitable for unsupervised representation evaluation.

5. Conclusion

We proposed a unified framework for multi-task, multi-modal unsupervised video representation learning and found it benefits recognition tasks. We further introduced the concept of loss function evolution to automatically find the weights of the self-supervised tasks and modalities, with unsupervised fitness measure. We find powerful unsupervised video representations that outperform prior self-supervised tasks and can match or improve the performance of networks trained on supervised data.
References


ings of European Conference on Computer Vision (ECCV), pages 69–84, 2016. 3
[30] Dilip Krishnan Phillip Isola, Daniel Zoran and Edward H Adelson. Learning visual groups from co-occurrences in space and time. 2015. 3