Learn2Augment: Learning to Composite Videos for Data Augmentation in Action Recognition

Shreyank N Gowda[1], Marcus Rohrbach[2],
Frank Keller[1], and Laura Sevilla-Lara[1]

[1] University of Edinburgh
[2] Meta AI

Abstract. We address the problem of data augmentation for video action recognition. Standard augmentation strategies in video are hand-designed and sample the space of possible augmented data points either at random, without knowing which augmented points will be better, or through heuristics. We propose to learn what makes a "good" video for action recognition and select only high-quality samples for augmentation. In particular, we choose video compositing of a foreground and a background video as the data augmentation process, which results in diverse and realistic new samples. We learn which pairs of videos to augment *without* having to actually composite them. This reduces the space of possible augmentations, which has two advantages: it saves computational cost and increases the accuracy of the final trained classifier, as the augmented pairs are of higher quality than average. We present experimental results on the entire spectrum of training settings: few-shot, semi-supervised and fully supervised. We observe consistent improvements across all of them over prior work and baselines on Kinetics, UCF101, HMDB51, and achieve a new state-of-the-art on settings with limited data. We see improvements of up to 8.6% in the semi-supervised setting. Project Page: `https://sites.google.com/view/learn2augment/home`

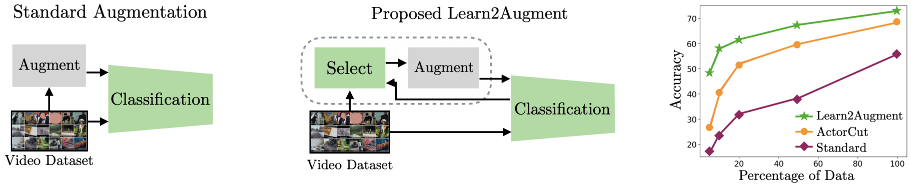

Fig. 1. Standard video augmentation techniques generate data using hand-designed heuristics (left). We propose to learn to select videos for augmentation, based on how effective they will be for learning to classify (middle). Our approach, Learn2Augment, improves classification across datasets and settings, including UCF101 (right).

1 Introduction

Large-scale datasets have played a key role in the progress of research across AI problems. In computer vision, neural networks have existed for decades, but one of the enabling factors for the current revolution was the development of the large ImageNet [8]. In the video domain, manually collecting and annotating data can be a prohibitively expensive process. In video action recognition, for example, collecting data requires an immense amount of manual labor, as it involves finding suitable videos, trimming them and classifying them.

Recent efforts in video focus on relieving the strong dependency of current methods to the size of labeled datasets. Some of these efforts [33,43] involve increasing the number of data samples through data augmentation. This strategy aims to create new videos in the training set by performing transformations on the original annotated videos, where labels are known. This process adds diversity to the training data, while new videos are still realistic and plausible. In the simplest version of data augmentation in video, new data samples are generated by flipping the input video horizontally, or by cropping a subsection of the video. New methods [43,38] propose more sophisticated processes like combining two videos. VideoMix [38] randomly crops regions of one video and pastes them onto another. ActorCut [43] goes one step further and uses the bounding box detections of humans on one video to paste them onto the background of another video. This increases the diversity of the new videos, and despite the lack of visual realism of the resulting videos, this strategy helps.

However, as datasets become larger, such data augmentation strategies become computationally expensive. The search space of possible video pairs and transformations is enormous and difficult to explore. The solution is often to sample the space randomly, or to manually design augmentation heuristics. Any exploration process is particularly burdening in the context of video data, where the augmentation process needs to be repeated in every frame, which may be orders of magnitude more expensive than for images.

In this paper we address the problem of sampling for data augmentation, and propose to learn to select pairs of videos. We show that this reduces the search space of augmented data points by orders of magnitude and improves the final accuracy of the classifier significantly. We leverage two observations. First, not all data points are as useful for classification. This idea has been exploited in the context of frame or clip selection [11,20,16]. Second, we can learn to predict which data points will be useful without actually generating them. This is essential, as the space of transformations is huge, and if we needed to create each candidate augmented video, the process would be prohibitively expensive.

More concretely, we propose a data augmentation method which we call Learn2Augment. The proposed method contains a “Selector” network, which predicts a score of how useful a combination of two videos will be, without having to actually composite them. The Selector is trained using the accuracy of the classification as the cue. Since this metric depends on the classifier, it is not differentiable with respect to the Selector’s parameters. Therefore we optimize the network using reinforcement learning. Once the Selector network is trained, we use it to choose good pairs of videos, composite them, and train a classification network. In our experiments, for example in the case of the UCF101 dataset, using the Selector reduces the number of augmented videos by 92% while increasing the classification accuracy.

In the proposed method, each augmented video is created from a pair of videos using a composition of the segmented foreground of one video, including actor and objects, onto the background of the other video. This process yields diverse and realistic new data samples, which we demonstrate is important for learning. More concretely, results show an improvement of 4.4% over using a simpler transformation.

The Selector is indeed useful to reduce the number of videos for training the classifier. However, we also need to reduce the space of possible pairs for training the Selector network itself. For example, the number of possible pairs of videos in video datasets can be in the order of millions for small datasets or billions for large datasets. For this, we leverage the natural correlation between the occurrence of foreground activities and background scenes [5]. This is, it is more likely to find someone playing football in a football field than at a restaurant. Instead of sampling at random the pairs of videos to train the Selector on, we sample pairs from classes that are semantically similar. In particular, we use the class names to obtain a semantic embedding, and match each class to their nearest neighbor in this space. Experiments show that this extremely simple design choice of Semantic Matching reduces the space of possible pairs of videos by several orders of magnitude (from quadratic to linear on the number of videos). This yields better results than choosing pairs at random, which may result in non-plausible scenarios, or choosing pairs from the same class, which may not add as much diversity.

In summary, the proposed **Learn2Augment** contains three core components: a Selector that learns to choose good videos to augment, a Semantic Matching method that improves optimization, and a Video Compositing that composites video pairs for augmentation. Experimental results show that all components contribute to the performance of the system in different ways, and the overall method obtains state-of-the-art in all datasets, and in all settings that involve limited training data. In addition, in the setting which considers the full training set, the proposed data augmentation technique improves upon the baseline on all datasets, including UCF101, HMDB51, and the large-scale Kinetics-400.

2 Related Work

Data Augmentation for Video Action Recognition. Standard data augmentation techniques in action recognition include horizontal flip and cropping, where new videos are created by selecting a box at each frame, and then resizing the resulting video to have the same size as the original one. While this strategy helps, generated videos do not add much diversity to the training set. Recent efforts such as ActorCut [43] and VideoMix [38] increase the diversity of new video

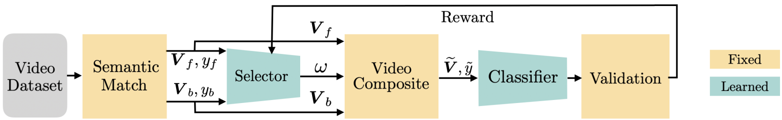

Fig. 2. Overview of the proposed Learn2Augment. Given a pair of videos and their labels, a Selector network gives a score ω of the quality of the potential composited video. At training time, the Selector is trained with the validation loss of the classification network. Once the Selector is trained, pairs of videos are sampled, and only the promising combinations with high score ω are composited and used for training the classifier.

samples by cutting and pasting the foreground of one video onto another. This general technique of combining two data samples has proven to be quite effective, even in the image domain [37]. However, the resulting videos are not very realistic, and are used for training regardless of their quality. Zhang et al. [42] go one step further and synthesize new samples using GANs, and use "self-paced selection" to train, starting with easy samples and progressively choosing harder samples. Instead, we propose to create realistic data samples by segmenting, inpainting and blending the foreground of one video onto the background of another. Crucially, we learn to discard novel video samples that are not expected to be useful for classification, overall producing a more accurate data augmentation strategy.

Learning to Augment Data. The idea of learning to augment data has been used in other computer vision problems. In the image classification domain, this strategy has been done using the final classification loss as the training criterion [25], augmenting in feature space [9], and learning data augmentation policies [6]. As in this paper, in the image domain it has been noted that the search space for data samples can be large and thus expensive [7].

Other computer vision domains like low level vision, also struggle with data dependency, as creating ground truth is particularly hard. In optical flow, AutoFlow [33] recently introduced the strategy of learning to generate good training data for a target dataset.

Semi-supervised Video Action Recognition. Semi-supervised learning (SSL) also aims to reduce data dependence by learning from large sets of unlabeled samples and a small set of labeled ones. SSL in images has been widely explored. For example, some strategies include giving pseudo-labels [1,24] to samples where the classifier has high confidence, and adding these to the labeled training data. Other common approaches use consistency regularization [22,23,34]. Approaches that combine consistency regularization and entropy minimization [13] have shown to be very effective in tackling the SSL task in images such as MixMatch [3] and RemixMatch [2].

SSL in videos however, has not been explored as much. One of the early works used extreme learning machines [17] to perform SSL on videos. Recently,

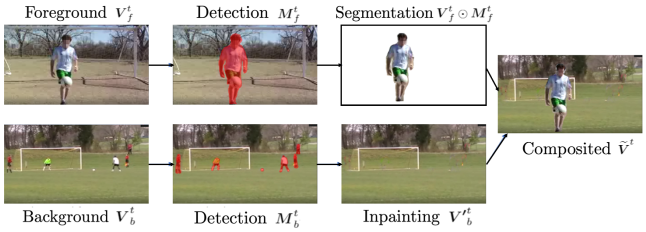

Fig. 3. Pipeline for compositing a single frame. The foreground is from the class "soccer juggling" and the background from the class "soccer penalty", which are semantic class neighbors. We can see objects such as 'person' and 'ball' are detected as objects of interest.

VideoSSL [18] and Temporal Contrastive Learning (TCL) [30] leverage SSL in videos. VideoSSL [18] uses pseudo-labels and object cues from unlabeled samples to guide the learning process. TCL [30] use a two-pathway contrastive learning model using unlabeled videos at two different speeds with the intuition that changing video speeds do not change the action being performed.

Data augmentation and SSL are two different families of techniques to relieve the dependence on labeled data, and in this paper we experiment with the combination of both, showing that they are actually complementary.

Sample Selection. Recent work [16] has shown that not all data samples are as useful. Selecting a subset of high quality frames or clips at test time shows better results than using the entire video for action recognition. In this spirit, SMART [11] uses an attention and relation network to learn scores for each frame in a video and then select only the high ranked ones for inference. Similarly, SCSampler [20] uses a lightweight clip sampling model to select the salient clips in a video and use only those. Unlike the proposed method, these learn to choose single videos, which are already available, while we learn to choose pairs of videos to be composited, which are not already combined.

The most relevant work to ours is data valuation in the image domain, using RL [36], in the image domain where each sample is given a score of how effective the sample is, and at training time the sample is multiplied by this score. In our work, instead of learning the effectiveness of the training set, we leverage that knowledge for augmentation.

3 Learn2Augment

In this section we describe in detail the architecture of the proposed Learn2Augment. In a nutshell, the goal is to learn to augment novel data points which are realistic and diverse, such that we can train a better classifier with them. For this, we train a Selector network, which predicts a score of how useful a given pair of videos is for augmentation. We pick pairs that have a high score to be augmented. The transformation we use for augmentation is Video Compositing.

Training the Selector using the entire dataset is infeasible, and sampling pairs of videos at random will yield unlikely pairs. Thus we sample pairs of videos using Semantic Matching. Figure 2 shows an overview of the proposed method and in Sec. 4 we describe how we train our approach.

3.1 Selector

Given two input videos $\boldsymbol{V}_1$ and $\boldsymbol{V}_2$, the goal of the Selector is to predict a weight ω, rating the quality of the potential composited video. Note that the input to the Selector is two putative videos instead of the composited one. This means that at test time, we can predict how useful the composited video will be without having to actually create it.

The architecture of the Selector includes a standard video classification network to extract video features, which is ResNet3D-18 [15] followed by a simple multi-layer perceptron (MLP) with 3 hidden layers of sizes 2048, 1024 and 512. Two videos are input to the Selector at a time, and their features and labels are concatenated and input to the MLP.

Since there is no ground truth of how "good" a video sample is for learning, we train the Selector using the change in validation loss of the classifier. This is, we argue that a "good" training sample is one which, if used for training, improves the validation loss of the classification network. In other words, if we take one optimization step training the classifier, after updating the weights, the validation loss will go down. Section 4.1 describes the training process in detail.

At test time, we use the Selector by sampling pairs of videos, choosing those pairs with high score ω, and input to the Video Compositing module, which we describe in Sec. 3.3. The resulting video is finally used to augment the training set for the classification network.

3.2 Semantic Matching (SM)

The number of pairs in the full dataset can be very large, as it grows with the square of the number of videos. For Kinetics [4], for example, we would encounter 360 billion pairs. Training the classifier using these is clearly infeasible, and thus we use the Selector. But training the Selector itself with all these samples is infeasible too. Sampling uniformly is a reasonable solution, but many video pairs may not be useful for learning. We leverage the observation that all combinations of actions and backgrounds are not equally likely [5]. This natural correlation between actions and backgrounds helps to prune unlikely class combinations.

For this, we make the assumption that classes that are semantically similar are more likely to contain a foreground and a background that are plausible in the real world, and therefore more realistic for our data augmentation purposes. Thus, we use the class names to extract a language embedding using sen2vec [28], and use these embeddings to match each class to its nearest neighbor. We sample videos $\boldsymbol{V}_1$ and $\boldsymbol{V}_2$ from class c_1 and its closest neighbor c_2 respectively. This simple decision reduces the number of pairs to grow linearly with the size of the dataset, and furthermore increases the accuracy significantly with respect

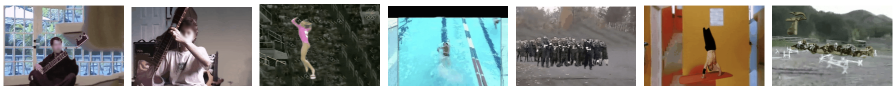

Fig. 4. Sample frames of rendered videos. While the segmentation contains errors, such as missing limbs or portions of the object, the action category remains clear.

to sampling video pairs at random. More details on the numerical impact can be found in Sec. 5.3. Semantic class pairs and additional experiments using intra-class augmentation can be found in the supplementary material.

3.3 Video Compositing (VC)

The goal of the augmentation process is to composite two videos, to produce realistic, plausible and diverse new videos, that will improve the classification. Figure 3 shows the overall pipeline for compositing a single frame.

Given two videos which will be used for foreground $\boldsymbol{V}_f$ and background $\boldsymbol{V}_b$, we use a standard object segmentation network (MaskRCNN [14]) to segment out people and objects in every frame of both videos. Objects categories in action datasets are not completely contained in the image dataset COCO [26], which is used for training MaskRCNN. However, we observe that object detections with high confidence tend to correspond to actual objects, even if the category is not correct (boxing bag is often classified as fire hydrant), and therefore are useful to our purpose. We could also have selected only the humans in the video, as action categories tend to be focused on humans. However, we find that the presence of specific objects is highly correlated with action categories (musical instruments in the classes "playing guitar" or "playing violin"). Therefore removing the original objects from the background and adding the ones from the foreground is essential for recognition. See numerical results of the impact of these decisions in the ablation study of Sec. 5.3.

We remove the segmented objects from the background video and fill in the holes using image inpainting [27], to obtain a clean background video $\boldsymbol{V'}_b$. Finally, we combine the foreground objects and the background at each frame by simple composition, as in:

$$\widetilde{\boldsymbol{V}}^t = \boldsymbol{V}_f^t \odot \boldsymbol{M}_f^t + \boldsymbol{V'}_b^t \odot (1 - \boldsymbol{M}_f^t), \tag{1}$$

where $\widetilde{\boldsymbol{V}}^t$ is the resulting composited frame at time t, $\boldsymbol{V}_f^t$ and $\boldsymbol{V'}_b^t$ are frames of the foreground and background videos respectively, $\boldsymbol{M}_f^t$ is the binary mask with the union of all detected objects, and $\odot$ is the element-wise multiplication. Figure 4 shows sample frames of the resulting videos.

4 Optimization of Learn2Augment

The optimization of the proposed Learn2Augment method has two stages. In the first stage, we train the Selector network using RL, as described in Sec. 4.1.

Once the Selector network is trained, in the second stage, we perform data augmentation to train the classifier. That is, we sample pairs of videos, pass them through the trained Selector, choose the pairs with high score, create new videos with these pairs through Video Compositing, and add them to the training set. We now describe the details of these two training stages.

4.1 Training the Selector

As mentioned before, there is no ground truth to tell us how good an augmented data sample is. Instead, we use the validation loss of the classification network to train the Selector network. This function is not differentiable with respect to the parameters of the Selector. A common solution to dealing with this is to use RL [36].

Specifically, the state s_t at time t is the batch of video pairs sampled using SM. The action a_t is the subset of these video pairs selected for compositing and is represented as a vector of values between 0 and 1. The environment is the classification network and the validation process. This environment is used to compute a reward $R(\theta)$ for choosing a particular action, where θ are the parameters of the Selector.

We calculate the reward in a single step, as the difference between the loss in the current batch and the moving average of losses in the previous S steps (where $S = 5$) denoted as δ, as in Eq. 2:

$$R(\phi) = \left(\frac{1}{|D_{\text{val}}|} \sum_{i=1}^{|D_{\text{val}}|} \mathcal{L}_{\text{cls}}(f_\phi(\boldsymbol{V}_i), y_i) \right) - \delta \tag{2}$$

where $\mathcal{L}_{\text{cls}}$ is the classification cross-entropy loss, f_ϕ is the classifier network of parameters ϕ, $\boldsymbol{V}_i$ and y_i are an input video and its label respectively, D_{val} is the validation set and $|D_{\text{val}}|$ is the number of samples in D_{val}. The objective function that we want to maximize is the expected value of the reward:

$$J(\theta) = \mathbb{E}(R(\phi)). \tag{3}$$

To find the optimal policy, we would typically differentiate the objective function with respect to the parameters θ. However, the reward function is dependent on the validation loss, calculated with the classifier network, which does not involve θ. Instead, using REINFORCE [35], we approximate the objective function as:

$$\nabla_\theta J(\theta) \approx \frac{1}{M} \sum_{i=1}^{M} R_{\tau^i}(\phi) \left(\sum_{t=0}^{T-1} \nabla_\theta \log \pi_\theta(a_i^t | s_i^t) \right), \tag{4}$$

where, τ^i is the i^{th} state-action trajectory under the policy π_θ, M is the number of sample trajectories and T is the number of actions performed in a trajectory. Note that as we have single-step episodes, we can make several simplifications as $M = 1$, as $T = 1$, and as there is only one trajectory τ^i, and thus $R_{\tau^i}(\phi)$ is just $R(\phi)$. With these simplifications and substituting Eq. 3 in Eq. 4, we obtain:

$$\nabla_\theta J(\theta) \approx R(\phi) \nabla_\theta \log \pi_\theta(D_M | D_B), \tag{5}$$

where D_M corresponds to the subset of pairs of samples to composite and D_B to all the pairs of samples in the batch. The Selector is updated by $\alpha \nabla_\theta J(\theta)$ where α is the learning rate and δ is updated with the last calculated loss as seen in Eq 6.

$$\delta_t = \frac{S-1}{S} \delta_{t-1} + \frac{1}{|D_{\text{val}}|} \sum_{i=1}^{|D_{\text{val}}|} \left(\mathcal{L}_{cls}(f_\theta(\boldsymbol{V}_i), y_i) \right). \tag{6}$$

Note that this training process does involve generating the composited videos for pairs in D_M, to input to the classifier and compute the loss. However, crucially, during training this is a small portion (one order of magnitude smaller) of how many videos would need to be generated if we were to composite all pairs of videos.

Once the Selector is trained, we use it for actually filtering good pairs. At that point, given two videos and their labels, the Selector network predicts a policy π of how likely it is to select the pair. The score ω is the value of π for each pair. We use a threshold on that score to select the pairs of videos to augment. In our experiments, we first determine a budget on the number of videos that we want to augment, and then pick the threshold to select the top-ranked video pairs. We use these selected pairs of videos as input to Video Compositing, add them to the training set, and use them to train the classifier.

4.2 Training the Classifier

Similar to previous work which combines multiple samples for augmentation [37,43], composited/mixed samples should include mixed labels. We adopt the strategy of Cutmix [37], where the foreground label y_f and the background label y_b are combined using a ratio λ, as:

$$\tilde{y} = \lambda y_f + (1-\lambda) y_b, \tag{7}$$

to obtain the mixed label $\tilde{y}$. A simple way to choose λ is to use the ratio of the foreground mask with respect to the overall video. Given the foreground video $\boldsymbol{V}_f$ of dimensions $T \times H \times W$, and mask at each frame M_f, the foreground ratio would be $\gamma = \sum \boldsymbol{M}_f/(THW)$. Instead of choosing λ to be directly proportional to the foreground ratio γ, we give slightly more weight to the foreground [43], as in Eq. 8, where $\alpha = 4$.

$$\lambda = -(\gamma - 1)^\alpha + 1, \gamma \in [0, 1] \tag{8}$$

We add composited videos $\widetilde{\boldsymbol{V}}$, and their mixed labels $\tilde{y}$ to the training set, and train the classifier network using a standard cross-entropy loss, with stochastic gradient descent.

The choice of classifier is not tied to our method. In our experiments, we choose the widely used 3D ResNet-18 architecture, which allows us to compare directly to other approaches.

5 Experiments

We experiment extensively with Learn2Augment using three data settings, four datasets, and two splits. We also present ablation studies. In this section we first describe the details of the experiments and then discuss our results.

5.1 Experimental Details

Datasets. In order to provide comparison to prior work (e.g. [43,30]), we use standard datasets for evaluation in action recognition, including HMDB51 [21], UCF101 [32], Kinetics-400 [4], and Kinetics-100, which includes the 100 classes with the largest amount of samples in Kinetics, as it is used in prior work [18] and helps us compare directly. For experiments on the effect of pre-training the Selector, we use Kinetics-400. For the semi-supervised setting, we split the datasets following the protocol of VideoSSL [18] and ActorCut [43]. For few-shot we use the standard split [40] and the Truze split [12] which ensures no overlap of novel classes with Kinetics-400.

Problem Settings. We test the proposed method in three different settings. In the *semi-supervised* setting, a portion of the training set is artificially held out, and the rest of the training data is assumed to be available, but unlabeled. Tests are performed on different percentages of held out data. In the *few-shot* setting, some classes (novel classes) are assumed to have a very small number of training samples (one to five instances), while other classes have the full number of samples (seen classes). We effectively change the n-shot learning problem to a $n + k$-shot problem where k is the number of augmented samples. Finally, in the standard *full set* setting, all training data is available.

Training Settings. We use mini-batch stochastic gradient descent, with momentum of 0.9 and weight decay 0.001. For each video, we use an 8-frame clip, where the frames are uniformly sampled. We use batch size of 8. For UCF101 and Kinetics100 in the SSL setting, we train the model for 400 epochs and for HMDB51, we train for 500 epochs. The initial learning rate is set to 0.1 and then decayed using cosine annealing policy. For the SSL setting, we use the data split proposed in VideoSSL [18]. For the few-shot setting, we use the default hyperparameters of TRX [29], ARN [40] and C3D-PN [31], respectively. In the fully supervised setting, we train R(2+1)D for 100 epochs on UCF101, HMDB51 and 50 epochs on Kinetics-400.

5.2 Architectural Changes for Different Settings

We briefly explain the structural adaptations of our approach for each of the settings.

Pairs Selector	Video Compositing	Semantic Matching	Accuracy in %	#Videos (S)
✓	✓	✓	**58.9**	12K
×	✓	✓	55.8	99K
✓	×	✓	54.5	12K
✓	✓	×	55.2	(1.2M)
✓	×	×	52.9	(1.2M)
×	✓	×	48.6	(10.4M)
×	×	✓	50.8	99K
×	×	×	45.5	(10.4M)

Table 1. Ablation study to explore the impact of each proposed component. All settings use the same number of samples for training, so that they can be compared fairly. The # Videos (S) corresponds to the search space in each scenario. As we can see, we obtain the best accuracy using just 12K instead of the standard scenario which would have had 10.4M i.e. a reduction of over 1000x.

Semi-supervised Learning. Similar to VideoSSL [18], we first train the classifier on the available labeled data using the categorical cross-entropy loss. Once this network is trained, we do a forward pass of the unlabeled examples and assign pseudo-labels to those samples with high confidence. We use these pseudo-labels as additional data for augmentation. We also add a knowledge distillation loss inspired by VideoSSL [18]. Details can be found in the supplementary material.

Few-shot Learning. We only augment the novel classes using Learn2Augment. We also do not perform label mixing and simply use the foreground label for the augmented sample. This incorporates our composited samples seamlessly into the meta-learning framework typically followed. We show results on the standard split, as on the recently proposed TruZe [12]. TruZe ensures that the novel classes do not overlap with Kinetics-400.

Fully-supervised Learning. This is the simplest setting, where the Selector is trained on the full training set, and used for data augmentation to train the classifier. We explore two scenarios: training the classifier from scratch and using a model pre-trained on Sports1M [19].

5.3 Ablation Study

Table 1 shows the ablation study of Learn2Augment, which illustrates the impact of each of the proposed elements in the design. The experiment is done on the UCF101 dataset, using 20% of the data i.e. in a semi-supervised setting. All three contributions (Selector, Semantic Matching and Video Compositing) improve accuracy. Crucially, Semantic Matching and the Selector also reduce greatly the number of possible video combinations, and the overall reduction is around three orders of magnitude. We see that Learn2Augment obtains a 13.4% improvement over the baseline. While there are improvements of up to 7.4%

Method	Accuracy
L2A	58.9
L2A w/o Inpaint	57.6
L2A w/o Segmentation	56.8
L2A w/o Objects	55.7
L2A w/o All	54.5

Table 2. Ablation study of compositing components. The version "w/o Inpaint" refers to pasting the foreground without first filling in the holes of removed objects in the background. The version "w/o Segmentation" refers to using bounding boxes instead of object segmentations. "w/o Objects" refers to copying and pasting only the humans in the scene, leaving the objects.

for each component, the combination of all three gives the best results. Further analysis can be found in the supplementary material.

The Video Compositing module also has multiple components. In Table 2, we ablate these components and observe that removing objects is actually essential, and has the most significant impact, followed by using segmentation instead of a bounding box, and finally inpainting.

Although the compositing process is more computationally expensive than previous simpler mixing strategies, it is important to note that 1) the overall accuracy indeed improves, 2) the actual composition for training the classifier is done on a small subset of pairs of videos and 3) the Selector can be trained on a large dataset (e.g.: Kinetics) just once and can be reused for the smaller datasets without the need of fine-tuning (see Table 3).

		Kinetics 100				UCF101				HMDB51		
Method	Conference	50%	20%	10%	5%	50%	20%	10%	5%	60%	50%	40%
CutMix [37]	ICCV19	53.7	46.1	43.2	39.9	46.1	36.5	34.6	25.8	33.9	30.8	27.8
MixUp [41]	ICLR18	53.4	45.5	43.0	39.6	45.8	36.1	34.2	25.5	33.7	31.0	27.5
CutOut [10]	Arxiv17	52.8	45.1	42.3	38.8	45.2	35.6	33.9	24.6	33.0	30.5	27.1
ST-VideoMix [38]	Arxiv21	55.3	46.6	43.9	40.4	46.4	36.4	35.2	25.9	34.8	31.3	28.7
PseudoLabel [24]	ICMLW13	59.0	48.0	38.9	27.9	47.5	37.0	24.7	17.6	33.5	32.4	27.3
MeanTeacher [34]	Neurips17	59.3	47.1	36.4	27.8	45.8	36.3	25.6	17.5	32.2	30.4	27.2
S4L [39]	ICCV19	54.6	51.1	43.3	33.0	47.9	37.7	29.1	22.7	35.6	31.0	29.8
VideoSSL [18]	WACV21	65.0	57.7	52.6	47.6	54.3	48.7	42.0	32.4	37.0	36.2	32.7
ActorCut [43]	Arxiv21	68.7	61.2	56.8	52.7	59.9	51.7	40.2	27.0	38.9	38.2	32.9
ActorCut+ID [43]	Arxiv21	72.2	68.7	63.9	59.1	64.7	57.4	53.0	45.1	40.8	39.5	35.7
TCL [30]	ICCV21	70.4	64.7	61.1	58.2	62.1	55.4	52.1	42.8	41.2	40.4	34.8
L2A		**75.9**	**72.1**	**67.5**	**63.7**	72.1	60.3	56.1	48.0	44.5	43.2	37.9
L2A +Pre-training		-	-	-	-	**73.3**	**64.8**	**60.1**	**50.9**	**47.1**	**46.3**	**42.1**

Table 3. Results on the semi-supervised setting. Results for TCL and ActorCut are obtained by us running the author's code. All methods are run with a 3D ResNet-18 backbone for fair comparison. L2A +Pre-training refers to pre-training the selector and fixing it.

5.4 Augmenting in the Semi-supervised Setting

In this setting we artificially hold out a portion of the training set, with the goal of observing the behavior of different methods as the size of the training set changes. In this setting, we use the remaining part of the dataset by producing pseudo-labels, similar to VideoSSL [18]. Table 3 shows results in this semi-supervised setting. The L2A version of the method uses a Selector and a classifier trained only on the target dataset (in this case UCF101, HMDB51 or Kinetics-100). We observe that Learn2Augment improves on all settings over all previous methods.

The "L2A +Pre-training" row refers to Learn2Augment where the Selector has been pre-trained on Kinetics-400, without fine-tuning on the target dataset. We make two observations: First that pre-training on a large dataset helps, as the results from the pre-trained model are higher for all datasets and settings. Second that the Selector trained on Kinetics generalizes quite well to the smaller datasets without the need for fine-tuning. We do not test on Kinetics-100 with the pre-trained model, as this would mix training and testing sets.

5.5 Augmenting in the Few-shot Setting

We also explore the impact of the proposed method on the more extreme few-shot setting, where there are only a few examples per class. This is interesting because few-shot methods are already designed to address data scarcity.

We compare with the current state of the art in this setting, including CD3-PN [31], ARN [40] and TRX [29], on the UCF101 and HMDB51 datasets. We observe that the proposed Learn2Augment method improves upon all existing approaches, suggesting data augmentation is complementary to few-shot methods. Table 4 shows the results of the experiments.

		UCF101					HMDB51				
Method	Split	1	2	3	4	5	1	2	3	4	5
C3D-PN [31]	S	57.1	66.4	71.7	75.5	78.2	38.1	47.5	50.3	55.6	57.4
C3D-PN + L2A	S	**60.8**	**68.9**	**73.3**	**76.6**	**79.1**	**39.8**	**48.9**	**51.5**	**57.3**	**58.2**
ARN [40]	S	66.3	73.1	77.9	80.4	83.1	45.5	50.1	54.2	58.7	60.6
ARN + L2A	S	**67.7**	**74.2**	**79.6**	**81.1**	**84.4**	**47.3**	**51.7**	**55.5**	**60.1**	**61.8**
TRX [29]	S	77.5	88.8	92.8	94.7	96.1	50.5	62.7	66.9	73.5	75.6
TRX + L2A	S	**79.2**	**89.2**	**93.2**	**95.0**	**96.3**	**51.9**	**63.8**	**68.2**	**74.4**	**77.0**
C3D-PN [31]	T	50.9	61.9	67.5	72.9	75.4	28.8	38.5	43.4	46.7	49.1
C3D-PN + L2A	T	**52.5**	**63.8**	**70.1**	**75.2**	**78.2**	**29.9**	**40.1**	**44.5**	**47.7**	50.8
ARN [40]	T	61.2	70.7	75.2	78.8	80.2	31.9	42.3	46.5	49.8	53.2
ARN + L2A	T	**63.9**	**73.1**	**77.4**	**80.4**	**81.3**	**33.6**	**43.7**	**48.0**	**51.1**	**53.8**
TRX [29]	T	75.2	88.1	91.5	93.1	93.5	33.5	46.7	49.8	57.9	61.5
TRX + L2A	T	**76.8**	**88.9**	**92.7**	**93.8**	**94.1**	**35.0**	**48.1**	**51.1**	**59.2**	**62.1**

Table 4. Results on UCF101 for the Few-Shot Learning setting, with different splits. Accuracies are reported for 5-way, 1, 2, 3, 4, 5-shot classification. S corresponds to the split used in [40,29] and T is the TruZe split [12], which avoids overlapping classes with Kinetics.

Augmentation	Dataset	Pretrained	Top-1
Standard	UCF101	No Pretraining	55.7
ActorCut [43]	UCF101	No Pretraining	68.3
L2A	UCF101	No Pretraining	**73.1**
Standard	HMDB51	No Pretraining	40.8
ActorCut [43]	HMDB51	No Pretraining	44.5
L2A	HMDB51	No Pretraining	**46.4**
Standard	UCF101	Sports1M	93.6
L2A	UCF101	Sports1M	**95.3**
Standard	HMDB51	Sports1M	66.6
L2A	HMDB51	Sports1M	**68.4**
Standard	Kinetics	Sports1M	75.4
L2A	Kinetics	Sports1M	**76.3**

Table 5. Augmenting standard datasets improves classification even with a model pre-trained on the largest existing dataset (Sports1M).

5.6 Augmenting the Full Training Set

We finally explore the effect of augmenting the full dataset, both for smaller datasets, and the large-scale Kinetics. Results can be found on Table 5. Again, Learn2Augment improves the performance on all datasets even for a pre-trained model.

6 Limitations and Future Work

The main area of improvement is the time needed for training. Optimizing the Selector with RL is time-consuming, and so is compositing the initial samples for training it. Future work could address this by parameterizing the composition process and learn these parameters instead of compositing the pairs directly. It could also learn to select particular frames in a video, and avoid the computational cost of temporal redundancy. Finally, another possible direction is to learn what samples to discard from the initial dataset itself.

7 Conclusion

While standard data augmentation strategies in action recognition are handcrafted, we propose to learn which pairs of videos are good to composite. In order to do this, our approach leverages three components. We train a Selector optimized with RL to choose which pairs of videos are good to composite. We reduce the search space by using samples from semantically similar classes. We perform a clean segmentation for mixing samples and remove actors as well as objects from foreground and background samples. With this, we obtain state-of-the-art results in semi-supervised and few-shot action recognition settings, and improve in the fully supervised setting. In particular, we see gains of up to 8.6% and 3.7% in the semi-supervised and few-shot settings. We also see an improvement of up to 17.4% when compared to standard augmentation in the fully supervised setting when training from scratch.

References

1. Arazo, E., Ortego, D., Albert, P., O'Connor, N.E., McGuinness, K.: Pseudo-labeling and confirmation bias in deep semi-supervised learning. In: 2020 International Joint Conference on Neural Networks (IJCNN). pp. 1–8. IEEE (2020)
2. Berthelot, D., Carlini, N., Cubuk, E.D., Kurakin, A., Sohn, K., Zhang, H., Raffel, C.: Remixmatch: Semi-supervised learning with distribution matching and augmentation anchoring. In: International Conference on Learning Representations (2019)
3. Berthelot, D., Carlini, N., Goodfellow, I., Papernot, N., Oliver, A., Raffel, C.A.: Mixmatch: A holistic approach to semi-supervised learning. Advances in Neural Information Processing Systems **32** (2019)
4. Carreira, J., Zisserman, A.: Quo vadis, action recognition? a new model and the kinetics dataset. In: proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. pp. 6299–6308 (2017)
5. Choi, J., Gao, C., Messou, J.C., Huang, J.B.: Why can't i dance in the mall? learning to mitigate scene bias in action recognition. In: NeurIPS (2019)
6. Cubuk, E.D., Zoph, B., Mane, D., Vasudevan, V., Le, Q.V.: Autoaugment: Learning augmentation strategies from data. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR) (June 2019)
7. Cubuk, E.D., Zoph, B., Shlens, J., Le, Q.: Randaugment: Practical automated data augmentation with a reduced search space. In: Larochelle, H., Ranzato, M., Hadsell, R., Balcan, M.F., Lin, H. (eds.) Advances in Neural Information Processing Systems. vol. 33, pp. 18613–18624. Curran Associates, Inc. (2020), https://proceedings.neurips.cc/paper/2020/file/d85b63ef0ccb114d0a3bb7b7d808028f-Paper.pdf
8. Deng, J., Dong, W., Socher, R., Li, L.J., Li, K., Fei-Fei, L.: Imagenet: A large-scale hierarchical image database. In: 2009 IEEE conference on computer vision and pattern recognition. pp. 248–255. Ieee (2009)
9. DeVries, T., Taylor, G.W.: Dataset augmentation in feature space. In: ICLR (Workshop) (2017)
10. DeVries, T., Taylor, G.W.: Improved regularization of convolutional neural networks with cutout. arXiv preprint arXiv:1708.04552 (2017)
11. Gowda, S.N., Rohrbach, M., Sevilla-Lara, L.: Smart frame selection for action recognition. Proceedings of the AAAI Conference on Artificial Intelligence **35**(2), 1451–1459 (May 2021), https://ojs.aaai.org/index.php/AAAI/article/view/16235
12. Gowda, S.N., Sevilla-Lara, L., Kim, K., Keller, F., Rohrbach, M.: A new split for evaluating true zero-shot action recognition. In: 43rd DAGM German Conference on Pattern Recognition (2021)
13. Grandvalet, Y., Bengio, Y., et al.: Semi-supervised learning by entropy minimization. CAP **367**, 281–296 (2005)
14. He, K., Gkioxari, G., Dollár, P., Girshick, R.: Mask R-CNN. In: ICCV (2017)
15. He, K., Zhang, X., Ren, S., Sun, J.: Identity mappings in deep residual networks. In: European conference on computer vision. pp. 630–645. Springer (2016)
16. Huang, D.A., Ramanathan, V., Mahajan, D., Torresani, L., Paluri, M., Li, F.F., Niebles, J.C.: What makes a video a video: Analyzing temporal information in video understanding models and datasets. pp. 7366–7375 (06 2018). https://doi.org/10.1109/CVPR.2018.00769

17. Iosifidis, A., Tefas, A., Pitas, I.: Semi-supervised classification of human actions based on neural networks. In: 2014 22nd International Conference on Pattern Recognition. pp. 1336–1341. IEEE (2014)
18. Jing, L., Parag, T., Wu, Z., Tian, Y., Wang, H.: Videossl: Semi-supervised learning for video classification. In: Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision (WACV). pp. 1110–1119 (January 2021)
19. Karpathy, A., Toderici, G., Shetty, S., Leung, T., Sukthankar, R., Fei-Fei, L.: Large-scale video classification with convolutional neural networks. In: Proceedings of the IEEE conference on Computer Vision and Pattern Recognition. pp. 1725–1732 (2014)
20. Korbar, B., Tran, D., Torresani, L.: Scsampler: Sampling salient clips from video for efficient action recognition. In: Proceedings of the IEEE International Conference on Computer Vision. pp. 6232–6242 (2019)
21. Kuehne, H., Jhuang, H., Garrote, E., Poggio, T., Serre, T.: Hmdb: a large video database for human motion recognition. In: 2011 International Conference on Computer Vision. pp. 2556–2563. IEEE (2011)
22. Kuo, C.W., Ma, C.Y., Huang, J.B., Kira, Z.: Featmatch: Feature-based augmentation for semi-supervised learning. In: European Conference on Computer Vision. pp. 479–495. Springer (2020)
23. Laine, S., Aila, T.: Temporal ensembling for semi-supervised learning. arXiv preprint arXiv:1610.02242 (2016)
24. Lee, D.H., et al.: Pseudo-label: The simple and efficient semi-supervised learning method for deep neural networks. In: Workshop on challenges in representation learning, ICML. vol. 3, p. 896 (2013)
25. Lemley, J., Bazrafkan, S., Corcoran, P.M.: Smart augmentation learning an optimal data augmentation strategy. IEEE Access **5**, 5858–5869 (2017)
26. Lin, T.Y., Maire, M., Belongie, S., Bourdev, L., Girshick, R., Hays, J., Perona, P., Ramanan, D., Zitnick, C.L., Dollár, P.: Microsoft coco: Common objects in context (2014), http://arxiv.org/abs/1405.0312
27. Liu, G., Reda, F.A., Shih, K.J., Wang, T.C., Tao, A., Catanzaro, B.: Image inpainting for irregular holes using partial convolutions. In: Proceedings of the European Conference on Computer Vision (ECCV). pp. 85–100 (2018)
28. Pagliardini, M., Gupta, P., Jaggi, M.: Unsupervised Learning of Sentence Embeddings using Compositional n-Gram Features. In: NAACL 2018 - Conference of the North American Chapter of the Association for Computational Linguistics (2018)
29. Perrett, T., Masullo, A., Burghardt, T., Mirmehdi, M., Damen, D.: Temporal-relational crosstransformers for few-shot action recognition. arXiv preprint arXiv:2101.06184 (2021)
30. Singh, A., Chakraborty, O., Varshney, A., Panda, R., Feris, R., Saenko, K., Das, A.: Semi-supervised action recognition with temporal contrastive learning. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 10389–10399 (2021)
31. Snell, J., Swersky, K., Zemel, R.S.: Prototypical networks for few-shot learning. arXiv preprint arXiv:1703.05175 (2017)
32. 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)
33. Sun, D., Vlasic, D., Herrmann, C., Jampani, V., Krainin, M., Chang, H., Zabih, R., Freeman, W.T., Liu, C.: Autoflow: Learning a better training set for optical flow. In: CVPR (2021)

34. Tarvainen, A., Valpola, H.: Mean teachers are better role models: Weight-averaged consistency targets improve semi-supervised deep learning results. arXiv preprint arXiv:1703.01780 (2017)
35. Williams, R.J.: Simple statistical gradient-following algorithms for connectionist reinforcement learning. Machine learning **8**(3), 229–256 (1992)
36. Yoon, J., Arik, S., Pfister, T.: Data valuation using reinforcement learning. In: International Conference on Machine Learning. pp. 10842–10851. PMLR (2020)
37. Yun, S., Han, D., Oh, S.J., Chun, S., Choe, J., Yoo, Y.: Cutmix: Regularization strategy to train strong classifiers with localizable features. In: International Conference on Computer Vision (ICCV) (2019)
38. Yun, S., Oh, S.J., Heo, B., Han, D., Kim, J.: Videomix: Rethinking data augmentation for video classification. arXiv preprint arXiv:2012.03457 (2020)
39. Zhai, X., Oliver, A., Kolesnikov, A., Beyer, L.: S4l: Self-supervised semi-supervised learning. In: Proceedings of the IEEE/CVF International Conference on Computer Vision. pp. 1476–1485 (2019)
40. Zhang, H., Zhang, L., Qi, X., Li, H., Torr, P.H., Koniusz, P.: Few-shot action recognition with permutation-invariant attention. In: Proceedings of the European Conference on Computer Vision (ECCV). Springer (2020)
41. Zhang, H., Cisse, M., Dauphin, Y.N., Lopez-Paz, D.: mixup: Beyond empirical risk minimization. In: International Conference on Learning Representations (2018)
42. Zhang, Y., Jia, G., Chen, L., Zhang, M., Yong, J.: Self-paced video data augmentation by generative adversarial networks with insufficient samples. In: Proceedings of the 28th ACM International Conference on Multimedia. p. 1652–1660. MM '20, Association for Computing Machinery, New York, NY, USA (2020). https://doi.org/10.1145/3394171.3414003, `https://doi.org/10.1145/3394171.3414003`
43. Zou, Y., Choi, J., Wang, Q., Huang, J.: Learning representational invariances for data-efficient action recognition. CoRR **abs/2103.16565** (2021), `https://arxiv.org/abs/2103.16565`