Pattern Analysis and Applications
https://doi.org/10.1007/s10044-023-01202-7

THEORETICAL ADVANCES

Unsupervised open-world human action recognition

Matheus Gutoski[1] · **André Eugenio Lazzaretti**[1] · **Heitor Silvério Lopes**[1]

Received: 1 February 2023 / Accepted: 6 September 2023

Abstract
Open-world recognition (OWR) is an important field of research that strives to develop machine learning models capable of identifying and learning new classes as they appear. Concurrently, human action recognition (HAR) has received increasing attention from the research community. We approach Open-World HAR in the unsupervised setting. In unsupervised OWR, class labels are available for the initial classes but not for new ones. Hence, we propose a clustering method to label unknown classes automatically for incremental learning (IL). Our framework consists of an Initial Learning phase for initializing the models, an open-set recognition phase for identifying unknown classes, an Automatic Clustering phase for estimating the number of clusters and generating labels, and an IL phase for incorporating new knowledge. The proposed framework was evaluated at each phase separately in eleven experimental settings of the UCF-101 dataset. We also presented parameter sensitivity studies of the main parameters and visual analysis of misclassified videos, revealing interesting visual similarities between overlapped classes. Experiments have shown promising results in all phases of Open-World HAR, even without labels, which closely resembles real-world problems.

Keywords Clustering · Incremental learning · Human action recognition · Open world

1 Introduction

Recent advances in image and video recognition have successfully solved many problems in the supervised learning scenario. However, these problems usually make two unrealistic assumptions in real-world scenarios. The first is the closed-world assumption. Conventional models can only recognize classes seen during the training phase. This shortcoming often hinders the model's ability to solve real-world problems in evolving environments. The second assumption is the availability of labeled data. Applications such as social media, entertainment, and security continuously generate a massive volume of unlabeled video data. This raises the need for models that can learn with raw data not previously labeled.

In light of the requirements for solving real-world problems, Open-World Recognition (OWR) has recently become a hot topic in machine learning [1–3]. Open-World models automatically discover and learn new classes while also preserving past knowledge. According to [1], an OWR model requires three main components. The first is a multi-class Open-Set Recognition function capable of classifying known classes and detecting new classes. Subsequently, a labeling process is required for the unknown classes. Some recent works have used a human oracle, annotated data, or assumed the number of clusters is known in advance [1, 4]. Regarding the supervision in this step, [5] defined the Open-World Unsupervised Learner, an Open-World model that does not use labeled data for discovering new classes. The third component is an Incremental Learning (IL) function. IL is concerned with learning new classes while preserving the knowledge acquired before. This function needs to be scalable in terms of time and memory consumption.

Most recent works have approached Human Action Recognition (HAR) from the closed-world scenario [6–8]. However, HAR is a natural candidate for OWR because new human actions may arise in the real-world. Hence, it would

✉ André Eugenio Lazzaretti
lazzaretti@utfpr.edu.br

Matheus Gutoski
matheusgutoski@alunos.utfpr.edu.br

Heitor Silvério Lopes
heitorslopes@utfpr.edu.br

[1] Electrial Engineering and Industrial Informatics, Federal University of Technology - Paraná, Sete de Setembro, 3165 - Rebouças, Curitiba, Paraná 80230-901, Brazil

Published online: 26 September 2023

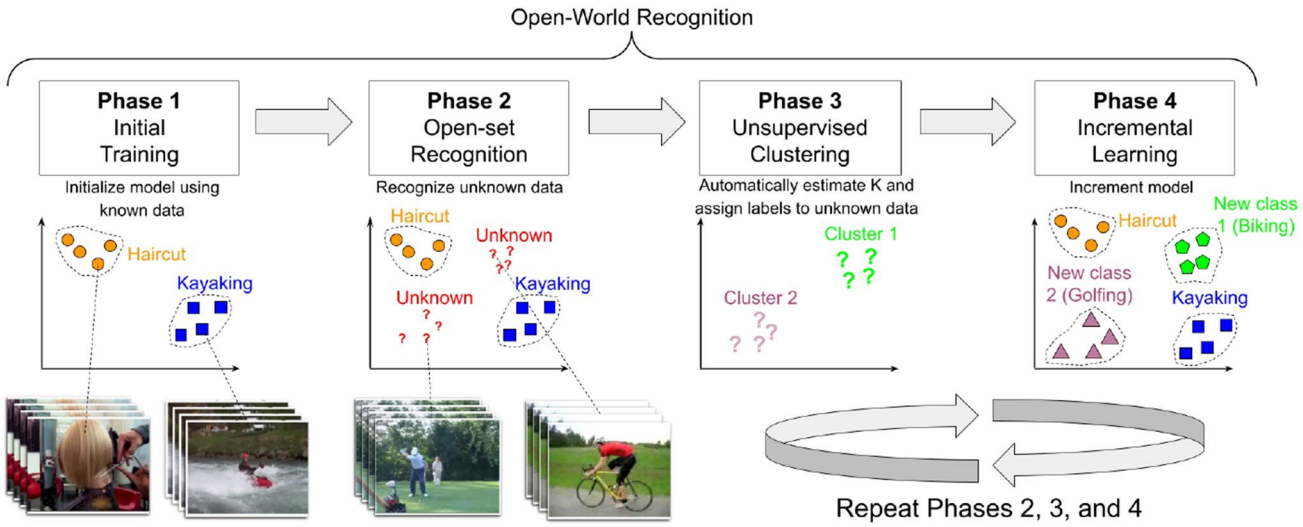

Fig. 1 Overview of the proposed model for solving OWR in videos. The model contains four phases: Initial Training, Open-Set Recognition, Unsupervised Clustering, and Incremental Learning. In phase 1, the initial classes "Haircut" and "Kayaking" are learned in a standard supervised training procedure. In phase 2, unlabeled videos from unknown classes are introduced to the problem. They are represented as "?". The model identifies videos from unknown classes by performing Open-Set Recognition. In phase 3, the unknown data is automatically clustered. The actual number of clusters is not provided in advance. In phase 4, the model incrementally learns the two newly discovered classes ("Biking" and "Golfing"). Phases 2, 3, and 4 may be repeated to include new classes. The representations and frames used in the figure are illustrative. Figure best viewed in color

not be possible to create, in advance, a dataset containing all human actions.

The model proposed in this work meets all the requirements for an Open-World, Unsupervised Learner in videos. This task is accomplished in four main phases shown in Fig. 1. In phase 1, our model performs the initial training. The model uses a standard supervised learning approach to learn the initial classes. Phase 2 introduces new classes to the problem. In this phase, we perform Open-Set Recognition (OSR). OSR is concerned with classifying known classes while rejecting unknown classes. In this work, OSR is a subtask of OWR. Once unknown videos are detected, our model automatically estimates the number of categories in the new data. This step is performed in phase 3 using a hierarchical agglomerative clustering algorithm. Finally, with the labels automatically assigned, the model executes IL. This stage is the fourth and final phase of our proposed model. The model repeats phases 2, 3, and 4 when a new task (set of classes) is introduced.

This article extends and generalizes our two previous works, which addressed OSR [9] and IL in HAR [10]. Our video feature learning model based on metric learning was introduced in [9]. The model uses the I3D [6] as the backbone convolutional network and a Triplet Network that minimizes the cosine triplet loss. The I3D outputs fixed representations that feed the Triplet Network, which learns dynamic representations using new data and a few exemplars. The I3D features are said to be fixed because the model is trained only once during the initial training phase, while the Triplet Network is updated multiple times based on cluster assignments. To perform IL, we employ the Dual-Memory Extreme Vector Machine (DM-EVM) [10], which is a variation of the original EVM proposed by [11]. Unlike the original EVM, the DM-EVM can perform IL on dynamic representations.

Following the extension of previous works, it is worth mentioning the operation of our approach in an actual open-world scenario. Indeed, the model performs a sequence of (Open-Set Recognition, Unsupervised Clustering, and Incremental Learning) that must be initiated whenever new classes are introduced. This model design is based on the assumption of a continuous data stream, where these steps are executed in cycles to discover and learn new classes continuously. While the process may not follow a specific "trigger," our model is structured for scenarios with regular monitoring for new incoming data, allowing it to adapt and learn from them spontaneously. This continuous cycle provides a dynamic capability that aligns with the changing nature of real-world scenarios.

By leveraging the models mentioned above, we assemble a framework that can perform OWR in videos without supervision after the initial training phase. Despite the high degree of difficulty posed by the problem, our model has shown promising performance. To our knowledge, ours is the first model to perform unsupervised Open-World HAR. We provide a test protocol that evaluates the models at

different stages using Open-set, clustering, and IL measures. We perform parameter sensitivity studies for the main parameters of our method. We also present a visual analysis of misclassified videos, revealing interesting relationships between classes.

The remainder of this paper is organized as follows. Section 2 presents the related works. Section 3 presents the proposed methods in detail. Section 4 presents the experimental settings and the results. Section 5 presents the parameter sensitivity studies regarding the main parameters responsible for the performance of our model. Section 6 presents a visual analysis at the frame level of some classification errors. Finally, Sect. 7 presents our conclusions and future works.

2 Related works

This Section presents a literature review of the areas involved in Open-World Video Recognition. First, we would like to briefly review relevant works in video classification. Next, we present OSR, IL, and OWR reviews.

2.1 Video classification

Convolutional Neural Networks (CNNs) have successfully been used to solve several interesting problems related to image processing [12–14]. However, most of these models were devised for non-temporal data and did not present a similar performance in video classification [15].

Unlike image classification, video classification models must consider spatiotemporal relationships between objects and actors. Donahue et al. [16] approached this problem by adding recurrent layers on top of the CNN as a temporal dimension. However, [6] pointed out that this approach may fail at capturing small motions, require more data, and are more likely to overfit.

Another strategy for video classification was the Two-Stream networks [17]. These models provided the network with an additional pre-computed Optical Flow channel. This concept was widely employed in the video classification literature [15, 18, 19].

Later, [20] proposed the 3D CNN to learn spatial and temporal features from a sequence of frames. Since then, 3D CNNs have become the state-of-the-art for video classification [7, 8, 21–26].

Carreira and Zisserman [6] raised an essential issue that 3D CNNs have a much higher number of parameters (compared with regular CNNs); thus, requiring more data and computational effort to be trained. The authors introduced the Kinetics dataset and the Inflated 3D Convolutional Neural Network (I3D). The I3D has since been highly used in video classification tasks [7, 8] due to its ability to generalize well to other datasets.

Gao et al. [27] proposed the attention-based graph convolution-guided third-order hourglass network (AGTH-Net) for sports video classification. Jing et al. [28] proposed the VideoSSL model for semi-supervised video classification. In a related area, [29] presented the Ordered Temporal Alignment Module (OTAM) for few-shot learning in video classification.

In [30], the authors presented a method for video-based action recognition using an incremental learning approach that leverages a combination of network sharing and knowledge distillation. This approach was designed to prevent catastrophic forgetting, a common problem in incremental learning models. Their work aimed to handle new classes without needing the data from previous ones. The algorithm was evaluated on multiple standard datasets, including UCF101, HMDB51, and Kinetics-400. However, our approach diverges significantly from this methodology. Primarily, our proposed approach focuses on an unsupervised paradigm for Open World Recognition, which is a much broader task than incremental learning. This choice eliminates the requirement for labeled data after the initial training, which is usually needed in incremental learning settings. Furthermore, our work proposes a different strategy for introducing new classes. Instead of initializing new classifiers using previous ones, our model focuses on determining the number of clusters from new, unknown classes and incrementally learning these new classes in a subsequent step.

2.2 Open-set recognition

Unlike the literature about video classification under the closed-world assumption, few works have approached the video classification problem from the open-set perspective. In [31], for instance, the authors investigated domain adaptation strategies applied to images and video HAR. The domain adaptation task refers to transferring knowledge from a source distribution to a different target distribution. In this case, the target distribution contains previously unavailable classes in the source data, making it an open-set problem. The authors used features extracted from the I3D model trained on Kinetics for video classification and introduced the ATI model for performing domain adaptation and classification.

In [32], the researchers also used I3D features for novelty detection in HAR and proposed a voting-based system for detecting unknown classes. Similarly, in [33], the authors performed open-set driver activity recognition using the I3D as the feature extractor and softmax confidence for recognizing unknown samples. Although softmax was not designed for open-set recognition, the authors proposed Dropout Sampling Statistics and Uncertainty-based Selective Voting to produce robust probability distributions at the output layer.

On the other hand, [34] employed micro-Doppler signatures for recognizing human activities. These authors introduced the Open-GAN model, which automatically generates fake unknown input samples for training.

In [35], the authors performed open-set face recognition in videos using handcrafted features and a fuzzy Adaptive Resonance Theory Map (ARTMAP) neural network. Differently, [36] proposed a Class-Conditional Extreme Value Theory classifier for open-set unsupervised video domain adaptation. The authors extracted frame level features using 2D CNNs and averaged the features in the temporal dimension. Wang et al. [37] also tackled the open-set video domain adaptation problem. The authors proposed the Dual Metric Discriminator (DMD), which measures similarities between samples from the source and target domains. This method combines pre-trained classifiers with prototypical optimal transport. Also, [38] proposed the Deep Evidential Action Recognition (DEAR) model that can extend closed-world architectures to open-set recognition. The authors also proposed two modules, the Evidential Uncertainty Calibration (EUC) and the Contrastive Evidential Debiasing (CED), which mitigate over-confident predictions and static bias.

Despite achieving state-of-the-art results for the proposed tasks, a limitation common to all works cited above is the absence of steps related to open-world classification, such as detecting new classes and incremental learning. In addition, the cited works did not explore metric learning characteristics during the open set step, which may hinder the full implementation of open-world recognition. In this sense, our previous work [9] focused on the feature learning process and showed that cosine metric learning performed by a Triplet Network improved the performance on Open-Set HAR. However, we also did not address issues of detection of new classes and incremental learning, leaving open the proposition of these tasks in the context of the open world.

2.3 Incremental learning

IL has been introduced as a problem in the early stages of Artificial Neural Networks [39], and is still a relevant problem in recent literature [40, 41].

The ability to learn new classes without forgetting pre-existing ones is often addressed in IL models. Catastrophic Forgetting and Intransigence [42, 43] are concepts that address the stability versus the plasticity of a network's knowledge. Since there is usually a trade-off between alleviating Catastrophic Forgetting or Intransigence, many models in the current literature require the user to choose which one to prioritize. This choice is typically through the selection of hyperparameters.

Task-recency bias is another common problem in IL models. This problem refers to the propensity of models to classify data as one of the classes learned in the most recent incremental training sessions [40].

IL models were divided into three main categories according to their strategy to tackle the abovementioned challenges. According to [41], these categories are replay, regularization, and parameter isolation methods.

Replay methods use a small memory to store exemplars of previously learned classes. These exemplars are used in the incremental training stage along with the new classes. This strategy helps the network maintain past knowledge while learning new classes. Incremental Classifier and Representation Learning (iCaRL) [44] was the first method to use replay in Class Incremental Learning. Later, several works showed that replay could mitigate task-recency bias and alleviate Forgetting [45–49]. Since replay methods have an additional storage cost, some methods employ feature rehearsal [40]. This method stores only feature representations of exemplars instead of raw data. This is especially important in video classification because of the elevated storage cost compared to images. In [10], feature rehearsal was successfully used for IL in videos with a low memory cost.

Regularization methods control Forgetting by introducing new terms in the loss function that prevent significant weights from changing excessively. Elastic Weight Consolidation (EWC) [50] introduced this concept by slowing down the training of significant weights for previous tasks. Path Integral [51], Memory Aware Synapses (MAS) [52], and Riemannian Walk (RWalk) [43] estimated the importance of each network parameter to prevent Forgetting. Other regularization methods use knowledge distillation to transfer knowledge from an old model to a new model. Learning Without Forgetting (LWF) [53] used a modified distillation loss to maintain weights close to their original values. Recently, [54] proposed a knowledge distillation model for incremental semantic segmentation.

Parameter Isolation methods mitigate Forgetting by learning a different model for each task. The main drawback of this category is that it often requires the task-id to be provided during the inference phase. Piggyback [55] and Ternary Feature Masks (TFM) [56] computed task-specific masks over network weights and features. Some methods grow different network branches to accommodate new tasks. For instance, Progressive Neural Networks (PNN) [57] created a copy of the network at each new task. Other growing network structures were presented in Progress and Compress [58] and Expert Gate [59]. More recently, [60] presented the SpaceNet, which does not require the task-id

to be given and creates space for new classes using sparse adaptive training.

Only some works have approached HAR in the IL scenario. Most of the works performed HAR in outdated small and controlled datasets. Ma et al. [61] presented the Grow When Required network (GWR). GWR learned classes incrementally using features from videos extracted at the frame level. The experiments were performed on two simple datasets: KTH [62], which contains only six classes, and Weizmann [63], which contains ten classes. Reddy et al. [64] used a feature tree for HAR in KTH the IXMAS [65] dataset with eleven classes. Tang et al. [66] performed online learning HAR over video data streams. Other older works perform Incremental HAR at the frame level but did not use current measures, such as Forgetting and Intransigence [67–70].

2.4 Open-world recognition

OWR has received increasing attention in recent years [1, 71]. The applications include face recognition and person re-identification [72–74], robotics [75], object detection [4], semantic segmentation [76], multi-modal human-robot interaction [77], and zero-shot learning [78]. Some works approached OWR from a traditional image classification perspective. Openmix [79] mixed labeled and unlabeled data at the visual level to increase unsupervised clustering performance. Liu et al. [80] introduced the Open Long Tailed Recognition model (OLTR), which uses dynamic meta-embedding for OWR in images. Jafarzadeh et al. [81] discussed the Open-World reliability problem and proposed automatic reliability assessment policies.

The most similar works to ours regarding Open-World HAR were the Open Deep Network (ODN) [82] and later the Prototype-based Open Deep Network (P-ODN) [83]. The ODN introduced a model based on Emphasis Initialization and Allometry Training for classifying human actions in an open-world scenario. P-ODN introduced the prototype and radius modules that assist in the feature learning process. Those works differ from ours in several ways. Our task-oriented open-world evaluation protocol allows for a deeper performance analysis of each task. While [82, 83] only present the classification accuracy, we provide a more comprehensive array of classification measures that were shown to be essential in the IL literature [43]. Another difference is that the ODN and P-ODN used a human observer to label unknown classes, while our method automatically estimates the number of clusters and their label assignments. In this sense, our model approaches the problem of unsupervised open-world video recognition, while ODN approaches a supervised version of the problem. Thus, both models are not directly comparable.

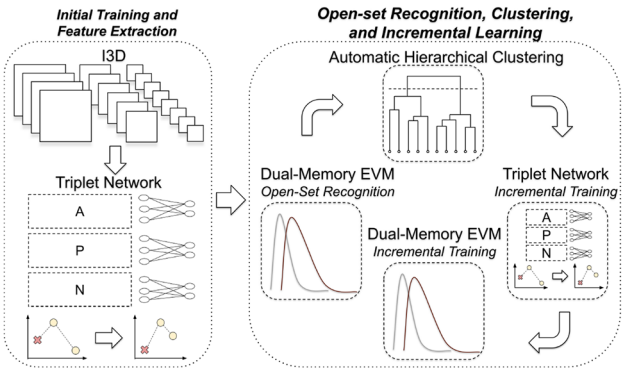

Fig. 2 Overview of the proposed method. The left panel shows the modules involved in the initial training phase and feature extraction. The right panel shows the modules involved in the OSR, automatic clustering, and IL phases

3 The proposed method

An overview of the proposed method is shown in Fig. 2. The I3D and the Triplet Network are trained in a supervised manner in the initial training phase. The models are also used in the following phases for extracting video features. When new classes are introduced, they go through the Dual-Memory EVM, which performs OSR. The rejected samples are automatically clustered using the Hierarchical Agglomerative model. The labels assigned by the clustering model and the previous EVs are used to form hard and semi-hard triplets, which fine-tune the Triplet Network. Finally, the updated feature representations and clustering assignments increment the Dual-Memory EVM. Each process stage is detailed as follows, starting with the dataset description. All the codes are made publicly available.[1]

The initial training process initializes the three main models of our method: (i) the I3D, (ii) the Triplet Network, and (iii) the DM-EVM. This is the only stage that uses the true labels of the dataset. All those stages are based on our previous works [9, 10]. However, we did not address a complete open-world perspective in those works, i.e., incremental learning with new class addition. Hence, this work is a complete proposal for HAR using videos in the context of open-world recognition.

The Inflated 3D Convolutional Neural Network (I3D) was proposed by [6] for the video classification task. It was introduced along with the Kinetics dataset, which allowed the I3D to achieve state-of-the-art performance on video classification. The I3D is trained using standard softmax cross-entropy loss. Our work used that model as the backbone for

[1] https://github.com/matheusgutoski/unsupervised-openworld-video-classification.

video feature extraction. However, our method is compatible with any other video classification network.

In the sequence, the triplet network is employed. The Triplet Network is a deep Metric Learning (ML) model first introduced by [84] for learning facial representations. ML aims to learn representations such that the distance between points has a semantic meaning. The Triplet Network receives its name for its architecture, which contains three inputs: Anchor, Positive, and Negative. Both the Anchor (a) and Positive (p) are points from the same class, while the Negative (n) is a point from a different class. Our model inputs the feature representations obtained from the I3D instead of raw video frames. This makes the training and fine-tuning process much faster. Given N (a, p, n) triplets, the Triplet Loss function L_Θ is:

$$L_\Theta = \sum_{i=1}^{N} \left[\Theta(g(\mathbf{x}_i^a), g(\mathbf{x}_i^p)) - \Theta(g(\mathbf{x}_i^a), g(\mathbf{x}_i^n)) + \alpha\right]_+, \quad (1)$$

in which i is the index, g is the Triplet Network, $g(\mathbf{x}^a)$, $g(\mathbf{x}^p)$, $g(\mathbf{x}^n)$ are the Anchor, Positive, and Negative output embeddings, α is the margin, and $_+$ indicates $L_\Theta \geq 0$. Θ represents the cosine distance between two feature representations. The cosine distance Θ between two vectors $\mathbf{d}$ and $\mathbf{e}$ is defined as:

$$\Theta(\mathbf{d}, \mathbf{e}) = 1 - \frac{\mathbf{d} \cdot \mathbf{e}}{\| \mathbf{d} \| \| \mathbf{e} \|}. \quad (2)$$

The Extreme Value Machine (EVM) [11] is an OSR model capable of performing IL. In recent work, we introduced a variation of the EVM called Dual-Memory EVM (DM-EVM) [10]. Unlike the original EVM, the DM-EVM is capable of IL with dynamical feature representations. It performs this task by storing an additional representation of each Extreme Vector compared to the original EVM. This representation corresponds to the fixed I3D features, later included in the triplet mining pool to fine-tune the Triplet Network and update the Extreme Vectors' dynamical features. For a more in-depth explanation of the DM-EVM, refer to [10].

Subsequently, the OSR process begins with the arrival of a new task. First, the I3D and Triplet representations were obtained by forwarding the new videos through the trained models. The DM-EVM used the representations to determine whether each video belongs to known or unknown classes. The rejected videos are sent to the clustering phase.

The automatic clustering method used in this work is based on Ward's algorithm for hierarchical agglomerative clustering [85]. Ward's method merges clusters by minimizing variance. Consider a hierarchical clustering with an observation set $\mathbb{I}$ with a predefined dissimilarity measure (e.g., Euclidean distance measure). Set each observation $i, j, k \in \mathbb{I}$ as a singleton cluster. Agglomerate the closest (i.e., least dissimilar) pair of clusters, removing agglomerated clusters. Redefine the inter-cluster dissimilarities concerning the newly created cluster. If n is the cardinality of observation set $\mathbb{I}$, this agglomerative hierarchical clustering algorithm completes in $n - 1$ agglomerative steps. Details of practical aspects of our implementation are presented in Sect. 4.4.

Since our Triplet Network operates using the cosine distance, we scaled our feature representations to vector length using the L2 norm during the clustering phase. The label assignments obtained by clustering are used to form new triplets and fine-tune the Triplet Network for a single epoch. The new task data and the previous EVs stored in the DM-EVM form the triplet mining pool.

Finally, incrementing the DM-EVM requires performing two operations: updating the current EVs and computing the new class EVs. For the first operation, the existing EVs were updated using the representations obtained from the Triplet Network. This step requires recomputing the dynamical feature representations and the Weibull parameters. The dynamical features were recomputed with a single pass through the Triplet Network. The updated parameters were computed by fitting the Weibull distribution. Model reduction is unnecessary in this step since the EVs have already been defined, thus making the process much faster. We learned new EVs for the second operation using the new data and their labels obtained by clustering. This step included model reduction.

4 Experiments and results

4.1 Dataset and task generation

We employed the UCF-101 dataset [86], which has been widely used in the previous literature [6, 8]. The UCF-101 dataset contains over 27 h of footage of human actions such as golfing, archery, and drumming. The dataset contains 13,320 clips of five main categories: body motions, sports, playing musical instruments, human-object interactions, and human-human interactions. The dataset contains cluttered background and camera motion, which introduces more challenges. The dataset also contains groups of similar videos within each class. These groups are snippets extracted from a common long-duration video.

In this work, the dataset was split in such a way as to ensure that all videos from the same group remained together in the train or test split. We used 70% of the dataset for training and 30% for testing. The tasks were generated by randomly selecting classes for each task. The number and size of tasks vary according to the experimental setting, detailed as follows.

Table 1 Experimental settings proposed for Open-World HAR. Small settings received the prefix "S", while the large settings received the prefix "L"

Setting	# Initial classes	# Classes/task	# Tasks	Total # Classes
S1	3	3	5	15
S2	4	4	5	20
S3	5	5	5	25
S4	6	6	5	30
S5	7	7	5	35
S6	8	8	5	40
S7	9	9	5	45
S8	10	10	5	50
L1	6	5	20	101
L2	11	10	10	101
L3	21	20	5	101

4.2 Experimental settings

We propose eleven experimental settings for evaluating Open-World Video Recognition. The settings vary the size and number of tasks available for learning. Table 1 shows the experimental settings and their parameters. The table is organized in a crescent order in the number of classes, which also means lowest to highest problem complexity.

The first column contains the reference code for each experiment. We classify the experimental settings into small (S) and large (L) experiments. Small experiments use only a fraction of the available classes and are restricted to five tasks. Extensive experiments use all 101 classes from the UCF-101 dataset. This experimental setup allows analyzing the results in a scenario of increasing complexity. Each experimental setting is run five times with different random seeds to shuffle the order in which classes appear in each task. We report the average results of the five runs in Sect. 4.7.

4.3 Initial training and feature extraction

The initial training process initializes the three main models of our method: the I3D, the Triplet Network, and the DM-EVM. This is the only stage that uses the true labels of the dataset.

4.3.1 Inflated 3D convolutional neural network

The I3D was trained using the initial classes with the softmax cross-entropy loss. We followed the training procedure proposed by [6]. The I3D received 64 consecutive frames with 224×224 pixels of size as input for each video. These frames were selected randomly in both spatial and temporal dimensions. We also applied a 50% chance of random horizontal flip to the frames. The batch size was set to 6. We used the Stochastic Gradient Descent (SGD) with a learning rate of 0.1, weight decay of 10^{-5}, and Nesterov momentum of 0.9. The network was trained for ten epochs or until stagnation. We used a window of 250 frames with 224×224 pixels of size centered in spatial and temporal dimensions during the test phase.

4.3.2 Metric learning with triplet networks

We defined the architecture of our Triplet Network with three fully connected layers containing 1024, 512, and 256 neurons. The Glorot uniform algorithm [87] was used to initialize the weights. Moreover, we employed a triplet mining strategy [84] to ensure that only useful triplets are used during the training process, i.e., triplets that yield $L_\Theta > 0$. In this phase, we trained the Triplet Network for ten epochs. The Triplet Network used the output features of the I3D as input. We set the margin parameter α to 0.2. A parameter sensitivity study can be found in Sect. 5. The learning rate was set to 0.001 and the batch size to 128.

4.3.3 The dual-memory extreme value machine

In our experiments, the tail size τ of the Weibull distribution was set to 0.2, i.e., 20% of the training set size. The classification threshold δ was set to 0.5. Section 5 presents parameter sensitivity studies for both parameters. According to [10], the cover threshold ς was set to 0.99.

The trained I3D and Triplet Network were later used for extracting features from new videos. This was done by forwarding the videos through the I3D and capturing the representations after the last pooling layer. The representations were then forwarded through a branch of the Triplet Network and captured at the last layer to obtain the metric representation.

4.4 Clustering

Our strategy for selecting the appropriate value for the number of clusters k contains two main steps. The first step is to select k_c candidate partitions. The partitions are defined by the k_c largest merge gaps in the dendrogram. The second step was to compute each candidate partition's silhouette coefficient [88] and pick the partition with the highest score.

In selecting the k_c value, we conducted a series of preliminary experiments with various k_c values to examine their potential influence on the overall performance of our method. These tests revealed that the variation in k_c had minimal effect on the outcomes. Regardless of the specific k_c value chosen, the performance of our clustering approach remained relatively stable. Based on this observation, we decided to fix k_c at 5 for all our experiments. This value

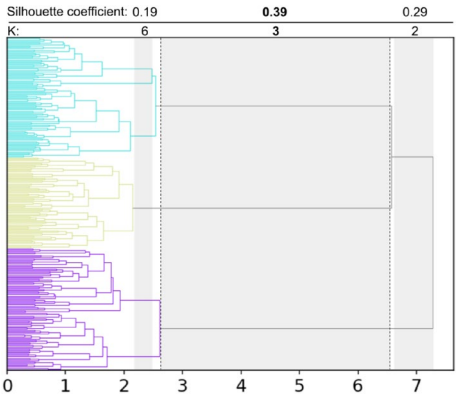

Fig. 3 Method for automatically selecting the value of k. First, candidate partitions were selected based on the most significant dendrogram merge gaps, as the gray areas show. Then, the partition with the largest silhouette value was chosen for defining k

was chosen because it balances computational efficiency and the range of candidate partitions considered, thus achieving a practical compromise between performance and computational demand. This strategy, coupled with the fact that the silhouette coefficient is calculated only for the selected candidate partitions and not the entire data set, contributes to the computational feasibility of our approach.

Figure 3 shows an example of the automatic selection of k using the method described above. The gray regions display the candidate partitions by selecting the largest merge gaps. For visualization purposes, we set $k_c = 3$ in this example. The number of clusters and silhouette coefficient of each partition are displayed above the gray areas. The partition with $k = 3$ was selected since it presented the largest silhouette value.

4.5 Incremental learning of the triplet network and the DM-EVM

The label assignments obtained by clustering are used to form new triplets and fine-tune the Triplet Network for a single epoch. The new task data and the previous EVs stored in the DM-EVM formed the triplet mining pool.

Incrementing the DM-EVM requires performing two operations: updating the current EVs and computing the new class EVs. For the first operation, the existing EVs were updated using the representations obtained from the Triplet Network. This step requires recomputing the dynamical feature representations and the Weibull parameters. The dynamical features were recomputed with a single pass through the Triplet Network. The updated parameters were computed by fitting the Weibull distribution. Model reduction is unnecessary at this step since the EVs have already been defined, thus making the process much faster. We learned new EVs for the second operation using the new data and their labels obtained by clustering. This step included model reduction.

4.6 Evaluation protocol and measures

We reported the averaged results of five initializations with different random seeds since they produce a different sequence of classes in each task. The initialization averaging was omitted in the following equations for simplicity.

The measure chosen for the OSR phase was the Youdens Index [89]. It accesses the model's capability to distinguish between known and unknown data. The Youdens Index combines the recall R and specificity S measures, as follows:

$$R = \frac{\text{TP}}{\text{TP} + \text{FN}}, \; S = \frac{\text{TN}}{\text{TN} + \text{FP}}. \tag{3}$$

To compute R and S, known classes receive the label "1", and unknown classes receive "0". The Youdens Index J is defined as $J = R + S - 1$ and assumes a value within the range $[-1, 1]$, where "-1" means a classifier that incorrectly classifies all samples, "0" means an uninformative classifier, and 1 means a perfect classifier.

For the main performance measure, we selected the Normalized Mutual Information (NMI) score [90]. This entropy-based measure is independent of label matchings and is often used for clustering [91]. The NMI for task t after learning up to task m is:

$$\text{NMI}_m^t(\mathbf{y}, \hat{\mathbf{y}}) = \frac{I(\mathbf{y}, \hat{\mathbf{y}})}{\sqrt{H(\mathbf{y})H(\hat{\mathbf{y}})}}, \tag{4}$$

in which H is the entropy, $\mathbf{y}$ are the true labels, and $\hat{\mathbf{y}}$ are the predicted labels. $I(\mathbf{y}, \hat{\mathbf{y}})$ represents mutual information between $\mathbf{y}$ and $\hat{\mathbf{y}}$:

$$I(\mathbf{y}, \hat{\mathbf{y}}) = \sum_i \sum_j \frac{\left|y_i \bigcap \hat{y}_j\right|}{N} \log\left(\frac{N\left|y_i \hat{y}_j\right|}{\left|y_i\right|\left|\hat{y}_j\right|}\right), \tag{5}$$

in which N is the total number of samples, and $\left|\cdot\right|$ is the cardinality. The entropy H is:

$$H(\mathbf{y}) = -\sum_i \frac{\left|y_i\right|}{N} \log\left(\frac{\left|y_i\right|}{N}\right). \tag{6}$$

For the IL phase, we also evaluated the Forgetting and the Inter-Task Intransigence (ITI) [10]. We compute Forgetting at the task level. According to [43], Forgetting can be defined as the maximum performance historically achieved at a task minus the performance achieved on the same task after incrementally learning all tasks. Since we use the NMI

as the main performance measure, Forgetting for task t after incremental training up to task m can be defined as:

$$f_m^t = \max_{l \in \{0,\ldots,m-1\}} \text{NMI}_l^t - \text{NMI}_m^t, \quad \forall t < m. \tag{7}$$

The Mean Forgetting is obtained by averaging the Forgetting of each task (after training up to the final task m). The mean ITI is computed by subtracting the NMI of the current task t from the NMI of the previous task $t-1$, up to the final task m. Then, all values are averaged as follows:

$$\text{ITI}_m^t = \frac{1}{m-1} \sum_{t=1}^{m} \text{NMI}_m^t - \text{NMI}_m^{t-1}. \tag{8}$$

Positive mean ITI reflects an average gain in performance as new tasks are learned. Negative mean ITI indicates that performance tends to degrade after learning each task.

4.7 Results

This Section is organized in such a way as to present and discuss the results of each phase separately. We omit the initial training measures since this phase is easily solved by the triplet network coupled with the I3D backbone, achieving near 100% accuracy in all experimental settings. Moreover, the initial training follows a traditional supervised training approach widely found in the literature. Instead, we focus on the critical aspects of OWR: OSR, clustering, IL, and, finally, the entire test set results. We compare other clustering techniques and indicate previous works that compare with other methods in the Open-set and IL phases. However, since ours is the first model to perform unsupervised Open-World HAR, it is not possible to compare the final classification results.

4.7.1 Phase 2: open-set recognition

OSR aims at performing classification while also rejecting samples from unknown classes. A comparison with other methods can be found in our previous work [9]. In the context of OWR, correctly differentiating between known and unknown is the most critical aspect of this phase. The reason is that unknown samples are further processed in the clustering and IL steps.

The Youdens Index measures the ability to differentiate between known and unknown. Table 2 (second column) shows the Youdens index averaged between tasks on all experimental settings. The test set includes the new and previously seen classes on each task. New classes are assigned the unknown label, and known classes are assigned the known label. The results show a decreasing performance trend as the experimental settings get more complex.

Table 2 Mean and standard deviation obtained in the eleven experimental settings grouped by phases. Results of phases 2, 3, and 4 are averaged between tasks, while the full test set results were averaged between the five random seed initializations

Setting	Phase 2	Phase 3			Phase 4			Full test set		
	Youdens index	k	$\hat{k}$	NMI	Forgetting	ITI	NMI	k	$\hat{k}$	NMI
S1	0.791 ± 0.085	3	3.05 ± 0.191	0.979 ± 0.027	0.040 ± 0.032	0.016 ± 0.050	0.932 ± 0.032	15	14.8 ± 0.400	0.945 ± 0.024
S2	0.816 ± 0.065	4	4.00 ± 0.163	0.972 ± 0.022	0.021 ± 0.014	− 0.014 ± 0.049	0.930 ± 0.029	20	19.8 ± 0.400	0.941 ± 0.027
S3	0.779 ± 0.081	5	5.05 ± 0.191	0.973 ± 0.016	0.021 ± 0.013	0.005 ± 0.043	0.932 ± 0.020	25	25.0 ± 0.632	0.937 ± 0.008
S4	0.781 ± 0.103	6	6.00 ± 0.163	0.969 ± 0.011	0.019 ± 0.014	− 0.009 ± 0.016	0.929 ± 0.015	30	29.8 ± 0.400	0.932 ± 0.010
S5	0.773 ± 0.093	7	6.95 ± 0.191	0.962 ± 0.013	0.015 ± 0.012	− 0.010 ± 0.016	0.923 ± 0.022	35	34.6 ± 1.200	0.926 ± 0.005
S6	0.754 ± 0.115	8	7.85 ± 0.100	0.963 ± 0.014	0.019 ± 0.012	− 0.004 ± 0.027	0.928 ± 0.021	40	39.4 ± 0.800	0.925 ± 0.018
S7	0.738 ± 0.102	9	8.55 ± 0.443	0.952 ± 0.022	0.018 ± 0.013	− 0.001 ± 0.040	0.924 ± 0.030	45	43.2 ± 1.470	0.923 ± 0.012
S8	0.729 ± 0.098	10	9.65 ± 0.191	0.963 ± 0.003	0.015 ± 0.010	− 0.002 ± 0.022	0.923 ± 0.016	50	48.6 ± 2.059	0.919 ± 0.014
L1	0.615 ± 0.106	5	4.92 ± 0.213	0.971 ± 0.016	0.043 ± 0.024	0.001 ± 0.044	0.882 ± 0.028	101	98.4 ± 1.744	0.910 ± 0.005
L2	0.640 ± 0.104	10	9.66 ± 0.412	0.956 ± 0.007	0.022 ± 0.013	− 0.003 ± 0.014	0.898 ± 0.013	101	98.6 ± 2.728	0.902 ± 0.007
L3	0.636 ± 0.081	20	16.9 ± 0.931	0.930 ± 0.015	0.013 ± 0.009	− 0.003 ± 0.010	0.897 ± 0.034	101	88.6 ± 5.004	0.890 ± 0.012

Table 3 Comparison among three clustering methods in four experimental settings

Setting	Method	k	$\hat{k}$	NMI
S1	Hierarchical	3	3.050 ± 0.191	0.979 ± 0.027
	FINCH	3	17.90 ± 11.87	0.845 ± 0.114
	X-Means	3	3.950 ± 0.700	0.917 ± 0.04
S4	Hierarchical	6	6.000 ± 0.163	0.969 ± 0.011
	FINCH	6	76.65 ± 27.03	0.737 ± 0.079
	X-Means	6	6.700 ± 0.663	0.944 ± 0.006
L2	Hierarchical	10	9.660 ± 0.412	0.956 ± 0.007
	FINCH	10	146.6 ± 52.25	0.728 ± 0.095
	X-Means	10	10.778 ± 0.524	0.946 ± 0.009
L3	Hierarchical	20	16.90 ± 0.931	0.930 ± 0.015
	FINCH	20	424.1 ± 17.27	0.658 ± 0.003
	X-Means	20	20.90 ± 0.739	0.937 ± 0.006

However, even in the large experimental settings, the model has shown an average Youdens index above 0.6.

4.7.2 Phase 3: clustering

Clustering is performed on the rejected data from the previous OSR phase using the hierarchical clustering algorithm. Since our framework operates under an unsupervised scenario, the clustering algorithm must automatically determine the number of clusters k. We compared our automatic hierarchical clustering strategy to two other automatic clustering methods.

The Efficient Parameter-free Clustering Using First Neighbor Relations (FINCH) [92] is also an agglomerative clustering technique that performs automatic clustering without hyperparameters. However, FINCH proposes a set of possible clustering partitions instead of a single solution. Hence, we selected the partition with the largest silhouette coefficient.

The X-Means [93] algorithm is an extension of K-Means that automatically estimates the number of clusters. We set the minimum number of clusters to 2 and the maximum number of clusters to 101. The initialization was performed using the K-Means ++ algorithm [94], and the tolerance parameter was set to 0.25. The Bayesian Information Criterion (BIC) was used to estimate the number of clusters.

We performed comparison experiments on two small and large experimental settings: S1, S4, L2, and L3. We replaced the hierarchical clustering module with the FINCH and X-Means models and performed the entire OWR experimental scenarios. The results of the clustering phase are shown in Table 3. The ground truth number of clusters is shown as k, while the predicted number of clusters is shown as $\hat{k}$.

The FINCH algorithm overestimated the number of clusters in all four experimental settings. FINCH clustered data based on similar video groups in the UCF-101 dataset rather than their classes. Video groups are snippets that were cut from the same large video. These results have shown that FINCH tended to form large numbers of small clusters that were not meaningful, particularly from the class perspective of this work.

The X-Means algorithm had a much better estimation of k when compared to FINCH. The model had a similar performance to our automatic hierarchical clustering strategy. The hierarchical model was superior in the S1, S4, and L2 settings regarding k estimation and NMI. The X-Means algorithm was superior in the L3 setting. Overall, the X-Means model has shown promising performance. However, it requires the user to set a range for the possible number of clusters which may be prohibitive in some applications.

From this point, we discuss the experimental results obtained with the automatic hierarchical clustering strategy. The actual number of clusters k and predicted number of clusters $\hat{k}$ per task are shown in Table 3, in the third and fourth columns, respectively. The predicted values were averaged over all tasks since they have a constant number of clusters.

The model has accurately estimated the value of k in most of the experiments. In the Small experimental settings, the average predicted values for k were offset by less than one. The L3 setting had the most significant gap between k and $\hat{y}$, with an average of 16.9 predicted for a ground truth of 20. This suggests that estimating k becomes more difficult as the number of clusters increases. Hence, open-world learning with fewer classes at each iteration may improve automatic clustering performance in a real-world application.

We also evaluate the clustering NMI, as shown in the fifth column of Table 3. The model achieved high values of NMI in all experimental settings. This suggests that the features learned by the triplet network are easily separable by the hierarchical clustering algorithm. Once more, The L3 setting presented the lowest NMI. This is caused by the larger gap between the true and predicted k.

4.7.3 Phase 4: incremental learning

IL models are often evaluated using three main measures: Forgetting, intransigence, and a classification measure such as the NMI. For intransigence, we adopt the Inter Task Intransigence (ITI) [10]. The results were averaged between tasks and are presented in Table 2.

In this work, the Forgetting of a model for a given task is the difference between the historical maximum NMI and the final NMI. In most small (S) experimental settings, our model presented a Forgetting below 0.021, except for S1, which can be considered an outlier. L1 gave a slightly higher average Forgetting when compared to the other large (L) settings. The cause may be the more significant number of

tasks in this experiment (20 tasks) than L2 and L3 (10 and 5 tasks). Despite having relatively low Forgetting values, this result suggests that eliminating Forgetting in large sequences of tasks is still a challenge in OWR, especially in unsupervised scenarios.

Intransigence is often portrayed in the literature as the opposite of Forgetting [43]. It measures the plasticity of a model when learning new classes. Models with intransigence problems cannot learn new classes effectively because they resist changes to avoid forgetting. The Inter-Task Intransigence (ITI) measures the NMI gain (positive values) or decay (negative values) of one task concerning the previous task. The average ITI, shown in Table 2, is obtained by averaging this change among all tasks. In the experiments, our model has maintained an ITI centered near zero. This suggests that our model's learning ability does not decay as new classes are learned.

The NMI measured in the IL phase is computed by averaging the NMI obtained in each task. The results show a slightly decreasing performance trend as the experimental settings become more complex. However, the model shows great promise, with an average NMI of over 0.92 in the small settings and above 0.88 in the large settings. A comparison of IL with other state-of-the-art methods can be found in our previous work [10].

4.7.4 Full test set

After all the tasks have been learned, the full test set results are obtained by evaluating the model using the complete test set. Similar to the clustering phase, we evaluate $\hat{y}$ and the NMI of the full test set. This time, the values are averaged over the five random seed initializations. The results are shown in Table 2.

Our model accurately estimates the number of clusters k in most experimental settings. The highest gap between true and predicted k occurred in the L3 setting, where classes are introduced in increments of 20. This setting also presented a slightly lower NMI when compared to other settings. Once more, this result suggests that our model obtains a better performance in scenarios where fewer classes are introduced in each task.

5 Parameter sensitivity studies

We investigate three main parameters of our method and their impact on performance: EVM classification threshold, EVM tail size, and the Triplet Network margin. Each parameter was evaluated on the L2 experimental setting (first fold) using a predefined range of values. We considered evaluation measures from phases 2 and 4 (OSR and IL). The NMI and $\hat{y}$ from the last iteration were also presented for a more general evaluation.

5.1 EVM classification threshold δ

The classification threshold δ controls the rejection aspect of the EVM in the OSR phase. Larger values of δ lead to a more restrictive EVM model when accepting data belonging to a known class, i.e., it causes more rejection. Relatively large values for δ may cause the rejection of known class members. On the contrary, reduced values may cause new classes to be confused with existing classes.

Figure 4 shows the results of six different measures as a function of δ. The values shown in the graphs are an average between all tasks. The standard deviations are displayed as error bars. The bottom center and bottom right graphs are the results obtained on the last iteration of the experiments using the entire test set. Hence, they are not task averaged.

The Youdens index is shown in the top left corner of Fig. 4. This measure is directly affected by the classification threshold δ since it measures the performance of distinguishing between known and unknown classes. As one can observe, the Youdens index is lower at the extremes and larger at the middle range of δ values. This suggests that 0.5 provides a good balance for rejection.

The following two measures, mean Forgetting and mean ITI, are not heavily affected by δ. The mean NMI among all tasks is slightly affected but stagnates after the 0.5 mark.

The predicted number of clusters $\hat{y}$ in the final test set (bottom center) was compromised when using small values of δ. At low values of δ, many new classes are wrongly classified as existing classes. Thus, some new classes merge with pre-existing classes and disappear from the training set. This suggests that δ needs to be sufficiently large to allow new classes to be discovered and formed in the clustering and IL steps.

Finally, the NMI in the final test set is shown in the bottom right of Fig. 4. As one can observe, δ does not significantly impact this measure as long as it is not extremely small. All values above 0.1 have shown similar performance. After evaluating the results shown in this Section, we have chosen a δ of 0.5 for the remaining experiments. This also includes parameter sensitivity studies for the parameters τ and α.

5.2 EVM tail size τ

The EVM tail size τ controls the fraction of the training data used to fit the Weibull distribution. Figure 5 shows the results obtained using each measure for four different values of τ, ranging from 20% to 50%. A negligible difference in performance was observed in all cases, suggesting that τ does not cause a significant impact on the

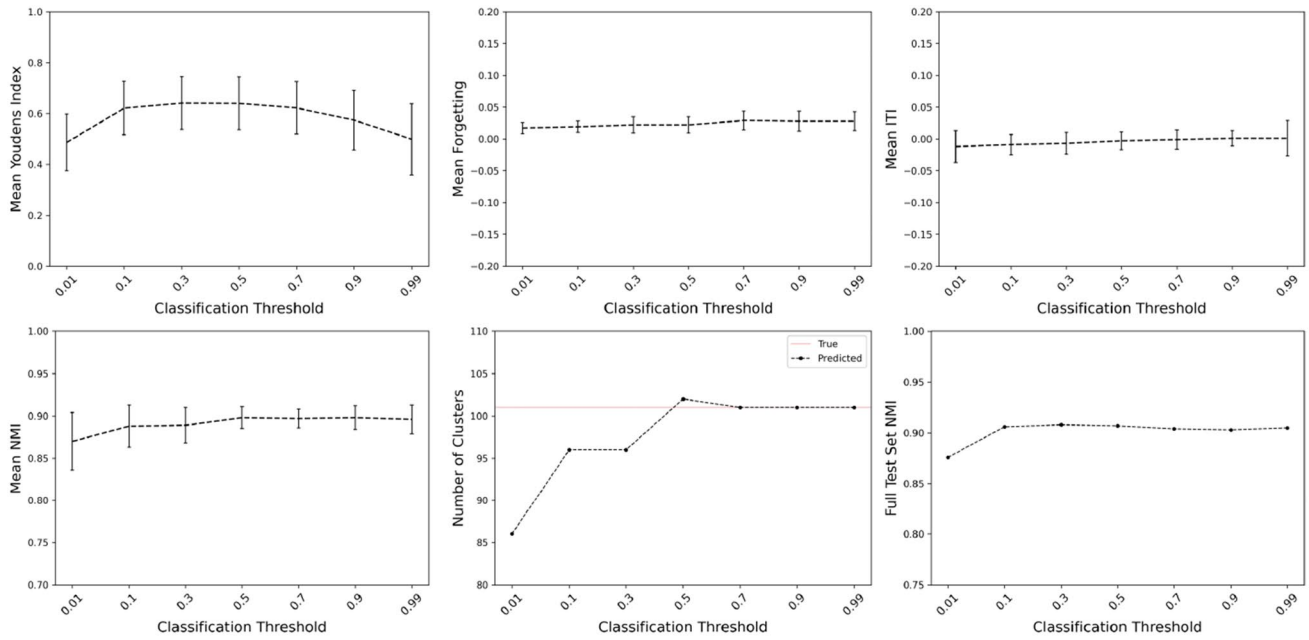

Fig. 4 Performance on the L2 setting using different values of δ

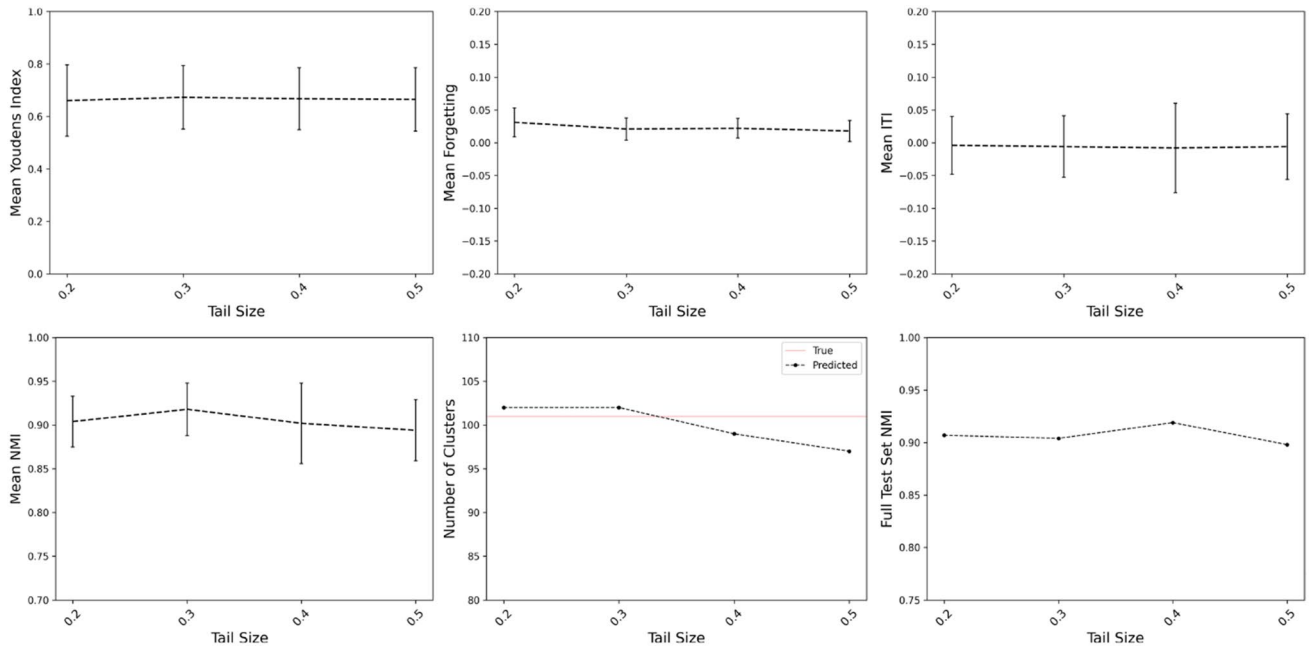

Fig. 5 Performance on the L2 setting using different values of τ

performance of our model. However, fitting the Weibull distribution using larger values of τ is more expensive in time and computational resources. Hence, we set $\tau = 0.2$ in the remaining experiments.

5.3 Triplet network margin α

The Triplet Network margin α impacts both the Triplet Loss function and the semi-hard triplet mining, as seen in Sect. 4.3.2. This parameter is related to the quality of the

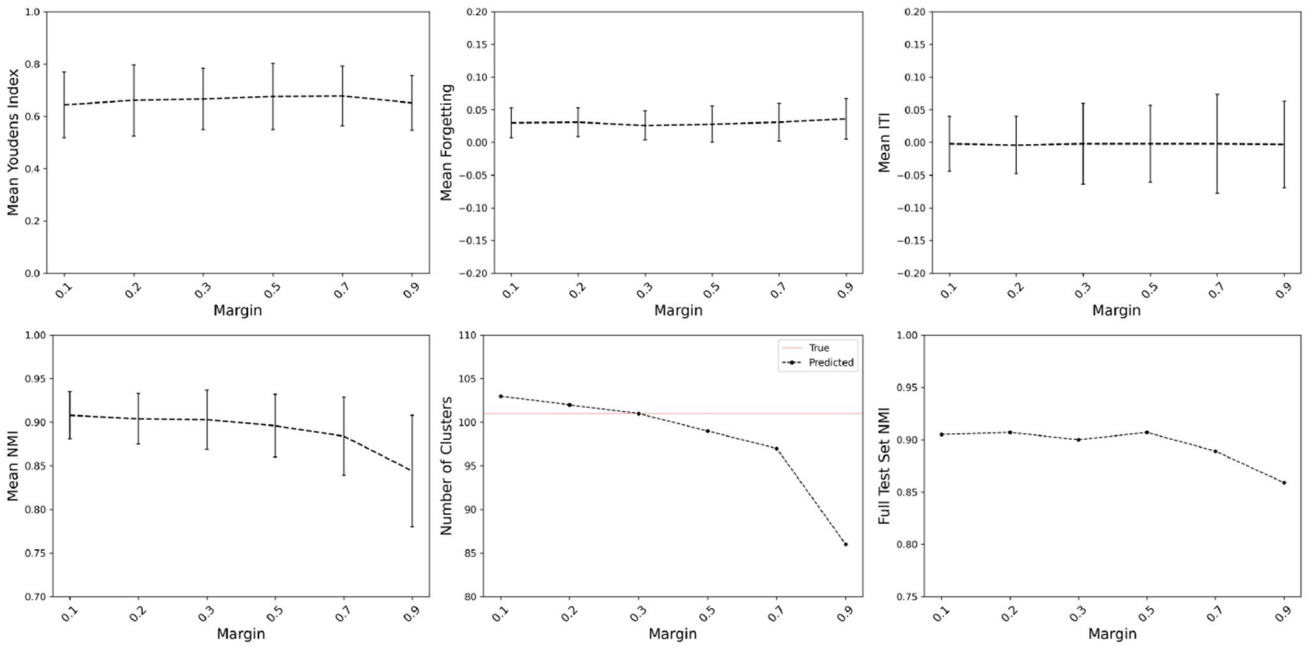

Fig. 6 Performance on the L2 setting using different values of α

features learned by the Triplet Network. Figure 6 shows the results obtained in each measure using different values of α. The Youdens index, Forgetting, and ITI were not significantly affected by the changes in α. However, a larger standard deviation can be observed for larger values of α in ITI. The same can be observed regarding NMI and a decrease in mean performance as α increases.

The estimated $\hat{y}$ and the NMI on the entire test set also suffered a performance decay as α became larger. This suggests that the metric learning process produced more informative features using lower values of α. Thus, we set the parameter α to 0.2 in the remaining experiments.

6 Visual analysis

This Section shows a visual analysis of the two first tasks of a fold of the S1 setting. We chose this setting because it has few classes. Therefore, a visual analysis is straightforward and easier than other settings with many classes. Such an analysis can lead to a better understanding of the behavior of our OWR model. For this purpose, videos were individually tracked to uncover possible reasons for misclassifications.

We employ t-SNE [95] to reduce the dimensionality of the triplet network features and allow the observation of their structure in two dimensions. Even if the dimensionality reduction causes a loss of information, we found the structure of the representations enlightening in many cases.

The first fold of the S1 setting was initialized by randomly choosing three initial classes. In this case, "Apply Lipstick," "Boxing Speed Bag," and "Cricket Shot" were selected. The model underwent the initial training phase trained the I3D and the triplet network in a standard supervised learning approach.

In phase 2, "Mopping Floor," "Shaving Beard," and "Skiing" were introduced in the new task. The OSR model predicted whether each video in the new task belongs to a known or a novel class. This binary classification was performed first in the training set where all data is unknown. Since we cannot determine the Youdens index of a set containing only unknown classes, the model was evaluated on a separate test set. The test set contains videos from all classes presented until this point. This includes the initial three classes and the three unknown classes.

The classification obtained by the Dual-Memory EVM is shown in the top left of Fig. 7. A single point represents each video in the test set. The shapes represent the true label of each video, while the predictions are represented as colors. It can be observed that each class forms well-separated clusters. The unknown videos were mostly correctly classified as unknown. False rejections occurred in few videos of the "Boxing Speed Bag" and "Apply Lipstick" classes.

The most interesting false prediction occurred near the "Apply Lipstick" cluster in the top center of the box. This cluster is slightly overlapped with a cluster formed by the unknown class "Shaving Beard." This caused false rejections (in "A") and one false recognition (in "B"). By examining the incorrectly classified videos, it can be observed they are visually similar. In both cases, a third person is performing the action on the target subject, which may have confused.

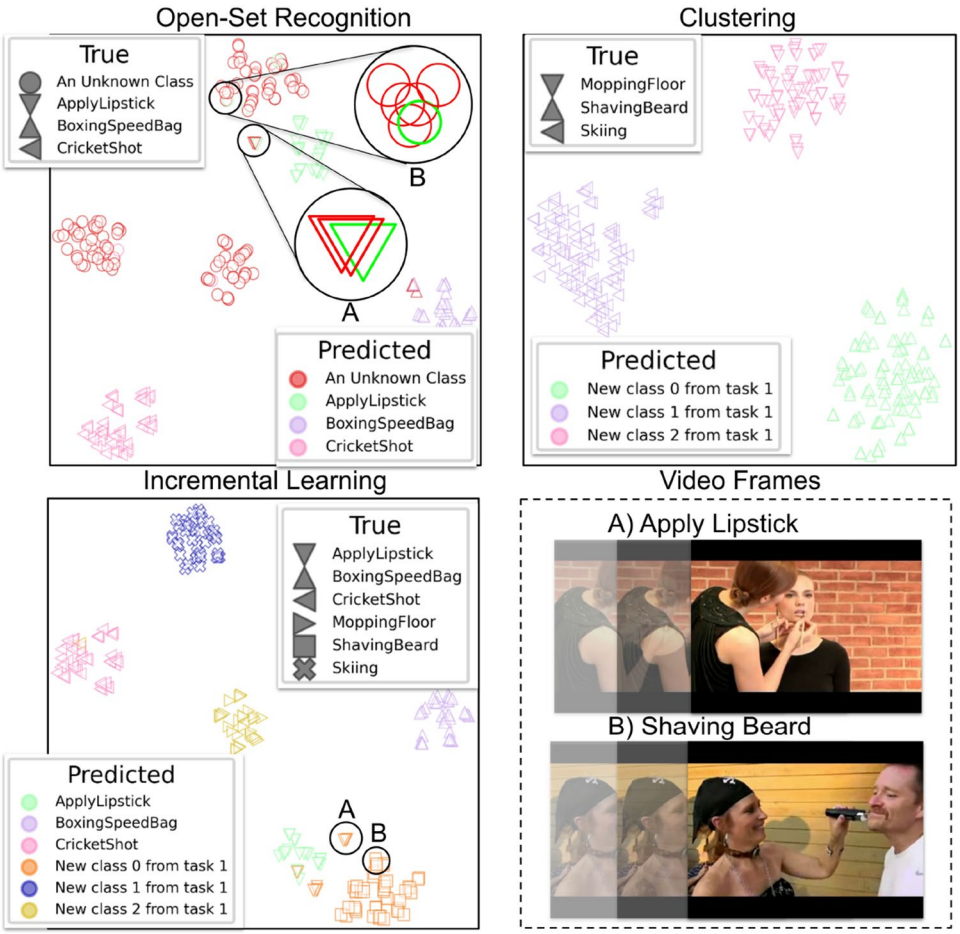

Fig. 7 Feature space obtained with t-SNE in three different phases of unsupervised OWR. Figure best viewed in color

The rejected data from the training set was automatically clustered in the next phase. The top right of Fig. 7 shows the clustering results. Our model accurately predicted the number of clusters and assignments in this case. Since labels are unavailable during the training phase, we name each cluster according to the number and task they were discovered.

After discovering the three new classes, the IL phase was conducted. Our model learns from the labels assigned by the clustering algorithm. From this point onward, the three new classes become known. Figure 7 (lower left) shows the final classification results on the test set. The test set includes all known classes learned up to this point. Once more, some points of "Apply Lipstick" and "Shaving Beard" appear overlapped and cause a classification error in "A".

For the second task of the S1 experiment, we skip the open set and clustering phases to show the final results after IL. Figure 8 shows the feature space after introducing three more classes: "Floor Gymnastics," "Surfing," and "YoYo."

With the introduction of the new task, two new classification errors come to our attention. It can be observed that the "Skiing" and "Surfing" clusters are relatively close to one another in the feature space. This causes some overlapping between those two classes. In "C", a "Skiing" video was misclassified as surfing. The cause for this error becomes evident upon a visual inspection of the particular video. The background and motion are very similar to a "Surfing" video.

Another interesting classification error occurred in "D". In this case, a "Skiing" video was misclassified as "YoYo" despite being two different classes in terms of appearance and motion. By inspecting the frames, it was observed that this particular video contains a television logo that lines perfectly with the ski pole. This creates the illusion of a "YoYo" in some frames. The "YoYo" appearance has prevailed over the "Skiing" background and motion in the feature space. This suggests that the presence of a "YoYo" in our feature extractor may be a stronger class indicator than objects and motions used in "Skiing."

7 Conclusions and future works

This work presented a framework for performing Unsupervised OWR in Human Action videos. We divided the problem into four phases: initial training, OSR, clustering, and IL.

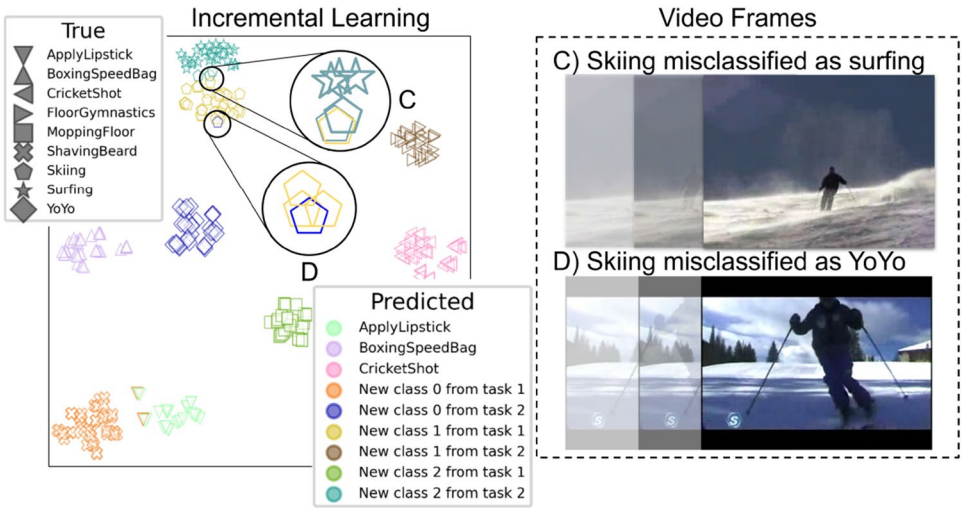

Fig. 8 Feature space obtained with t-SNE in the IL phase. Figure best viewed in color

The initial training phase was conducted using a supervised learning approach, while the remaining phases received no label information. Despite the lack of labels, our model closely estimated the number of clusters k and provided precise label assignments based on the mean NMI.

We observed a negative performance trend regarding the experimental settings proposed in this work, as the settings included more classes in each task. This suggests that our method performs best by introducing a small number of classes at each increment rather than a large number of classes.

Our visual inspection of some experimental results has revealed interesting aspects of the feature learning process. First, classes formed well-defined and separated clusters in most cases, suggesting the I3D coupled with the Triplet Network output easily separable feature representations. We also observed instances in which clusters were partially overlapped. In those cases, we noted a substantial visual similarity between them. It is possible that obtaining more training data or introducing mechanisms in the feature learning models could mitigate this problem. However, the solution must consider the limited resources of Open-World models, which pose an additional challenge for future works.

OWR is an important step toward effective models for solving real-world problems. It is imperative in the unsupervised setting because of the large amount of unlabeled daily data. Hence, we encourage future works that point in that direction.

Acknowledgements Author M. Gutoski would like to thank CNPq for the scholarship number 141983/2018-3. Author H. S. Lopes would like to thank to CNPq for the research grant 311785/2019- 0, and Fundação Araucária for grant PRONEX 042/2018. Author A. E. Lazzaretti would like to thank to CNPq for the research grant 306569/2022-1. All authors would like to thank NVIDIA Corp. for the donation of the Titan-Xp GPUs used in the experiments.

Data availability The UCF-101 dataset [86] that support the findings of this study are available from the University of Central Florida (UCF) website, [https://www.crcv.ucf.edu/data/UCF101.php].

Declarations

Conflict of interest The authors have no conflict of interest to declare to the best of their knowledge.

References

1. Bendale A, Boult T (2015) Towards open world recognition. In: Proceedings of the IEEE conference on computer vision and pattern recognition (CVPR). IEEE Press, Piscataway, pp 1893–1902
2. Willes J, Harrison J, Harakeh A, Finn C, Pavone M, Waslander S (2022) Bayesian embeddings for few-shot open world recognition. IEEE Trans Pattern Anal Mach Intell. https://doi.org/10.1109/TPAMI.2022.3201541
3. Mundt M, Hong Y, Pliushch I, Ramesh V (2023) A wholistic view of continual learning with deep neural networks: forgotten lessons and the bridge to active and open world learning. Neural Netw 160:306–336
4. Joseph K, Khan S, Khan FS, Balasubramanian VN (2021) Towards open world object detection. In: Proceedings of the IEEE/CVF conference on computer vision and pattern recognition. IEEE Press, Piscataway, pp 5830–5840
5. Jafarzadeh M, Dhamija AR, Cruz S, Li C, Ahmad T, Boult TE (2020) Open-world learning without labels. arXiv preprint arXiv:2011.12906
6. Carreira J, Zisserman A (2017) Quo vadis, action recognition? A new model and the kinetics dataset. In: Proceedings of the 30th IEEE conference on computer vision and pattern recognition (CVPR). IEEE Press, Piscataway, pp 4724–4733
7. Xie S, Sun C, Huang J, Tu Z, Murphy K (2018) Rethinking spatiotemporal feature learning: speed-accuracy trade-offs in video classification. In: Proceedings of the European conference on computer vision (ECCV). Springer, Heidelberg, pp 305–321
8. Tran D, Wang H, Torresani L, Ray J, LeCun Y, Paluri M (2018) A closer look at spatiotemporal convolutions for action recognition.

In: Proceedings of the IEEE conference on computer vision and pattern recognition (CVPR). IEEE Press, Piscataway, pp 6450–6459
9. Gutoski M, Lazzaretti AE, Lopes HS (2021) Deep metric learning for open-set human action recognition in videos. Neural Comput Appl 33:1207–1220
10. Gutoski M, Lazzaretti AE, Lopes HS (2021) Incremental human action recognition with dual memory. Image Vis Comput 116:1–15
11. Rudd EM, Jain LP, Scheirer WJ, Boult TE (2018) The extreme value machine. IEEE Trans Pattern Anal Mach Intell 40(3):762–768
12. He K, Zhang X, Ren S, Sun J (2016) Deep residual learning for image recognition. In: Proceedings of the IEEE conference on computer vision and pattern recognition (CVPR). IEEE Press, Piscataway, pp 770–778
13. Krizhevsky A, Sutskever I, Hinton GE (2012) ImageNet classification with deep convolutional neural networks. In: Proceedings of the 25th international conference on neural information processing systems, vol 1. Curran Associates, Red Hook, pp 1097–1105
14. Szegedy C, Liu W, Jia Y, SermarXivanet P, Reed S, Anguelov D, Erhan D, Vanhoucke V, Rabinovich A (2015) Going deeper with convolutions. In: Proceedings of the IEEE conference on computer vision and pattern recognition (CVPR). IEEE Press, Piscataway, pp 1–9
15. Wu CY, Zaheer M, Hu H, Manmatha R, Smola AJ, Krähenbühl P (2018) Compressed video action recognition. In: Proceedings of the IEEE conference on computer vision and pattern recognition (CVPR). IEEE Press, Piscataway, pp 6026–6035
16. Donahue J, Anne Hendricks L, Guadarrama S, Rohrbach M, Venugopalan S, Saenko K, Darrell T (2015) Long-term recurrent convolutional networks for visual recognition and description. In: Proceedings of the IEEE conference on computer vision and pattern recognition (CVPR). IEEE Press, Piscataway, pp 2625–2634
17. Simonyan K, Zisserman A (2014) Two-stream convolutional networks for action recognition in videos. In: Proceedings of the advances in neural information processing systems. MIT Press, Cambridge, pp 568–576
18. Wang L, Xiong Y, Wang Z, Qiao Y, Lin D, Tang X, Van Gool L (2016) Temporal segment networks: Towards good practices for deep action recognition. In: Proceedings of the European conference on computer vision (ECCV). Springer, Heidelberg, pp 20–36
19. Zhu Y, Lan Z, Newsam S, Hauptmann A (2018) Hidden two-stream convolutional networks for action recognition. In: Proceedings of the Asian conference on computer vision. Springer, Heidelberg, pp 363–378
20. Tran D, Bourdev L, Fergus R, Torresani L, Paluri M (2015) Learning spatiotemporal features with 3D convolutional networks. In: Proceedings of the IEEE international conference on computer vision (CVPR). IEEE Press, Piscataway, pp 4489–4497
21. Wang X, Girshick R, Gupta A, He K (2018) Non-local neural networks. In: Proceedings of the IEEE conference on computer vision and pattern recognition (CVPR). IEEE Press, Piscataway, pp 7794–7803
22. Wang Y, Zhou W, Zhang Q, Zhu X, Li H (2018) Low-latency human action recognition with weighted multi-region convolutional neural network. arXiv preprint arXiv:1805.02877
23. Ng JYH, Choi J, Neumann J, Davis LS (2018) Actionflownet: learning motion representation for action recognition. In: Proceedings of the IEEE winter conference on applications of computer vision (WACV). IEEE Press, Piscataway, pp 1616–1624
24. Wang L, Li W, Li W, van Gool L (2018) Appearance-and-relation networks for video classification. In: Proc. of the IEEE conference on computer vision and pattern recognition (CVPR). IEEE Press, Piscataway, pp 1430–1439
25. Hara K, Kataoka H, Satoh Y (2018) Can spatiotemporal 3D CNNs retrace the history of 2D CNNs and imagenet?. In: Proceedings of the IEEE conference on computer vision and pattern recognition (CVPR). IEEE Press, Piscataway, pp 6546–6555
26. Chen Y, Kalantidis Y, Li J, Yan S, Feng J (2018) Multi-fiber networks for video recognition. In: Proceedings of the European conference on computer vision (ECCV). Springer, Switzerland, pp 352–367
27. Gao M, Cai W, Liu R (2021) AGTH-Net: attention-based graph convolution-guided third-order hourglass network for sports video classification. J Healthc Eng 2021:1–10
28. Jing L, Parag T, Wu Z, Tian Y, Wang H (2021) Videossl: semi-supervised learning for video classification. In: Proceedings of the IEEE/CVF winter conference on applications of computer vision. IEEE Press, Piscataway, pp 1110–1119
29. Cao K, Ji J, Cao Z, Chang CY, Niebles JC (2020) Few-shot video classification via temporal alignment. In: Proceedings of the IEEE/CVF conference on computer vision and pattern recognition. IEEE Press, Piscataway, pp 10618–10627
30. Fu H, Maraghi VO, Faez K (2022) Class-incremental learning on video-based action recognition by distillation of various knowledge. Comput Intell Neurosci 2022:4879942
31. Busto PP, Iqbal A, Gall J (2020) Open set domain adaptation for image and action recognition. IEEE Trans Pattern Anal Mach Intell 42(2):1–15
32. Roitberg A, Al-Halah Z, Stiefelhagen R (2018) Informed democracy: voting-based novelty detection for action recognition. In: Proceedings of the British machine vision conference. BMVA, Durham, pp 1–14
33. Roitberg A, Ma C, Haurilet M, Stiefelhagen R (2020) Open set driver activity recognition. In: 2020 IEEE intelligent vehicles symposium (IV). IEEE Press, Piscataway, pp 1048–1053
34. Yang Y, Hou C, Lang Y, Guan D, Huang D, Xu J (2019) Open-set human activity recognition based on micro-Doppler signatures. Pattern Recogn 85:60–69
35. Al-Obaydy WNI, Suandi SA (2020) Automatic pose normalization for open-set single-sample face recognition in video surveillance. Multimed Tools Appl 79(3):2897–2915
36. Chen Z, Luo Y, Baktashmotlagh M (2021) Conditional extreme value theory for open set video domain adaptation. In: ACM multimedia Asia. Association for Computing Machinery, New York, pp 1–8
37. Wang Y, Song X, Wang Y, Xu P, Hu R, Chai H (2021) Dual metric discriminator for open set video domain adaptation. In: IEEE international conference on acoustics, speech and signal processing (ICASSP). IEEE Press, Piscataway, pp 8198–8202
38. Bao W, Yu Q, Kong Y (2021) Evidential deep learning for open set action recognition. In: Proceedings of the IEEE/CVF international conference on computer vision. IEEE Press, Piscataway, pp 13349–13358
39. French RM (1999) Catastrophic forgetting in connectionist networks. Trends Cogn Sci 3(4):128–135
40. Masana M, Liu X, Twardowski B, Menta M, Bagdanov AD, van de Weijer J (2020) Class-incremental learning: survey and performance evaluation. arXiv preprint arXiv:2010.15277
41. Delange M, Aljundi R, Masana M, Parisot S, Jia X, Leonardis A, Slabaugh G, Tuytelaars T (2021) A continual learning survey: defying forgetting in classification tasks. IEEE Trans Pattern Anal Mach Intell 1–26
42. Pfülb B, Gepperth A (2019) A comprehensive, application-oriented study of catastrophic forgetting in DNNs. In: Proceedings of the international conference on learning representations. OpenReview.net, Amherst, pp 1–14
43. Chaudhry A, Dokania PK, Ajanthan T, Torr PH (2018) Riemannian walk for incremental learning: understanding forgetting and intransigence. In: Proceedings of the European

conference on computer vision (ECCV). Springer, Heidelberg, pp 532–547
44. Rebuffi SA, Kolesnikov A, Sperl G, Lampert CH (2017) iCaRL: incremental classifier and representation learning. In: Proceedings of the IEEE conference on computer vision and pattern recognition. IEEE Press, Piscataway, pp 2001–2010
45. Castro FM, Marín-Jiménez MJ, Guil N, Schmid C, Alahari K (2018) End-to-end incremental learning. In: Proceedings of the European conference on computer vision (ECCV). Springer, Heidelberg, pp 233–248
46. Wu Y, Chen Y, Wang L, Ye Y, Liu Z, Guo Y, Fu Y (2019) Large scale incremental learning. In: Proceedings of the IEEE/CVF conference on computer vision and pattern recognition. IEEE Press, Piscataway, pp 374–382
47. Belouadah E, Popescu A (2019) Il2m: class incremental learning with dual memory. In: Proceedings of the IEEE/CVF international conference on computer vision (ICCV). IEEE Press, Piscataway, pp 583–592
48. Hou S, Pan X, Loy CC, Wang Z, Lin D (2019) Learning a unified classifier incrementally via rebalancing. In: Proceedings of the IEEE/CVF conference on computer vision and pattern recognition (CVPR). IEEE Press, Piscataway, pp 831–839
49. Kim Y, Kim E (2021) Clustering-guided incremental learning of tasks. In: International conference on information networking (ICOIN). IEEE Press, Piscataway, pp 417–421
50. Kirkpatrick J, Pascanu R, Rabinowitz N, Veness J, Desjardins G, Rusu AA, Milan K, Quan J, Ramalho T, Grabska-Barwinska A et al (2017) Overcoming catastrophic forgetting in neural networks. Proc Natl Acad Sci 114(13):3521–3526
51. Zenke F, Poole B, Ganguli S (2017) Continual learning through synaptic intelligence. In: Proceedings of the international conference on machine learning. PMLR, Sydney, pp 3987–3995
52. Aljundi R, Babiloni F, Elhoseiny M, Rohrbach M, Tuytelaars T (2018) Memory aware synapses: learning what (not) to forget. In: Proceedings of the European conference on computer vision (ECCV). Springer, Heidelberg, pp 139–154
53. Li Z, Hoiem D (2017) Learning without forgetting. IEEE Trans Pattern Anal Mach Intell 40(12):2935–2947
54. Michieli U, Zanuttigh P (2021) Knowledge distillation for incremental learning in semantic segmentation. Comput Vis Image Underst 205:1–16
55. Mallya A, Davis D, Lazebnik S (2018) Piggyback: adapting a single network to multiple tasks by learning to mask weights. In: Proceedings of the European conference on computer vision (ECCV). Springer, Heidelberg, pp 67–82
56. Masana M, Tuytelaars T, van Weijer J (2020) Ternary feature masks: continual learning without any forgetting. arXiv preprint arXiv:2001.08714
57. Rusu AA, Rabinowitz NC, Desjardins G, Soyer H, Kirkpatrick J, Kavukcuoglu K, Pascanu R, Hadsell R (2016) Progressive neural networks. arXiv preprint arXiv:1606.04671
58. Schwarz J, Czarnecki W, Luketina J, Grabska-Barwinska A, Teh YW, Pascanu R, Hadsell R (2018) Progress & compress: a scalable framework for continual learning. In: Proceedings of the international conference on machine learning. PMLR, Stockholm, pp 4528–4537
59. Aljundi R, Chakravarty P, Tuytelaars T (2017) Expert gate: lifelong learning with a network of experts. In: Proceedings of the IEEE conference on computer vision and pattern recognition (CVPR). IEEE Press, Piscataway, pp 3366–3375
60. Sokar G, Mocanu DC, Pechenizkiy M (2021) Spacenet: make free space for continual learning. Neurocomputing 439:1–11
61. Ma J, Tao X, Ma J, Hong X, Gong Y (2021) Class incremental learning for video action classification. In: IEEE international conference on image processing (ICIP). IEEE Press, Piscataway, pp 504–508
62. Wong SF, Kim TK, Cipolla R (2007) Learning motion categories using both semantic and structural information. In: Proceedings of the 2007 IEEE conference on computer vision and pattern recognition. IEEE press, Piscataway, pp 1–6
63. Blank M, Gorelick L, Shechtman E, Irani M, Basri R (2005) Actions as space-time shapes. In: IEEE international conference on computer vision, vol 1. IEEE Press, Piscataway, pp 1395–1402
64. Reddy KK, Liu J, Shah M (2009) Incremental action recognition using feature-tree. In: Proceedings of the 12th IEEE international conference on computer vision. IEEE press, Piscataway, pp 1010–1017
65. Weinland D, Boyer E, Ronfard R (2007) Action recognition from arbitrary views using 3d exemplars. In: Proceedings of the 2007 IEEE international conference on computer vision. IEEE Press, Piscataway, pp 1–7
66. Tang C, Li W, Wang P, Wang L (2018) Online human action recognition based on incremental learning of weighted covariance descriptors. Inf Sci 467:219–237
67. Wu X, Jia Y, Liang W (2010) Incremental discriminant-analysis of canonical correlations for action recognition. Pattern Recogn 43(12):4190–4197
68. Lu Y, Boukharouba K, Boonært J, Fleury A, Lecœuche S (2014) Application of an incremental SVM algorithm for on-line human recognition from video surveillance using texture and color features. Neurocomputing 126:132–140
69. Minhas R, Mohammed AA, Wu QMJ (2012) Incremental learning in human action recognition based on snippets. IEEE Trans Circuits Syst Video Technol 22(11):1529–1541
70. De Rosa R, Cesa-Bianchi N, Gori I, Cuzzolin F (2014) Online action recognition via nonparametric incremental learning. In: Proceedings of the British machine vision conference. BMVA Press, Guildford, pp 1–15
71. Boult TE, Cruz S, Dhamija AR, Gunther M, Henrydoss J, Scheirer WJ (2019) Learning and the unknown: surveying steps toward open world recognition. In: Proceedings of the AAAI conference on artificial intelligence, vol 33, pp 9801–9807
72. Li X, Wu A, Zheng WS (2018) Adversarial open-world person re-identification. In: Proceedings of the European conference on computer vision (ECCV). Springer, Switzerland, pp 280–296
73. Matta A, Pinto JR, Cardoso JS (2021) Mixture-based open world face recognition. In: World conference on information systems and technologies. Springer, Switzerland, pp 653–662
74. Leng Q, Ye M, Tian Q (2020) A survey of open-world person re-identification. IEEE Trans Circuits Syst Video Technol 30(4):1092–1108
75. Mancini M, Karaoguz H, Ricci E, Jensfelt P, Caputo B (2019) Knowledge is never enough: towards web aided deep open world recognition. In: IEEE international conference on robotics and automation (ICRA). IEEE Press, Piscataway, pp 9537–9543
76. Cen J, Yun P, Cai J, Wang MY, Liu M (2021) Deep metric learning for open world semantic segmentation. In: Proceedings of the IEEE/CVF international conference on computer vision (ICCV). IEEE Press, Piscataway, pp 15333–15342
77. Irfan B, Ortiz MG, Lyubova N, Belpaeme T (2021) Multi-modal open world user identification. ACM Trans Hum Robot Interact (THRI) 11(1):1–50
78. Mancini M, Naeem MF, Xian Y, Akata Z (2021) Open world compositional zero-shot learning. In: Proceedings of the IEEE/CVF conference on computer vision and pattern recognition (CVPR). IEEE Press, Piscataway, pp 5222–5230
79. Zhong Z, Zhu L, Luo Z, Li S, Yang Y, Sebe N (2021) Openmix: reviving known knowledge for discovering novel visual categories in an open world. In: IEEE/CVF conference on computer vision and pattern recognition (CVPR). IEEE Press, Piscataway, pp 9457–9465

80. Liu Z, Miao Z, Zhan X, Wang J, Gong B, Yu SX (2019) Large-scale long-tailed recognition in an open world. In: Proceedings of the IEEE/CVF conference on computer vision and pattern recognition. IEEE Press, Piscataway, pp 2537–2546
81. Jafarzadeh M, Ahmad T, Dhamija AR, Li C, Cruz S, Boult TE (2021) Automatic open-world reliability assessment. In: Proceedings of the IEEE/CVF winter conference on applications of computer vision. IEEE Press, Piscataway, pp 1984–1993
82. Shu Y, Shi Y, Wang Y, Zou Y, Yuan Q, Tian Y (2018) ODN: opening the deep network for open-set action recognition. In: Proceedings of the IEEE international conference on multimedia and expo (ICME). IEEE Press, Piscataway, pp 1–6
83. Shu Y, Shi Y, Wang Y, Huang T, Tian Y (2020) P-odn: prototype-based open deep network for open set recognition. Sci Rep 10:1–13
84. Hoffer E, Ailon N (2015) Deep metric learning using triplet network. In: Proceedings of the international workshop on similarity-based pattern recognition. Springer, Heidelberg, pp 84–92
85. Ward JH Jr (1963) Hierarchical grouping to optimize an objective function. J Am Stat Assoc 58(301):236–244
86. Soomro K, Zamir AR, Shah M (2012) UCF101: a dataset of 101 human actions classes from videos in the wild. arXiv preprint arXiv:1212.0402
87. Glorot X, Bengio Y (2010) Understanding the difficulty of training deep feedforward neural networks. In: Proceedings of the 13th international conference on artificial intelligence and statistics. Microtome Publishing, Brookline, pp 249–256
88. Rousseeuw PJ (1987) Silhouettes: a graphical aid to the interpretation and validation of cluster analysis. J Comput Appl Math 20:53–65
89. Youden WJ (1950) Index for rating diagnostic tests. Cancer 3(1):32–35
90. Strehl A, Ghosh J (2002) Cluster ensembles–a knowledge reuse framework for combining multiple partitions. J Mach Learn Res 3:583–617
91. Min E, Guo X, Liu Q, Zhang G, Cui J, Long J (2018) A survey of clustering with deep learning: from the perspective of network architecture. IEEE Access 6:39501–39514
92. Sarfraz S, Sharma V, Stiefelhagen R (2019) Efficient parameter-free clustering using first neighbor relations. In: Proceedings of the IEEE/CVF conference on computer vision and pattern recognition. IEEE Press, Piscataway, pp 8934–8943
93. Pelleg D, Moore AW et al (2000) X-means: extending k-means with efficient estimation of the number of clusters. In: Proceedings of the seventeenth international conference on machine learning, vol 1. PMLR, San Francisco, pp 727–734
94. Arthur D, Vassilvitskii S (2007) K-means++: the advantages of careful seeding. In: Proceedings of the Eighteenth Annual ACM-SIAM Symposium on Discrete Algorithms. Society for Industrial and Applied Mathematics, USA, pp 1027–1035
95. Van der Maaten L, Hinton G (2008) Visualizing data using t-SNE. J Mach Learn Res 9(11):2579–2605