CLASTER: Clustering with Reinforcement Learning for Zero-Shot Action Recognition

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

[1] University of Edinburgh
[2] Meta AI

Abstract. Zero-Shot action recognition is the task of recognizing action classes without visual examples. The problem can be seen as learning a representation on seen classes which generalizes well to instances of unseen classes, without losing discriminability between classes. Neural networks are able to model highly complex boundaries between visual classes, which explains their success as supervised models. However, in Zero-Shot learning, these highly specialized class boundaries may overfit to the seen classes and not transfer well from seen to unseen classes. We propose a novel cluster-based representation, which regularizes the learning process, yielding a representation that generalizes well to instances from unseen classes. We optimize the clustering using reinforcement learning, which we observe is critical. We call the proposed method CLASTER and observe that it consistently outperforms the state-of-the-art in all standard Zero-Shot video datasets, including UCF101, HMDB51 and Olympic Sports; both in the standard Zero-Shot evaluation and the generalized Zero-Shot learning. We see improvements of up to 11.9% over SOTA.
Project Page: `https://sites.google.com/view/claster-zsl/home`

Keywords: Zero-Shot, Clustering, Action Recognition

1 Introduction

Research on action recognition in videos has made rapid progress in the last years, with models becoming more accurate and even some datasets becoming saturated. Much of this progress has depended on large scale training sets. However, it is often not practical to collect thousands of video samples for a new class. This idea has led to research in the Zero-Shot learning (ZSL) domain, where training occurs in a set of seen classes, and testing occurs in a set of unseen classes. In particular, in the case of video ZSL, each class label is typically enriched with semantic embeddings. These embeddings are sometimes manually annotated, by providing attributes of the class, and other times computed automatically using language models of the class name or class description. At test time the semantic embedding of the predicted seen class is used to search for a

nearest neighbor in the space of semantic embeddings of unseen classes.

While ZSL is potentially a very useful technology, this standard pipeline poses a fundamental representation challenge. Neural networks have proven extraor-

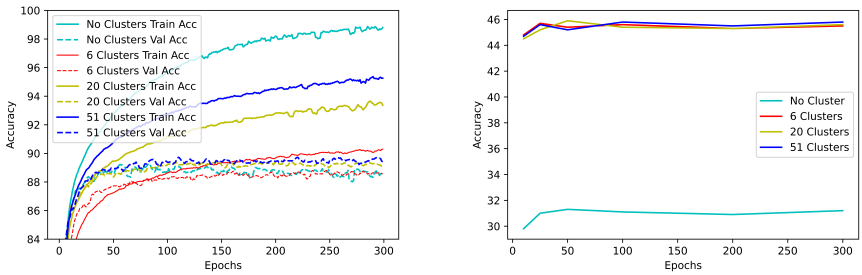

Fig. 1. Left: Learning curve for the seen classes. Right: Learning curve for the unseen classes. The clustering-based representation avoids overfitting, which in the case of seen classes means that the gap between validation and training accuracy is smaller than in the vanilla representation. This regularization effect improves the validation accuracy in unseen classes.

dinarily powerful at learning complex discriminative functions of classes with many modes. In other words, instances of the same class can be very different and still be projected by the neural network to the same category. While this works well in supervised training, it can be a problem in Zero-Shot recognition, where the highly specialized discriminative function might not transfer well to instances of unseen classes. In this work, we adress this representation problem using three main ideas.

First, we turn to clustering, and use the centroids of the clusters to represent a video. We argue that centroids are more robust to outliers, and thus help regularize the representation, avoiding overfitting to the space of seen classes. Figure 1 shows that the gap between training and validation accuracy is smaller when using clustering in seen classes (left). As a result, the learned representation is more general, which significantly improves accuracy in unseen classes (right).

Second, our representation is a combination of a visual and a semantic representation. The standard practice at training time is to use a visual representation, and learn a mapping to the semantic representation. Instead, we use both cues, which we show yields a better representation. This is not surprising, since both visual and semantic information can complement each other.

Third, we use the signal from classification as direct supervision for clustering, by using Reinforcement Learning (RL). Specifically, we use the REINFORCE algorithm to directly update the cluster centroids. This optimization improves the clustering significantly and leads to less noisy and more compact representations for unseen classes.

These three pieces are essential to learn a robust, generalizable representation of videos for Zero-Shot action recognition, as we show in the ablation study. Crucially, none of them have been used in the context of Zero-Shot or action recognition. They are simple, yet fundamental and we hope they will be useful to anyone in the Zero-Shot and action recognition communities.

We call the proposed method CLASTER, for *CLustering with reinforcement learning for Action recognition in zero-ShoT lEaRning*, and show that it significantly outperforms all existing methods across all standard Zero-Shot action recognition datasets and tasks.

2 Related Work

Fully Supervised Action Recognition. This is the most widely studied setting in action recognition, where there is a large amount of samples at training time and the label spaces are the same at training and testing time. A thorough survey is beyond our scope, but as we make use of these in the backbone of our model, we mention some of the most widely used work. The seminal work of Simonyan and Zisserman [39] introduced the now standard two-stream deep learning framework, which combines spatial and temporal information. Spatio-temporal CNNs [41,35,5] are also widely used as backbones for many applications, including this work. More recently, research has incorporated attention [42,14] and leveraged the multi-modal nature of videos [2]. In this work, we use the widely used I3D [5].

Zero-Shot Learning. Early approaches followed the idea of learning semantic classifiers for seen classes and then classifying the visual patterns by predicting semantic descriptions and comparing them with descriptions of unseen classes. In this space, Lampert et al. [21] propose attribute prediction, using the posterior of each semantic description. The SJE model [1] uses multiple compatibility functions to construct a joint embedding space. ESZSL [37] uses a Frobenius norm regularizer to learn an embedding space. Repurposing these methods for action classification is not trivial. In videos, there are additional challenges: action labels need more complex representations than objects and hence give rise to more complex manual annotations.

ZSL for Action Recognition. Early work [36] was restricted to cooking activities, using script data to transfer to unseen classes. Gan et al. [11] consider each action class as a domain, and address semantic representation identification as a multi-source domain generalization problem. Manually specified semantic representations are simple and effective [49] but labor-intensive to annotate. To overcome this, the use of label embeddings has proven popular, as only category names are needed. Some approaches use common embedding space between class labels and video features [46,45], pairwise relationships between classes [9], error-correcting codes [34], inter-class relationships [10], out-of-distribution detectors [26], and Graph Neural networks [13]. In contrast, we are learning to

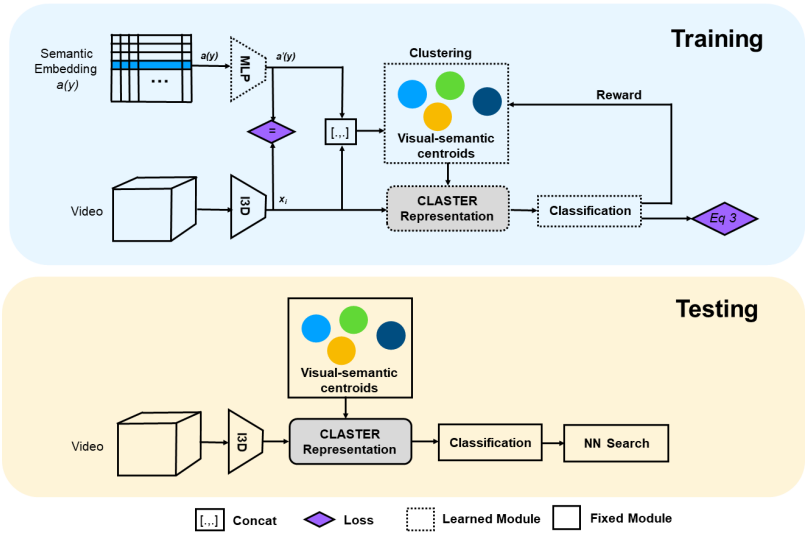

Fig. 2. Overview of CLASTER. We map the semantic embedding $a(y_i)$ to the space of visual features x_i, and concatenate both to obtain a visual-semantic representation. We cluster these visual-semantic representations with K-means to obtain initial cluster centroids. Each video is represented as the sum of the visual-semantic representation and the centroid clusters, weighted by their distance (see Sec. 3.3 and Fig. 3). This is used as input for classification (Sec 3.4 and Eq. 3). Based on the classification result, we send a reward to optimize the cluster centroids using REINFORCE (Sec. 3.5). At test time, we first perform classification on the seen classes and then do a nearest neighbor (NN) search to predict the unseen class.

optimize centroids of visual semantic representations that generalize better to unseen classes.

Reinforcement Learning for Zero-Shot Learning. RL for ZSL in images was introduced by Liu et al. [23] by using a combination of ontology and RL. In Zero-Shot text classification, Ye et al. [47] propose a self-training method to leverage unlabeled data. RL has also been used in the Zero-Shot setting for task generalization [32], active learning [7], and video object segmentation [16]. To the best of our knowledge, there is no previous work using RL for optimizing centroids in Zero-Shot recognition.

Deep Approaches to Centroid Learning for Classification. Since our approach learns cluster centroids using RL, it is related to the popular cluster learning strategy for classification called Vector of Locally Aggregated Descriptors (VLAD) [3]. The more recent NetVLAD [3] leverages neural networks which helps outperform the standard VLAD by a wide margin. ActionVLAD [15] aggregates NetVLAD over time to obtain descriptors for videos. ActionVLAD uses

clusters that correspond to spatial locations in a video while we use joint visual semantic embeddings for the entire video. In general, VLAD uses residuals with respect to cluster centroids as representation while CLASTER uses a weighting of the centroids. The proposed CLASTER outperforms NetVLAD by a large margin on both HMDB51 and UCF101.

3 CLASTER

We now describe the proposed CLASTER, which leverages clustering of visual and semantic features for video action recognition and optimizes the clustering with RL. Figure 2 shows an overview of the method.

3.1 Problem Definition

Let S be the training set of seen classes. S is composed of tuples $(x, y, a(y))$, where x represents the spatio-temporal features of a video, y represents the class label in the set of Y_S seen class labels, and $a(y)$ denotes the category-specific semantic representation of class y. These semantic representations are either manually annotated or computed using a language-based embedding of the category name, such as word2vec [28] or sentence2vec [33].

Let U be the set of pairs $(u, a(u))$, where u is a class in the set of unseen classes Y_U and $a(u)$ are the corresponding semantic representations. The seen classes Y_S and the unseen classes Y_U do not overlap.

In the Zero-Shot Learning (ZSL) setting, given an input video the task is to predict a class label in the unseen classes, as $f_{ZSL} : X \rightarrow Y_U$. In the related generalized Zero-Shot learning (GZSL) setting, given an input video, the task is to predict a class label in the union of the seen and unseen classes, as $f_{GZSL} : X \rightarrow Y_S \cup Y_U$.

3.2 Visual-Semantic Representation

Given video i, we compute visual features x_i and a semantic embedding $a(y_i)$ of their class y_i (see Sec. 4 for details). The goal is to map both to the same space, so that they have the same dimensionality and magnitude, and therefore they will have a similar weight during clustering. We learn this mapping with a simple multi-layer perceptron (MLP) [48], trained with a least-square loss. This loss minimizes the distance between x_i and the output from the MLP, which we call $a'(y)$. Finally, we concatenate x_i and $a'(y)$ to obtain the visual-semantic representations that will be clustered. The result is a representation which is not only aware of the visual information but also the semantic.

3.3 CLASTER Representation

We now detail how we represent videos using the proposed *CLASTER* representation, which leverages clustering as a form of regularization. In other words, a

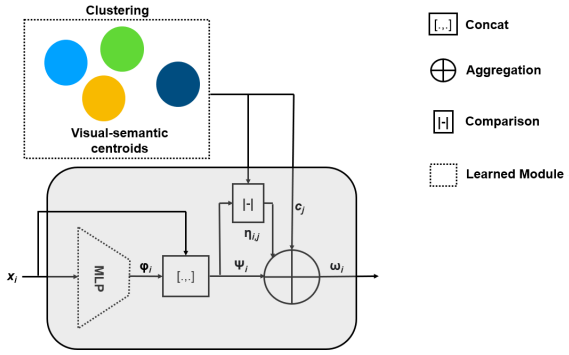

Fig. 3. The proposed CLASTER Representation in detail (see Fig. 2 for the overview of the full method). The visual feature is mapped to match the space of the visual-semantic cluster centroids with an MLP and concatenation. Based on the distances to the cluster centroids the final representation ω is a weighted representation of the centroids, more robust to the out-of-distribution instances of the unseen test classes. Details in Sec. 3.3 and Eq. 1.

representation w.r.t centroids is more robust to outliers, which is helpful since all instances of the unseen classes are effectively outliers w.r.t. the training distribution.

We initialize the clustering of the training set S using K-means [8]. Each resulting cluster j has a centroid c_j, that is the average of all visual-semantic samples in that particular cluster. The CLASTER representation of a given video is the sum of the visual-semantic representation and the centroids, weighted by the inverse of the distance, such that closer clusters will have more weight. Figure 3 shows this process in detail.

Specifically, given video i, we compute the visual representation x_i. We estimate the semantic vector ϕ_i using an MLP. This is a necessary step, as during test time we do not have any semantic information. Concatenating the visual x_i and semantic ϕ_i we obtain the intermediate representation ψ_i, which is in the same space as the cluster centroids.

We compute the Euclidean distance $d_{i,j}$ between the visual-semantic point ψ_i and each cluster j, which we refer to as $d_{i,j}$. We take the inverse $1/d_{i,j}$ and normalize them using their maximum and minimum values, such that they are between 0 and 1. We refer to these normalized values as $\eta_{i,j}$, and they are used as the weights of each cluster centroid in the final CLASTER representation ω_i:

$$\omega_i = \psi_i + \sum_{j=1}^{k} \eta_{i,j} c_j. \tag{1}$$

3.4 Loss Function

Given the CLASTER representation ω_i we predict a seen class using a simple MLP, V. Instead of the vanilla softmax function, we use semantic softmax [18], which includes the semantic information $a(y_i)$ and thus can transfer better to Zero-Shot classes:

$$\hat{y}_i = \frac{e^{a(y_i)^T V(\omega_i)}}{\sum_{j=1}^{S} e^{a(y_j)^T V(\omega_i)}}. \tag{2}$$

The output $\hat{y}_i$ is a vector with a probability distribution over the S seen classes. We train the classifier, which minimizes the cross-entropy loss with a regularization term:

$$\min_W \sum_{i=1}^{N} \mathcal{L}(x_i) + \lambda \left\| W \right\|, \tag{3}$$

where W refers to all weights in the network.

3.5 Optimization with Reinforcement Learning

As there is no ground truth for clustering centroids, we use the classification accuracy as supervision. This makes the problem non-differentiable, and therefore we cannot use traditional gradient descent. Instead, we use RL to optimize the cluster centroids. For this, we compute two variables that will determine each centroid update: the reward, which measures whether the classification is correct and will determine the direction of the update, and the classification score, which measures how far the prediction is from the correct answer and will determine the magnitude of the update.

Given the probabilistic prediction $\hat{y}_i$ and the one-hot representation of the ground truth class y_i, we compute the classification score as the dot product of the two: $z_i = y_i \hat{y}_i$. To obtain the reward, we check if the maximum of $\hat{y}_i$ and y_i lie in the same index:

$$r = \begin{cases} 1 \ if \ \ \arg\max \hat{y}_i = \arg\max y_i \\ -1 \ otherwise \end{cases} \tag{4}$$

This essentially gives a positive reward if the model has predicted a correct classification and a negative reward if the classification was incorrect. This formulation is inspired by Likas [22], which was originally proposed for a different domain and the problem of competitive learning.

For each data point ψ_i we only update the closest cluster centroid c_j. We compute the update Δc_j using the REINFORCE [22] algorithm as:

$$\Delta c_j = \alpha \ r \ (z_i - p_j) \ (\psi_i - c_j). \tag{5}$$

For further details on this derivation, please see the Supplementary Material as well as Likas [22]. The main difference between our model and Likas' is that we do not consider cluster updates to be Bernoulli units. Instead, we modify the

cluster centroid with the classification score z_i, which is continuous in the range between 0 and 1.

4 Implementation Details

Visual features. We use RGB and flow features extracted from the *Mixed 5c* layer of an I3D network pre-trained on the Kinetics [5] dataset. The *Mixed 5c* output of the flow network is averaged across the temporal dimension and pooled by four in the spatial dimension and then flattened to a vector of size 4096. We then concatenate the two.

Network architecture. The MLP that maps the semantic features to visual features consists of two fully-connected (FC) layers and a ReLU. The MLP in the CLASTER Representation module, which maps the visual feature to the semantic space is a two-layer FC network, whose output after concatenation with the video feature has the same dimensions as the cluster representatives. The size of the FC layers is 8192 each. The final classification MLP (represented as a classification block in Figure 2) consists of two convolutional layers and two FC layers, where the last layer equals the number of seen classes in the dataset we are looking at. All the modules are trained with the Adam optimizer with a learning rate of 0.0001 and weight decay of 0.0005.

Number of clusters. Since the number of clusters is a hyperparameter, we evaluate the effect of the number of clusters on the UCF101 dataset for videos and choose 6 after the average performance stablizes as can be seen in the supplementary material. We then use the same number for the HMDB51 and Olympics datasets.

RL optimization. We use 10,000 iterations and the learning rate α is fixed to 0.1 for the first 1000 iterations, 0.01 for the next 1000 iterations and then drop it to 0.001 for the remaining iterations.

Semantic embeddings. We experiment with three types of embeddings as semantic representations of the classes. We have human-annotated semantic representations for UCF101 and the Olympic sports dataset of sizes 40 and 115 respectively. HMDB51 does not have such annotations. Instead, we use a skip-gram model trained on the news corpus provided by Google to generate word2vec embeddings. Using action classes as input, we obtain a vector representation of 300 dimensions. Some class labels contain multiple words. In those cases, we use the average of the word2vec embeddings. We also use sentence2vec embeddings, trained on Wikipedia. These can be obtained for both single words and multi-word expressions. The elaborate descriptions are taken from [6] and only evaluated for fair comparison to them.

For the elaborative descriptions, we follow ER [6] and use the provided embeddings in their codebase.

Rectification of the Semantic Embedding Sometimes, in ZSL, certain data points tend to appear as nearest-neighbor of many other points in the projection space. This is referred to as the hubness problem [38]. We avoid this problem using semantic rectification [24], where the class representation is modified by averaging the output generated by the projection network, which in our case is the penultimate layer of the classification MLP. Specifically, for the unseen classes, we perform rectification by first using the MLP trained on the seen classes to project the semantic embedding to the visual space. We add the average of projected semantic embeddings from the k-nearest neighbors of the seen classes, specifically as follows:

$$\hat{a}(y_i) = a'(y_i) + \frac{1}{k} \sum_{n \in N} cos\left(a'(y_i), n\right) \cdot n, \tag{6}$$

where $a'(y)$ refers to the embedding after projection to the visual space, $cos(a, n)$ refers to the cosine similarity between a and n, the operator $\cdot$ refers to the dot product and N refers to the k-nearest neighbors of $a'(y_{u_i})$.

Nearest Neighbor Search At test time in ZSL, given a test video, we predict a seen class and compute or retrieve its semantic representation. After rectification, we find the nearest neighbor in the set of unseen classes. In the GZSL task, class predictions may be of seen or unseen classes. Thus, we first use a bias detector [12] which helps us detect if the video belongs to the seen or unseen class. If it belongs to a seen class, we predict the class directly from our model, else we proceed as in ZSL.

5 Experimental Analysis

In this section, we look at the qualitative and quantitative performance of the proposed model. We first describe the experimental settings, and then show an ablation study, that explores the contribution of each component. We then compare the proposed method to the state-of-the-art in the ZSL and GZSL tasks, and give analytical insights into the advantages of CLASTER.

5.1 Datasets

We choose the Olympic Sports [31], HMDB-51 [20] and UCF-101 [40], so that we can compare to recent state-of-the-art models [11,26,34]. We follow the commonly used 50/50 splits of Xu et al. [45], where 50 percent are seen classes and 50 are unseen classes. Similar to previous approaches [49,11,34,27,19], we report average accuracy and standard deviation over 10 independent runs. We report

results on the split proposed by [44], in the standard inductive setting. We also report on the recently introduced TruZe [17]. This split accounts for the fact that some classes present on the dataset used for pre-training (Kinetics [5]) overlap with some of the unseen classes in the datasets used in the Zero-Shot setting, therefore breaking the premise that those classes have not been seen.

5.2 Ablation Study

Table 1 shows the impact of using the different components of CLASTER.

Impact of Clustering. We consider several baselines. First, omitting clustering, which is the equivalent of setting $\omega_i = \psi_i$, in Eq. 1. This is, ignoring the cluster centroids in the representation. This is referred to in Table 1 as "No clustering". Second, we use random clustering, which is assigning each instance to a random cluster. Finally, we use the standard K-means. We observe that using clusters is beneficial, but only if they are meaningful, as in the case of K-means.

Impact of using a visual-semantic representation. We compare to the standard representation, which only includes visual information, and keep everything the same. This is, clustering and classification are done using only the visual features. This is referred to in the table as "CLASTER w/o SE". We observe that there is a very wide gap between using and not using the semantic features at training time. This effect is present across all datasets, suggesting it is a general improvement in the feature learning. We also show a comparison of aggregation strategies and interaction between visual and semantic features in the supplementary material.

Impact of different optimization choices. We make cluster centroids learnable parameters and use the standard SGD to optimize them ('CLASTER w/o RL") We also test the use of the related work of NetVLAD to optimize the cluster ("CLASTER w/ NetVLAD"). We see that the proposed model outperforms NetVLAD by an average of 4.7% and the CLASTER w/o RL by 7.3% on the UCF101 dataset. A possible reason for this difference is that the loss is back-propagated through multiple parts of the model before reaching the centroids. However, with RL the centroids are directly updated using the reward signal. Section 5.6 explores how the clusters change after the RL optimization. In a nutshell, the RL optimization essentially makes the clusters cleaner, moving most instances in a class to the same cluster.

5.3 Results on ZSL

Table 2 shows the comparison between CLASTER and several state-of-the-art methods: the out-of-distribution detector method (OD) [26], a generative approach to Zero-Shot action recognition (GGM) [30], the evaluation of output

Component	HMDB51	Olympics	UCF101
No clustering	25.6 ± 2.8	57.7 ± 3.1	31.6 ± 4.6
Random clustering (K=6)	20.2 ± 4.2	55.4 ± 3.1	24.1 ± 6.3
K-means (K=6)	27.9 ± 3.7	58.6 ± 3.5	35.3 ± 3.9
CLASTER w/o SE	27.5 ± 3.8	55.9 ± 2.9	39.4 ± 4.4
CLASTER w/o RL	30.1 ± 3.4	60.5 ± 1.9	39.1 ± 3.2
CLASTER w/ NetVLAD	33.2 ± 2.8	62.6 ± 4.1	41.7 ± 3.8
CLASTER	**36.8 ± 4.2**	**63.5 ± 4.4**	**46.4 ± 5.1**

Table 1. Results of the ablation study of different components of CLASTER ZSL. The study shows the effect of clustering, using visual-semantic representations, and optimizing with different methods. All three components show a wide improvement over the various baselines, suggesting that they are indeed complementary to improve the final representation.

embeddings (SJE) [1], the feature generating networks (WGAN) [43], the end-to-end training for realistic applications approach (E2E) [4], the inverse autoregressive flow (IAF) based generative model, bi-directional adversarial GAN(Bi-dir GAN) [29] and prototype sampling graph neural network (PS-GNN) [13]. To make results directly comparable, we use the same backbone across all of them, which is the I3D [5] pre-trained on Kinetics.

We observe that the proposed CLASTER consistently outperforms all other state-of-the-art methods across all datasets. The improvements are significant: up to 3.5% on HMDB51 and 13.5% on UCF101 with manual semantic embedding. We also measure the impact of different semantic embeddings, including using sentence2vec instead of word2vec. We show that sentence2vec significantly improves over using word2vec, especially on UCF101 and HMDB51. Combination of embeddings resulted in average improvements of 0.3%, 0.8% and 0.9% over the individual best performing embedding of CLASTER.

5.4 Results on GZSL

We now compare to the same approaches in the GZSL task in Table 3, the reported results are the harmonic mean of the seen and unseen class accuracies. Here CLASTER outperforms all previous methods across different modalities. We obtain an improvement on average of 2.6% and 5% over the next best performing method on the Olympics dataset using manual representations and word2vec respectively. We obtain an average improvement of 6.3% over the next best performing model on the HMDB51 dataset using word2vec. We obtain an improvement on average performance by 1.5% and 4.8% over the next best performing model on the UCF101 dataset using manual representations and word2vec respectively. Similarly to ZSL, we show generalized performance improvements using sentence2vec. We also report results on the combination of embeddings. We see an improvement of 0.3%, 0.6% and 0.4% over the individual

Method	SE	Olympics	HMDB51	UCF101
SJE [1]	M	47.5 ± 14.8	-	12.0 ± 1.2
Bi-Dir GAN [29]	M	53.2 ± 10.5	-	24.7 ± 3.7
IAF [29]	M	54.9 ± 11.7	-	26.1 ± 2.9
GGM [30]	M	57.9 ± 14.1	-	24.5 ± 2.9
OD [26]	M	65.9 ± 8.1	-	38.3 ± 3.0
WGAN [43]	M	64.7 ± 7.5	-	37.5 ± 3.1
CLASTER (ours)	M	**67.4 ± 7.8**	-	**51.8 ± 2.8**
SJE [1]	W	28.6 ± 4.9	13.3 ± 2.4	9.9 ± 1.4
IAF [29]	W	39.8 ± 11.6	19.2 ± 3.7	22.2 ± 2.7
Bi-Dir GAN [29]	W	40.2 ± 10.6	21.3 ± 3.2	21.8 ± 3.6
GGM [30]	W	41.3 ± 11.4	20.7 ± 3.1	20.3 ± 1.9
WGAN [43]	W	47.1 ± 6.4	29.1 ± 3.8	25.8 ± 3.2
OD [26]	W	50.5 ± 6.9	30.2 ± 2.7	26.9 ± 2.8
PS-GNN [13]	W	61.8 ± 6.8	32.6 ± 2.9	43.0 ± 4.9
E2E [4]*	W	61.4 ± 5.5	33.1 ± 3.4	46.2 ± 3.8
CLASTER (ours)	W	**63.8 ± 5.7**	**36.6 ± 4.6**	**46.7 ± 5.4**
CLASTER (ours)	S	**64.2 ± 3.3**	**41.8 ± 2.1**	**50.2 ± 3.8**
CLASTER (ours)	C	**67.7 ± 2.7**	**42.6 ± 2.6**	**52.7 ± 2.2**
ER [6]	ED	60.2 ± 8.9	35.3 ± 4.6	51.8 ± 2.9
CLASTER (ours)	ED	**68.4 ± 4.1**	**43.2 ± 1.9**	**53.9 ± 2.5**

Table 2. Results on ZSL. SE: semantic embedding, M: manual representation, W: word2vec embedding, S: sentence2vec, C: Combination of embeddings. The proposed CLASTER outperforms previous state-of-the-art across tasks and datasets.

best embedding for CLASTER. The seen and unseen accuracies are shown in the Supplemental Material.

5.5 Results on TruZe

We also evaluate on the more challenging TruZe split. The proposed UCF101 and HMDB51 splits have 70/31 and 29/22 classes (represented as training/testing). We compare to WGAN [43], OD [26] and E2E [4] on both ZSL and GZSL scenarios. Results are shown in Table 4.

5.6 Analysis of the RL optimization

We analyze how optimizing with RL affects clustering on the UCF101 training set. Figure 4 shows the t-SNE [25] visualization. Each point is a video instance in the unseen classes, and each color is a class label. As it can be seen, the RL optimization makes videos of the same class appear closer together.

We also do a quantitative analysis of the clustering. For each class in the training set, we measure the distribution of clusters that they belong to, visualized in the Fig 5. We observe that after the RL optimization, the clustering

Method	SE	Olympics	HMDB51	UCF101
Bi-Dir GAN [29]	M	44.2 ± 11.2	-	22.7 ± 2.5
IAF [29]	M	48.4 ± 7.0	-	25.9 ± 2.6
GGM [30]	M	52.4 ± 12.2	-	23.7 ± 1.2
WGAN [43]	M	59.9 ± 5.3	-	44.4 ± 3.0
OD[26]	M	66.2 ± 6.3	-	49.4 ± 2.4
CLASTER (ours)	M	**68.8 ± 6.6**	-	**50.9 ± 3.2**
IAF [29]	W	30.2 ± 11.1	15.6 ± 2.2	20.2 ± 2.6
Bi-Dir GAN [29]	W	32.2 ± 10.5	7.5 ± 2.4	17.2 ± 2.3
SJE [1]	W	32.5 ± 6.7	10.5 ± 2.4	8.9 ± 2.2
GGM[30]	W	42.2 ± 10.2	20.1 ± 2.1	17.5 ± 2.2
WGAN [43]	W	46.1 ± 3.7	32.7 ± 3.4	32.4 ± 3.3
PS-GNN [13]	W	52.9 ± 6.2	24.2 ± 3.3	35.1 ± 4.6
OD [26]	W	53.1 ± 3.6	36.1 ± 2.2	37.3 ± 2.1
CLASTER (ours)	W	**58.1 ± 2.4**	**42.4 ± 3.6**	**42.1 ± 2.6**
CLASTER (ours)	S	**58.7 ± 3.1**	**47.4 ± 2.8**	**48.3 ± 3.1**
CLASTER (ours)	C	**69.1 ± 5.4**	**48.0 ± 2.4**	**51.3 ± 3.5**

Table 3. Results on GZSL. SE: semantic embedding, M: manual representation, W: word2vec embedding, S: sentence2vec, C: combination of embeddings. The seen and unseen class accuracies are listed in the supplementary material.

becomes "cleaner". This is, most instances in a class belong to a dominant cluster. This effect can be measured using the purity of the cluster:

$$Purity = \frac{1}{N}\sum_{i=1}^{k} max_j \left|c_i \cap t_j\right|, \tag{7}$$

where N is the number of data points (video instances), k is the number of clusters, c_i is a cluster in the set of clusters, and t_j is the class which has the maximum count for cluster c_i. Poor clustering results in purity values close to 0, and a perfect clustering will return a purity of 1. Using K-means, the purity is 0.77, while optimizing the clusters with RL results in a purity of 0.89.

Method	UCF101		HMDB51	
	ZSL	GZSL	ZSL	GZSL
WGAN	22.5	36.3	21.1	31.8
OD	22.9	42.4	21.7	35.5
E2E	45.5	45.9	31.5	38.9
CLASTER	**45.8**	**47.3**	**33.2**	**44.5**

Table 4. Results on TruZe. For ZSL, we report the mean class accuracy and for GZSL, we report the harmonic mean of seen and unseen class accuracies. All approaches use sen2vec annotations as the form of semantic embedding.

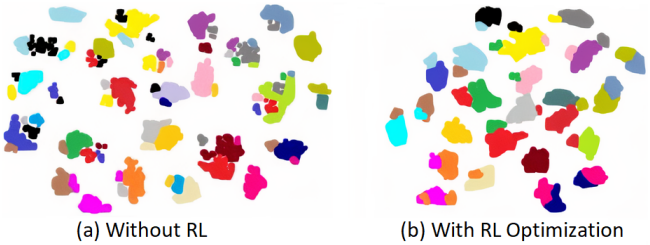

Fig. 4. CLASTER improves the representation and clustering in unseen classes. The figure shows t-SNE [25] of video instances, where each color corresponds to a unique unseen class label. The RL optimization improves the representation by making it more compact: in (b) instances of the same class, i.e. same color, are together and there are less outliers for each class compared to (a).

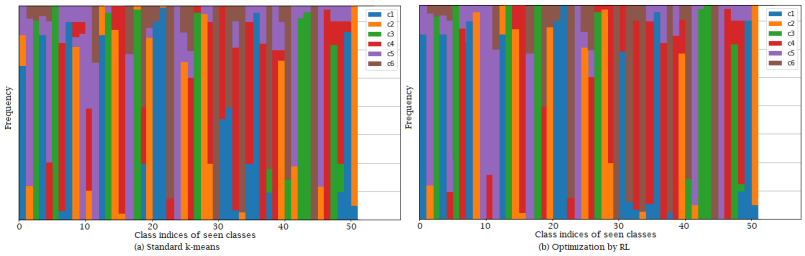

Fig. 5. Analysis of how RL optimization changes the cluster to which an instance belongs. The frequencies are represented as percentages of instances in each cluster. We can see that the clusters are a lot "cleaner" after the optimization by RL.

Finally, we observe another interesting side effect of clustering. Some of the most commonly confused classes before clustering (e.g. "Baby crawling" vs. "Mopping floor", "Breaststroke" vs. "front crawl", "Rowing vs. front crawl") are assigned to different clusters after RL, resolving confusion. This suggests that clusters are also used as a means to differentiate between similar classes.

6 Conclusion

Zero-Shot action recognition is the task of recognizing action classes without any visual examples. The challenge is to map the knowledge of seen classes at training time to that of novel unseen classes at test time. We propose a novel model that learns clustering-based representation of visual-semantic features, optimized with RL. We observe that all three of these components are essential. The clustering helps regularizing, and avoids overfitting to the seen classes. The visual-semantic representation helps improve the representation. And the RL yields better, cleaner clusters. The results is remarkable improvements across datasets and tasks over all previous state-of-the-art, up to 11.9% absolute improvement on HMDB51 for GZSL.

References

1. Akata, Z., Reed, S., Walter, D., Lee, H., Schiele, B.: Evaluation of output embeddings for fine-grained image classification. In: Proceedings of the IEEE conference on computer vision and pattern recognition. pp. 2927–2936 (2015)
2. Alwassel, H., Mahajan, D., Torresani, L., Ghanem, B., Tran, D.: Self-supervised learning by cross-modal audio-video clustering. arXiv preprint arXiv:1911.12667 (2019)
3. Arandjelovic, R., Gronat, P., Torii, A., Pajdla, T., Sivic, J.: Netvlad: Cnn architecture for weakly supervised place recognition. In: Proceedings of the IEEE conference on computer vision and pattern recognition. pp. 5297–5307 (2016)
4. Brattoli, B., Tighe, J., Zhdanov, F., Perona, P., Chalupka, K.: Rethinking zero-shot video classification: End-to-end training for realistic applications. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 4613–4623 (2020)
5. Carreira, J., Zisserman, A.: Quo vadis, action recognition? a new model and the kinetics dataset. In: IEEE Conf. Comput. Vis. Pattern Recog. (2017)
6. Chen, S., Huang, D.: Elaborative rehearsal for zero-shot action recognition. In: Proceedings of the IEEE/CVF International Conference on Computer Vision. pp. 13638–13647 (2021)
7. Fan, Y., Tian, F., Qin, T., Bian, J., Liu, T.Y.: Learning what data to learn. arXiv preprint arXiv:1702.08635 (2017)
8. Forgy, E.W.: Cluster analysis of multivariate data: efficiency versus interpretability of classifications. biometrics **21**, 768–769 (1965)
9. Gan, C., Lin, M., Yang, Y., De Melo, G., Hauptmann, A.G.: Concepts not alone: Exploring pairwise relationships for zero-shot video activity recognition. In: Thirtieth AAAI conference on artificial intelligence (2016)
10. Gan, C., Lin, M., Yang, Y., Zhuang, Y., Hauptmann, A.G.: Exploring semantic inter-class relationships (sir) for zero-shot action recognition. In: Proceedings of the National Conference on Artificial Intelligence (2015)
11. Gan, C., Yang, T., Gong, B.: Learning attributes equals multi-source domain generalization. In: Proceedings of the IEEE conference on computer vision and pattern recognition. pp. 87–97 (2016)
12. Gao, J., Zhang, T., Xu, C.: I know the relationships: Zero-shot action recognition via two-stream graph convolutional networks and knowledge graphs. In: Proceedings of the AAAI Conference on Artificial Intelligence. vol. 33, pp. 8303–8311 (2019)
13. Gao, J., Zhang, T., Xu, C.: Learning to model relationships for zero-shot video classification. IEEE Transactions on Pattern Analysis and Machine Intelligence (2020)
14. Girdhar, R., Ramanan, D.: Attentional pooling for action recognition. In: Advances in Neural Information Processing Systems. pp. 34–45 (2017)
15. Girdhar, R., Ramanan, D., Gupta, A., Sivic, J., Russell, B.: Actionvlad: Learning spatio-temporal aggregation for action classification. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. pp. 971–980 (2017)
16. Gowda, S.N., Eustratiadis, P., Hospedales, T., Sevilla-Lara, L.: Alba: Reinforcement learning for video object segmentation. arXiv preprint arXiv:2005.13039 (2020)
17. Gowda, S.N., Sevilla-Lara, L., Kim, K., Keller, F., Rohrbach, M.: A new split for evaluating true zero-shot action recognition. arXiv preprint arXiv:2107.13029 (2021)

18. Ji, Z., Sun, Y., Yu, Y., Guo, J., Pang, Y.: Semantic softmax loss for zero-shot learning. Neurocomputing **316**, 369–375 (2018)
19. Kodirov, E., Xiang, T., Fu, Z., Gong, S.: Unsupervised domain adaptation for zero-shot learning. In: Proceedings of the IEEE international conference on computer vision. pp. 2452–2460 (2015)
20. 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)
21. Lampert, C.H., Nickisch, H., Harmeling, S.: Learning to detect unseen object classes by between-class attribute transfer. In: 2009 IEEE Conference on Computer Vision and Pattern Recognition. pp. 951–958. IEEE (2009)
22. Likas, A.: A reinforcement learning approach to online clustering. Neural computation **11**(8), 1915–1932 (1999)
23. Liu, B., Yao, L., Ding, Z., Xu, J., Wu, J.: Combining ontology and reinforcement learning for zero-shot classification. Knowledge-Based Systems **144**, 42–50 (2018)
24. Luo, C., Li, Z., Huang, K., Feng, J., Wang, M.: Zero-shot learning via attribute regression and class prototype rectification. IEEE Transactions on Image Processing **27**(2), 637–648 (2017)
25. Maaten, L.v.d., Hinton, G.: Visualizing data using t-sne. Journal of machine learning research **9**(Nov), 2579–2605 (2008)
26. Mandal, D., Narayan, S., Dwivedi, S.K., Gupta, V., Ahmed, S., Khan, F.S., Shao, L.: Out-of-distribution detection for generalized zero-shot action recognition. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. pp. 9985–9993 (2019)
27. Mettes, P., Snoek, C.G.: Spatial-aware object embeddings for zero-shot localization and classification of actions. In: Proceedings of the IEEE International Conference on Computer Vision. pp. 4443–4452 (2017)
28. 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)
29. Mishra, A., Pandey, A., Murthy, H.A.: Zero-shot learning for action recognition using synthesized features. Neurocomputing **390**, 117–130 (2020)
30. Mishra, A., Verma, V.K., Reddy, M.S.K., Arulkumar, S., Rai, P., Mittal, A.: A generative approach to zero-shot and few-shot action recognition. In: 2018 IEEE Winter Conference on Applications of Computer Vision (WACV). pp. 372–380. IEEE (2018)
31. Niebles, J.C., Chen, C.W., Fei-Fei, L.: Modeling temporal structure of decomposable motion segments for activity classification. In: European conference on computer vision. pp. 392–405. Springer (2010)
32. Oh, J., Singh, S., Lee, H., Kohli, P.: Zero-shot task generalization with multi-task deep reinforcement learning. In: International Conference on Machine Learning. pp. 2661–2670 (2017)
33. Pagliardini, M., Gupta, P., Jaggi, M.: Unsupervised learning of sentence embeddings using compositional n-gram features. In: Proceedings of NAACL-HLT. pp. 528–540 (2018)
34. Qin, J., Liu, L., Shao, L., Shen, F., Ni, B., Chen, J., Wang, Y.: Zero-shot action recognition with error-correcting output codes. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. pp. 2833–2842 (2017)
35. Qiu, Z., Yao, T., Mei, T.: Learning spatio-temporal representation with pseudo-3d residual networks. In: proceedings of the IEEE International Conference on Computer Vision. pp. 5533–5541 (2017)

36. Rohrbach, M., Regneri, M., Andriluka, M., Amin, S., Pinkal, M., Schiele, B.: Script data for attribute-based recognition of composite activities. In: Eur. Conf. Comput. Vis. (2012)
37. Romera-Paredes, B., Torr, P.: An embarrassingly simple approach to zero-shot learning. In: International Conference on Machine Learning. pp. 2152–2161 (2015)
38. Shigeto, Y., Suzuki, I., Hara, K., Shimbo, M., Matsumoto, Y.: Ridge regression, hubness, and zero-shot learning. In: Joint European Conference on Machine Learning and Knowledge Discovery in Databases. pp. 135–151. Springer (2015)
39. Simonyan, K., Zisserman, A.: Two-stream convolutional networks for action recognition in videos. In: Advances in neural information processing systems. pp. 568–576 (2014)
40. 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)
41. Tran, D., Bourdev, L., Fergus, R., Torresani, L., Paluri, M.: Learning spatiotemporal features with 3d convolutional networks. In: Proceedings of the IEEE international conference on computer vision. pp. 4489–4497 (2015)
42. Wang, X., Girshick, R., Gupta, A., He, K.: Non-local neural networks. In: Proceedings of the IEEE conference on computer vision and pattern recognition. pp. 7794–7803 (2018)
43. Xian, Y., Lorenz, T., Schiele, B., Akata, Z.: Feature generating networks for zero-shot learning. In: Proceedings of the IEEE conference on computer vision and pattern recognition. pp. 5542–5551 (2018)
44. Xian, Y., Schiele, B., Akata, Z.: Zero-shot learning-the good, the bad and the ugly. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. pp. 4582–4591 (2017)
45. Xu, X., Hospedales, T., Gong, S.: Transductive zero-shot action recognition by word-vector embedding. International Journal of Computer Vision **123**(3), 309–333 (2017)
46. Xu, X., Hospedales, T.M., Gong, S.: Multi-task zero-shot action recognition with prioritised data augmentation. In: European Conference on Computer Vision. pp. 343–359. Springer (2016)
47. Ye, Z., Geng, Y., Chen, J., Chen, J., Xu, X., Zheng, S., Wang, F., Zhang, J., Chen, H.: Zero-shot text classification via reinforced self-training. In: Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics. pp. 3014–3024 (2020)
48. Zhang, L., Xiang, T., Gong, S.: Learning a deep embedding model for zero-shot learning. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. pp. 2021–2030 (2017)
49. Zhu, Y., Long, Y., Guan, Y., Newsam, S., Shao, L.: Towards universal representation for unseen action recognition. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. pp. 9436–9445 (2018)