Reconstructing Humpty Dumpty: Multi-feature Graph Autoencoder for Open Set Action Recognition

Dawei Du Ameya Shringi Anthony Hoogs Christopher Funk
Kitware

https://github.com/Kitware/graphautoencoder

Abstract

Most action recognition datasets and algorithms assume a closed world, where all test samples are instances of the known classes. In open set problems, test samples may be drawn from either known or unknown classes. Existing open set action recognition methods are typically based on extending closed set methods by adding post hoc analysis of classification scores or feature distances and do not capture the relations among all the video clip elements. Our approach uses the reconstruction error to determine the novelty of the video since unknown classes are harder to put back together and thus have a higher reconstruction error than videos from known classes. We refer to our solution to the open set action recognition problem as "Humpty Dumpty", due to its reconstruction abilities. Humpty Dumpty is a novel graph-based autoencoder that accounts for contextual and semantic relations among the clip pieces for improved reconstruction. A larger reconstruction error leads to an increased likelihood that the action can not be reconstructed, i.e.*, can not put Humpty Dumpty back together again, indicating that the action has never been seen before and is novel/unknown. Extensive experiments are performed on two publicly available action recognition datasets including HMDB-51 and UCF-101, showing the state-of-the-art performance for open set action recognition.*

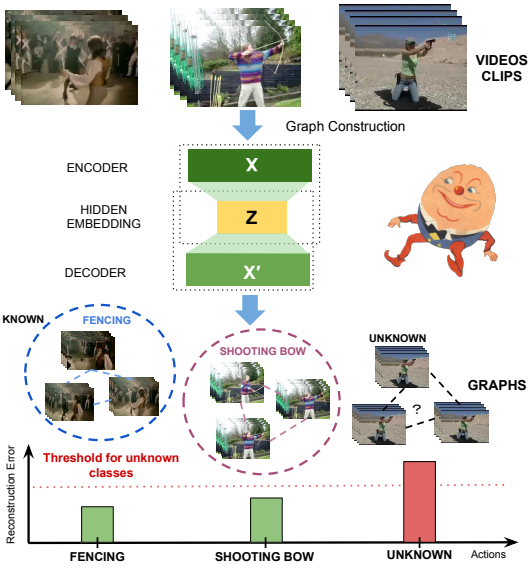


Figure 1: Humpty Dumpty is trained to minimize the reconstruction error between the extracted relations X in a video and the reconstructed relations X'. A large reconstruction error during inference is indicative of unknown relations, *i.e.*, an unknown action, in the video. The namesake of our approach is the Humpty Dumpty nursery rhyme, which tells that when "Humpty Dumpty had a great fall" and broke, analogous to converting the clip into the latent feature space, "all the King's horses and all the King's men couldn't put Humpty Dumpty" back together again [4] since his structure was unknown.

1. Introduction

Action recognition has garnered significant interest due to its potential applications in sports [38], surveillance [17], smart homes [49], etc. While significant progress has been made [3, 7, 12, 37, 40, 44] in recognizing activities on action recognition datasets such as HMDB-51 [19], UCF-101 [39], and Kinetics-700 [3], almost all this progress has been made under a closed world assumption [34] such that the test data only contains instances of trained classes. In this closed world setting, the algorithm predicts which of the known classes a test instance belongs to, and ignores the problem of whether the instance belongs to any of the unknown classes. The closed world assumption is not realistic for many real-world problems, where knowing all classes is not feasible. Closed world methods can be adapted to this open set paradigm by predicting that a test instance is unknown when the maximal prediction of any known class is sufficiently weak. This is because, discriminatively-trained, closed world models tend to be in-

effective at defining boundaries against background data not included in training [16]. This makes it difficult to tell the difference between an unusual instance of a known class and an instance of an unknown class.

To address these challenges, we develop an action recognition algorithm under the open set premise [34] that distinguishes known activities from unknown activities in the videos. Thereby, detecting novel activities [11] in videos. Unknown activities are not present in the training set. While they might share some semantic or contextual characteristics with known activities, the relations among these characteristics are significantly different. Our approach models these characteristics, and their relations as nodes and edges of a graph. It learns to reconstruct the graph for known activities using a multi-feature graph autoencoder. During testing, we use the reconstruction error to determine if the activity occurring in the video is known or unknown.

Recent work in open set action recognition has relied on thresholding softmax scores [29] or uncertainty in the classifier prediction [30] obtained from videos associated with known classes. A common theme among these techniques is treating the video as a single entity rather than exploiting the semantic relations within the video. Our approach captures these relations explicitly in a graph. To construct the graph, we divide the video into non-overlapping clips of fixed length. We obtain the contextual and semantic information for every clip using average pooling and max pooling, respectively, on the clip features. Using different pooling techniques to obtain multi-features or "views" for clustering is a common practice [50]. As shown in Figure 2, each node in the graph denotes the pooled feature associated with a clip, while edges correspond to contextual or semantic relations between different clips or within the same clip. The graph is encoded in a latent space using a graph convolutional network (GCN), which effectively learns the common contextual and semantics relationships for known classes. The graph is then reconstructed during the decoding phase. We rely on the reconstruction error obtained from the graph autoencoder to estimate open set risk directly based on these relations.

We conduct extensive experiments on two benchmark datasets for action recognition: HMDB-51 [19] and UCF-101 [39], where 50% of the action labels are designated as novel for each dataset. We evaluate our method using the Receiver Operating Characteristic's Area Under the Curve (ROC-AUC) and the mean Average Precision (mAP). Humpty Dumpty achieves about 2% and 3% improvement in ROC-AUC scores along with 4%, and 1% improvement in mAP scores over existing open set action recognition methods [15, 28, 30, 35] on the HMDB-51 and UCF-101 datasets, respectively. Extensive ablation studies also show the impact of different components of our proposed method. The main contributions of our work are:

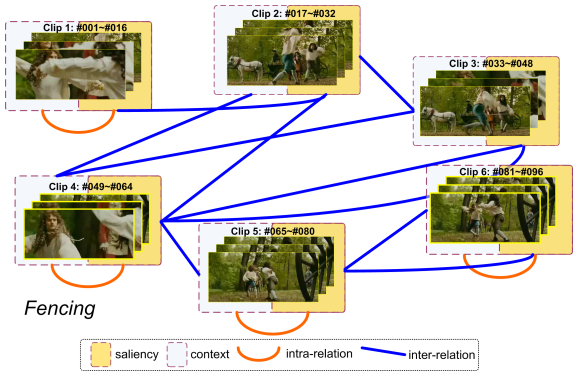


Figure 2: The underlying semantic relations among clips in a *Fencing* video. Video clips and the corresponding relations are partly displayed for more clarity.

- A novel multi-feature graph to represent actions, constructed from recognizing similar contextual (average pooling) and salient (max pooling) relations across video clips for improved reconstruction;
- A new method for open set action recognition through open-space risk estimation [34] that uses a graph-based autoencoder to learn the common semantics and features of known classes;
- Extensive experiments and ablation studies showing superior performance over existing methods on two action recognition datasets that have been previously established as benchmarks for novel action recognition.

2. Related Work

2.1. Open Set Recognition

Open set recognition [34] is originally proposed in the image recognition domain and has received significant attention [26, 47, 48] in recent years, including GMM [28], One-Class SVM [35], sparse coding [51], extreme value theory [23], and CNNs [27]. For example, Mundt *et al.* [23] combine model uncertainty with extreme-value theory for open set recognition using prediction uncertainty on a variety of classification tasks. Perera *et al.* [27] learn globally negative filters from an external dataset and then threshold the maximal activation of the proposed network to identify novel objects effectively.

However, limited open set work has been done in the action recognition domain. Recently, Roitberg *et al.* [30] leverage the estimated uncertainty of individual classifiers in their predictions and propose a voting-based scheme to measure the novelty of a new input sample for action recognition. Shi *et al.* [36] propose an open deep network to detect new categories by applying a multi-class triplet thresholding method, and then dynamically reconstructed

the classification layer by continually adding predictors for new categories. Furthermore, Busto *et al.* [2] propose an open set domain adaptation method for action recognition where the target domain contains instances of categories that are not present in the source domain. These methods focus on modeling the novelty with respect to individual video clips within a video; which does not account for the semantic relations across clips. Our method, via its graph construction, explicitly learns how the spatial-temporal relations within a clip are temporally linked across the video clips, resulting in improved performance.

Although our approach focuses on open set recognition, the relations captured by our approach can be used to detect anomalies in a video. It should be noted that there are fundamental differences between anomaly detection and open set recognition. Anomalies are instance outliers, usually still within one of the known classes, that generally occur rarely, whereas unknowns/novelties come from different classes and can occur frequently during inference [8, 22]. The label spaces are also different - anomaly detectors produce frame level output while activity recognition algorithms label the entire video as an activity. Despite the difference in formulation, autoencoders [10, 13, 52] have been applied frequently for detecting anomalies in video.

In addition, zero-shot learning focuses on the shared attributes between the known classes in training and the unknown classes during testing [8], while the focus of open set recognition is on determining the novelty of samples. They differ in the goal of the problem based on the information available during training and the desired result during testing. Therefore, a direct comparison would not be valid. There is a key difference in how each problem defines an unknown class - in zero-shot learning, unknown classes are specified through some shared side-information (such as attributes), while in open set recognition the unknown classes are truly not specified in any way during training [8].

2.2. Graph Convolution Networks

Increasing research attention has been devoted to generalizing neural networks [1, 9, 18, 21, 33, 46] to work on arbitrarily structured graphs. The early prominent research [1] developed a graph convolution based on spectral graph theory. Kipf and Welling [18] developed a framework for unsupervised learning on graph-structured data based on the variational auto-encoder. In the work of Salehi and Davulcu [33], the masked self-attentional layers are leveraged to attend over their neighbourhoods' features for a more discriminative representation. Gilmer *et al.* [9] propose message passing neural networks to exploit discriminative features from graph molecules. Wang *et al.* [46] apply graph networks to point clouds by developing the EdgeConv layer. Recently, Li *et al.* [21] adapted residual/dense connections, and dilated convolutions to train deeper graph convolutional networks. Inspired by [33], we introduce a self-attention encoding scheme in the graph autoencoder to enhance the multi-features by weighting the nodes based-on information from neighboring nodes, achieving better graph reconstruction ability.

2.3. Action Recognition

In recent years, deep learning based methods [3, 12, 40] have dominated the field of video action recognition compared with traditional methods [20, 32, 43]. Among these approaches, 3D CNNs [12,41] have been effective in encoding the temporal information in the video. Tran *et al.* [40] proposed temporal feature learning using deep 3D convolutional networks (C3D). 3D ResNets [12] extended 2D ResNets [14] blocks and obtained better performance than the shallower C3D network. I3D [3] is a two-stream inflated 3D Convolutional Neural Network that uses RGB and Optical Flow to considerably improve the state-of-the-art in action recognition. Feichtenhofer *et al.* [6] proposed the SlowFast network for video recognition, including a Slow pathway to capture spatial semantics, and a Fast pathway to capture motion at fine temporal resolution. Moreover, Feichtenhofer developed another X3D [5] backbone, which is progressively expanded from a tiny spatial ResNet network.

Since 3D CNNs model the temporal extent of a video using convolution operations, they have a fairly limited temporal range due to memory constraints on a GPU. Xi *et al.* [24] combined 3D CNN with LSTM for action recognition. During graph construction, we use appearance similarity and temporal threshold to create an edge among the video clips. Wang *et al.* [45] used a non-local self-attention layer to calculate temporal similarity for action recognition. Our approach extracts temporal features from each non-overlapping video clip associated with the video. Unlike other approaches, we also then define contextual and semantic relations across these clips.

3. Approach

As discussed above, our task is open set action recognition. Given a set of labeled videos with $\mathcal{L}_\mathcal{K}$ known classes available for training, and a set of unlabeled videos that contains a mixture of $\mathcal{L}_\mathcal{K}$ known classes along with $\mathcal{L}_\mathcal{U}$ unknown classes that appear only during testing, our goal is to classify the unlabeled video x_j as either "known" if $x_j \in \mathcal{L}_\mathcal{K}$ or "unknown" if $x_j \in \mathcal{L}_\mathcal{U}$ by exploiting underlying semantic relations obtained from the training videos. To this end, our network represents intra-clip and inter-clip semantic relations associated with a video, and reconstructs the topographical information present in the graph. As illustrated in Figure 3, our model consists of three components: action feature extraction, video graph construction, and multi-feature graph autoencoder.

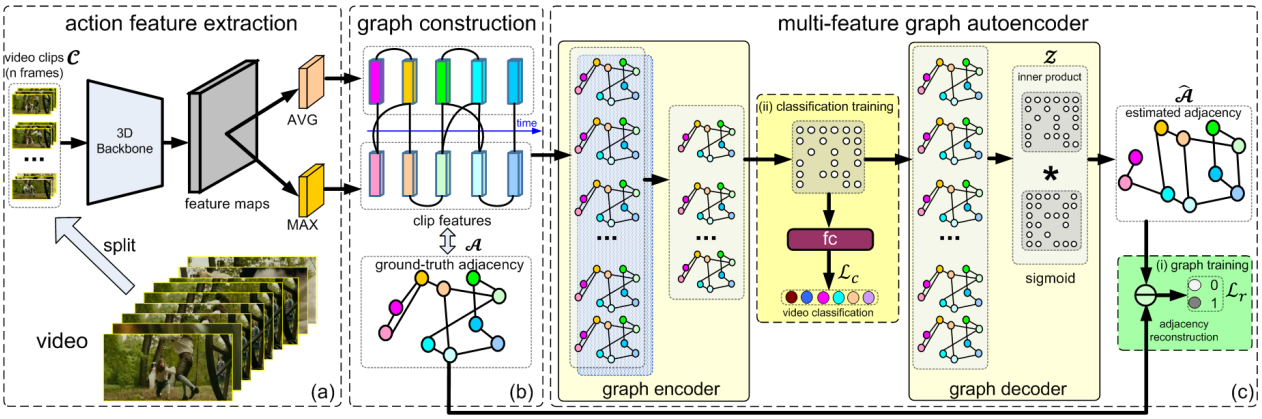


Figure 3: Architecture of the Humpty Dumpty approach, including the (a) action feature extraction, (b) video graph construction, and (c) multi-feature graph autoencoder, as well as the two losses (i) reconstruction and (ii) known-class classification. The reconstruction error is used during evaluation to determine how the novelty of a given video. AVG and MAX stand for average pooling and max pooling layers respectively.

3.1. Overview

As shown in Figure 3(a), we split the video into $\mathcal{C}$ clips such that the clips have no temporal overlap and each clip has n image frames. For each clip $c \in \mathcal{C}$, we extract the feature vectors associated with all the n frames in c. To obtain the multi-feature representation, we employ two kinds of pooling layers: maximal pooling and average pooling. The maximal pooling layer encodes the salient features present in the video clip, while the average pooling layer encodes the context information for the video clip. The multi-features are used to build the video graph, where each node represents the pooled feature obtained from a video clip and each edge connects two nodes with sufficient temporal and appearance similarity.

During the learning stage, a graph autoencoder is trained to reconstruct the graph obtained in the previous step. As with typical autoencoder methods, each training video provides its own groundtruth in the form of its graph. During the encoding phase, we apply a graph convolutional network (GCN) to map the video graph to a latent space. Furthermore, we employ self attention [42] to capture the most discriminative relations present in the graph. To learn to distinguish between known classes, we add a fully connected layer upon the latent space to classify the known instances in $\mathcal{L}_\mathcal{K}$ classes. In the decoding phase, the adjacency matrix is reconstructed by applying GCN over the latent space followed by a Sigmoid layer. During training, we minimize the reconstruction error between the original graph and the reconstructed graph. During inference (testing), the reconstruction error is used to measure the novelty of a video.

3.2. Multi-Feature Graph Autoencoder

The graph autoencoder is the crucial component of our approach. Inspired from the recent successes of relational modeling using a graph autoencoder [18], we use it to reconstruct temporal clip similarities based on the underlying relations.

Video Graph Construction. Given the multi-feature representation of all the clips in a video, we construct an undirected unweighted graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ to represent pair-wise relations among the video clips. Using the 3D backbone B, the pooled features obtained from a video clip are represented by the nodes and the semantic relations present between these features are represented by the edges, as shown in Figure 3(b). Formally, the nodes are a set of pooled features, *i.e.*, $\mathcal{V} = \{v_i \mid \forall v_i = \delta_{\text{avg}}(B(c_i)) \cup \delta_{\text{max}}(B(c_i)), c_i \in \mathcal{C}\}$ where δ_{max}, and δ_{avg} denotes the maximal and average pooling operations respectively. $\mathcal{C}$ is defined as the number of pooled node features within a video. The edge $e_{i,j} \in \mathcal{E}$ between nodes v_i and v_j is obtained using appearance similarity $f_a(v_i, v_j)$ and temporal distance $f_t(v_i, v_j)$ based on

$$e_{i,j} = \mathbb{1}(f_a(v_i, v_j) \leq \theta_a, f_t(v_i, v_j) \leq \theta_t), \qquad (1)$$

where $\mathbb{1}(\cdot) = 1$ if the argument is true and 0 otherwise, and θ_a and θ_t are the thresholds for the appearance and temporal distance respectively. Thus the video graph can model the short-term temporal consistency and semantic similarity. We use a binary adjacency matrix $\mathcal{A}$ to represent the edges of the graph when graph convolution is applied on the graph $\mathcal{G}$. The value 1/0 at row i and column j of $\mathcal{A}$ indicates that there is an/no edge between node v_i and v_j. The ground-truth adjacency matrix $\mathcal{A}$ is calculated as

$$\mathcal{A}(i,j) = \begin{cases} 1 & \text{if} \quad e_{i,j} = 1, \\ 0 & \text{otherwise.} \end{cases} \qquad (2)$$

In our experiments, we use the euclidean distance between the features associated with node v_i and v_j to calculate

the appearance similarity $f_a(v_i, v_j)$. The temporal distance $f_t(v_i, v_j)$ is the absolute difference (counted by the number of frames) between the center frames for clips c_i and c_j. The node features f are aggregated in a $C \times D$ matrix $\mathcal{F}$, where D is the dimension of the clip feature. Since the graph is undirected, the adjacency matrix is symmetric, *i.e.*, $\mathcal{A}(i, j) = \mathcal{A}(j, i)$.

Self-attention Graph Encoder. To learn the underlying semantic relations captured by the graph $\mathcal{G}$, we use a multi-feature graph autoencoder. As shown in Figure 3(c), the graph encoder learns a layer-wise transformation using the graph convolution (GC) over the the adjacency matrix $\mathcal{A}$ and the feature matrix $\mathcal{F}$ as follows.

$$\mathcal{Z}^{l+1} = \text{GC}(\mathcal{Z}^l, \mathcal{A}; \mathcal{W}^l), \tag{3}$$

where $\mathcal{Z}^l$ and $\mathcal{Z}^{l+1}$ are the input and output of the graph convolution at layer l and $\mathcal{Z}^0 = \mathcal{F} \in \mathbb{R}^{C \times D}$, the original graph constructed as above. $\mathcal{W}^l$ is the filter parameters of the corresponding graph convolutional layer in the network. For clarity, the index of layer l is omitted in the following.

To enhance the discriminative representation of graph data, we further introduce the graph attention layer [42] to perform self-attention encoding. The input of the graph attention layer is a set of node features $\mathcal{Z} = \{z_1, z_2, \cdots, z_C\}$; the output is a new set of node features with different cardinality $\mathcal{Z}' = \{z'_1, z'_2, \cdots, z'_C\}$. We use K attention heads without sharing weights to obtain a diverse set of representations associated with the node features $\mathcal{Z}$. The node features z_i are computed by aggregating the features associated with node v_i and its neighbouring nodes. The aggregated feature are averaged over the attention heads to generate the updated feature z'_i of node v_i, *i.e.*,

$$z'_i = \sigma_s\Big(\frac{1}{K}\sum_{k=1}^{K}\sum_{j \in \mathcal{N}_{v_i} \cup v_i} \alpha_{ij}^k \mathcal{W}_k z_j\Big), \tag{4}$$

where σ_s denotes a non-linear softmax activation function. $\mathcal{N}_{v_i}$ denotes the neighbouring nodes of node v_i, and $\mathcal{W}_k$ is the corresponding input linear transformation's weight matrix. α_{ij}^k denotes normalized attention coefficients computed by the k-th attention head, which are computed as

$$\alpha_{ij}^k = \frac{\exp\big(\sigma_l(\alpha^T(\mathcal{W}_k z_i \mathbin{+\!\!+} \mathcal{W}_k z_j))\big)}{\sum_{j \in \mathcal{N}_{v_i} \cup i} \exp\big(\sigma_l(\alpha^T(\mathcal{W}_k z_i \mathbin{+\!\!+} \mathcal{W}_k z_j))\big)}, \tag{5}$$

where $\mathbin{+\!\!+}$ is the concatenation operation, and σ_l is the leaky ReLU activation function. The weight vector $\alpha^T = \gamma(\mathcal{W}_k z_i, \mathcal{W}_k z_j)$ measures the importance of node v_j's features to node v_i, which is calculated by a single-layer feed-forward neural network γ. As shown in Figure 3, we only use the self-attention layer in the first graph convolutional layer of the graph encoder.

Graph Decoder. After encoding the graph, we obtain the hidden embedding $\mathcal{Z}$ for graph reconstruction. As shown in Figure 3(c), we use another graph convolutional layer to refine $\mathcal{Z}$ with higher cardinality. Followed by the dropout operation, we use the graph decoder to learn the similarity of each row in the hidden embedding to obtain the output adjacency matrix. Similar to the work of Kipf and Welling [18], we use the inner product to calculate the cosine similarity between each feature in the hidden embedding since it is invariant to the magnitude of features. In other words, by applying the inner product on the hidden embedding $\mathcal{Z}$ and $\mathcal{Z}^T$, we can learn the similarity of each node in $\mathcal{Z}$ and generate a reconstructed adjacency matrix $\hat{\mathcal{A}}$ as

$$\hat{\mathcal{A}} = \text{Sigmoid}(\mathcal{Z}\mathcal{Z}^T). \tag{6}$$

3.3. Training and Inference

The network is trained by minimizing the discrepancy between the ground-truth adjacency matrix $\mathcal{A}$ and the reconstructed adjacency matrix $\hat{\mathcal{A}}$.

Loss Function. To train the proposed network, we use two loss terms - (i) classification loss $\mathcal{L}_c$ and (ii) reconstruction loss $\mathcal{L}_r$. The classification loss penalizes the network for misclassifying known class samples. The reconstruction loss penalizes the network for poor reconstruction of the original adjacency matrix. The overall loss function is defined as

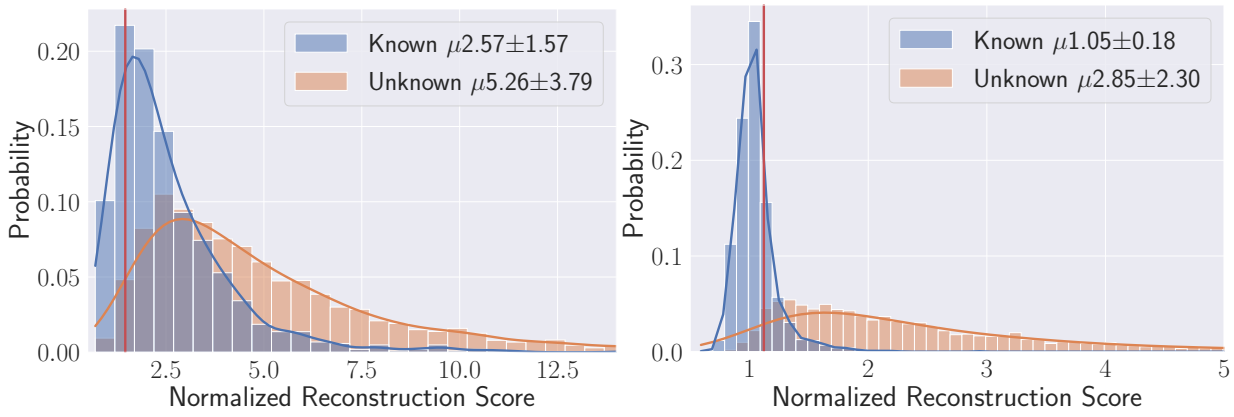
$$\begin{aligned}\mathcal{L} &= \mathcal{L}_c + \mathcal{L}_r \\ &= \frac{1}{N}\sum_{i=1}^{N} \ell_c(\mathcal{P}(x_i), y_{x_i}; \mathcal{Z}_{x_i}) + \frac{1}{N}\sum_{i=1}^{N} \ell_r(\mathcal{A}_{x_i}, \hat{\mathcal{A}}_{x_i}),\end{aligned} \tag{7}$$

where $\mathcal{P}(x_i)$ and y_{x_i} are the softmax probability and the ground-truth class label of sample x_i respectively. $\mathcal{Z}_{x_i}$ is the corresponding latent representation. The function ℓ_c is cross-entropy for the classification loss and the function ℓ_r is binary cross-entropy for the reconstruction loss.

Adjacency Matrix Re-construction. Since the length of every video varies in the dataset, we use the normalized reconstruction score $\mathcal{S}(x_i)$ of the graph $\mathcal{G}(x_i)$ to measure the novelty degree of the corresponding sample x_i, *i.e.*,

$$\mathcal{S}(x_i) = \beta \cdot \ell_r(\hat{\mathcal{A}}(x_i), \mathcal{A}(x_i)), \tag{8}$$

where $\hat{\mathcal{A}}$ and $\mathcal{A}$ are the estimated and ground-truth adjacency matrix respectively. ℓ_r is the binary cross-entropy function. To adjust to different sizes of adjacency, we use $\beta = \frac{N_\mathcal{V}^2 - N_\mathcal{E}}{N_\mathcal{E}}$ as the normalization ratio. $N_\mathcal{V}$ and $N_\mathcal{E}$ denote the number of nodes and edges in graph $\mathcal{G}$ respectively.



(a) HMDB-51 dataset

(b) UCF-101 dataset

Figure 4: Normalized Reconstruction Score (NRS) $\mathcal{S}(x_i)$ histograms for the known and unknown classes on the testing set. The x-axis indicates the normalized reconstruction score while the y-axis indicates the counts of instances. The red line denotes the max F1 score threshold for the dataset.

Inference. To determine the open-space risk associated with a video, we compute the multi-feature representation of every video in the test set. These representations are passed through the graph autoencoder to determine the reconstruction error associated with the video. Given the ground-truth adjacency matrix computed by Equation (2), we can estimate the reconstruction error to measure the degree of novelty by empirically setting the threshold. As shown in Figure 4, the video is regarded as "unknown" if the reconstruction error is above the threshold.

4. Experiments

The proposed method is implemented using Pytorch [25] and all experiments are conducted on a workstation with an NVIDIA Titan X GPU card.

As discussed in the related work, few novelty detection methods have been used for video action recognition[1]. According to [30], we compare our model to several baseline novelty detection methods. The One-Class SVM [35] with an RBF kernel can model the videos of one (or all) known class(es), where the upper bound on the fraction of training errors n is set to 0.1. The Gaussian Mixture Model (GMM) [28] generatively represents the videos in sub-spaces with 8 components, indicating novelty for any sample with sufficiently low probability. Softmax probabilities [15] of the I3D network [3] use the score output for a rejection thresholding to determine the novelty degree. Uncertainty [30] is calculated by a Bayesian neural network (BNN) posterior distribution with different network parameters. Informed Democracy [30] exploits the uncertainty of the output neurons in a voting scheme for novelty detection.

[1]Since there are no available source codes for the open set action recognition methods [2, 36], we cannot compare them in our experiment.

Datasets. We use HMDB-51 [19] and UCF-101 [39] to evaluate our method. HMDB-51 is a video dataset with $6,766$ manually annotated videos and 51 action classes. Each class includes at least 101 clips, each with duration of at least 1 second. It was collected from digitized movies and YouTube. UCF-101 is another large video dataset with $13,320$ clips and 101 action classes. The clip length varies from 1.06 second to 71.04 second. The videos are collected from user-uploaded videos over the Internet. Following the work in [30], each dataset is split evenly into known/unknown classes. For a fair comparison, we use the same dataset splits as Roitberg *et al.* [30] and formulate open set action recognition as a binary classification problem. That is, we split HMDB-51 in $26/25$ known/unknown classes and UCF-101 in $51/50$ known/unknown classes. Without determining the specific threshold, we use area under curve (AUC) values of the receiver operating characteristic (ROC), precision-recall (PR) curves and F1 scores to evaluate compared methods.

Implementation Details. The evaluation datasets contain many short clips with durations of several seconds. The network is optimized using Adam with a learning rate of $1e-5$ for 300 epochs. The similarity thresholds in Equation (2) are determined empirically and set to $\theta_a = 30$ and $\theta_t = 2n = 32$. The number of heads for self-attention encoding is set to $K = 4$.

4.1. Result Analysis

As presented in Table 1, it can be seen that our method outperforms existing approaches considerably. Our method achieves an ROC-AUC gain of 3.17% and 1.90% on the HMDB-51 and UCF-101 datasets, respectively. Meanwhile, the corresponding standard deviation is smaller than

Values reported as $\mu \pm \sigma$	HMDB-51 [19]		UCF-101 [39]	
	ROC-AUC	mAP	ROC-AUC	mAP
One-class SVM [35]	54.09(±3.0)	77.86(±4.0)	53.55(±2.0)	78.57(±2.4)
Gaussian Mixture Model [28]	56.83(±4.2)	78.40(±3.6)	59.21(±4.2)	79.50(±2.2)
Softmax Confidence [15]	67.58(±3.3)	84.21(±3.0)	84.28(±1.9)	93.92(±0.7)
Uncertainty [30]	71.78(±1.8)	86.81(±2.5)	91.43(±2.3)	96.72(±1.0)
Informed Democracy [30]	75.33(±2.7)	88.66(±2.3)	92.94(±1.7)	97.52(±0.6)
Humpty Dumpty (ours)	**78.50**(±**0.8**)	**91.39**(±**0.9**)	**94.84**(±**1.2**)	**98.38**(±**0.3**)

Table 1: Open set recognition results (mean and standard deviation over ten dataset splits).

that in existing methods. This can be attributed to two reasons. First, we focus on modeling the temporal semantic relations among video clips instead of the appearance information of individual video clips. Second, the self-attention encoding scheme can capture the common representation among multi-features far more robustly.

In Figure 4, we show the reconstruction score distributions for known and unknown classes on the testing sets of the above two datasets. The mean reconstruction scores of unknown classes are larger than that of known classes, 5.26 vs. 2.57 (Figure 4a). However, there exists some overlap between known and unknown distributions, resulting in weaker performance on HMDB-51. In contrast, the majority of reconstruction scores of known testing videos in UCF-101 is less than 1.00 (Figure 4b). This could be because we have fewer training samples on HMDB-51 than UCF-101. Figure 5 shows some qualitative results obtained at different values of reconstruction error, where we chose the threshold exclusively for the qualitative visualization.

4.2. Ablation Studies

We further study the influence of three important components of our method: (a) multi-feature representation, (b) self-attention encoding, and (c) graph autoencoder. The ablation studies are conducted on a single split of HMDB-51.

Effectiveness of multiple features. To investigate the importance of the multiple feature representation, we test the system using only the average or the maximal pooling layer of the backbone (denoted as "AVG" and "MAX" in Figure 3(a)) to calculate the features of clips. From Table 2, the "MAX" variant performs similarly compared to the "AVG" variant. Moreover, using both features performs best by a comfortable margin in terms of all three metrics, a 2.1 gain in ROC-AUC, 1.1 gain in mAP and 0.67 gain in max F1. We conclude that the multiple features are partly complementary in capturing semantic information in videos.

Effectiveness of self-attention encoding. To verify the effectiveness of self-attention encoding, we enumerate the number of heads of self-attention encoding $K = \{1, 2, 4, 6\}$ in Equation (4). $K = 1$ implies that our method uses the convolutional graph layer instead of self-attention encoding to capture the relations among video clips. From the results shown in Table 3, the ROC-AUC and AP scores increase with K before the optimal performance ($K = 4$) is achieved. This result indicates that self-attention encoding can model the temporal semantic relations among video clips effectively. However, the effect is not large and it is difficult for more heads of self-attention encoding to improve the performance. For $K = 6$, we achieve slightly inferior ROC-AUC and mAP scores even though our F1 score improves. Due to better performance on both ROC-AUC and mAP scores, we set $K = 4$ in our primary results.

Pooling Features	F1 Max	ROC-AUC	mAP
MAX	87.81	76.89	91.07
AVG	87.39	76.61	91.06
Multi-Feature	**88.06**	**79.01**	**92.16**

Table 2: Open set recognition results in terms of multiple features.

#K	F1 Max	ROC-AUC	mAP
1	87.88	76.38	91.04
2	88.07	77.76	91.38
4	88.06	**79.01**	**92.16**
6	**88.14**	78.93	92.04

Table 3: Open set recognition results in terms of different heads of self-attention encoding.

Effectiveness of graph autoencoder. To investigate the effectiveness of the graph autoencoder, we compare our method to two well-known open set algorithms, the Extreme Value Machine (EVM) [31] and Auto-encoder [52], which are trained on the same multiple features as that in Humpty Dumpty. As shown in Table 4, we find a sharp decline in two metrics using EVM, *i.e.*, 69.24 vs. 79.01 in

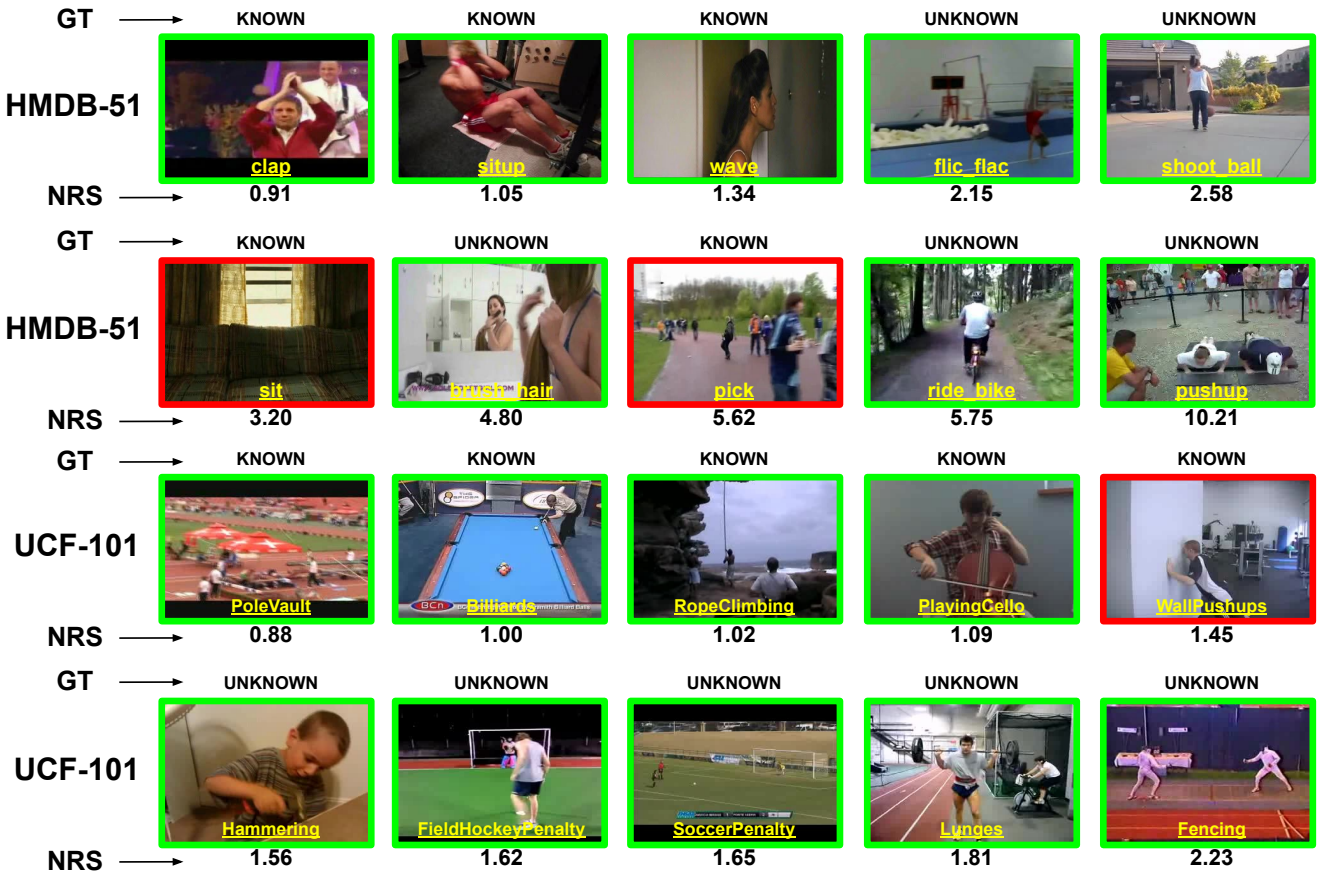


Figure 5: Qualitative results on the HMDB-51 and UCF-101 datasets, where the videos are randomly selected. The color of the box denotes if our method correctly (green box) or incorrectly (red box) identified whether the clip is from a known or unknown class.

Open Set Algorithms	F1 Max	ROC-AUC	mAP
EVM [31]	87.49	69.24	86.51
Auto-encoder [52]	86.93	44.39	73.27
Humpty Dumpty (ours)	**88.06**	**79.01**	**92.16**

Table 4: Open set recognition results of EVM, Autoencoder, and our method.

ROC-AUC score, 86.51 vs. 92.16 in mAP score. This indicates that the hidden embedding in our graph autoencoder along with reconstruction error is more effective than the Weibull distribution used by EVM [31].

Additionally, although our graph autoencoder and the standard autoencoder share similar concepts such as an encoder, decoder, and reconstruction, the standard autoencoder learns an approximation to the identity function such that the output features are similar to the original features, but it fails to capture the underlying relations among different features. In contrast, our graph autoencoder exploits contextual and semantic relations among video clips and then reconstructs the adjacency matrix of the video graph, resulting in much better performance.

5. Conclusion

In this work, we propose a new multi-feature graph autoencoder, Humpty Dumpty, to address open set action recognition. These results show that multiple features help the autoencoder learn a more robust reconstruction for known classes by exploiting salient and context information from the video. A self-attention step helps focus the network on the most important nodes, though too many self-attention heads have diminishing returns. Finally, we show that Humpty Dumpty can achieve state-of-the-art performance for open set action recognition on two datasets.

Acknowledgements. This material is based upon work supported by the Defense Advanced Research Projects Agency (DARPA) under Contract No. HR001120C0055. Any opinions, findings and conclusions or recommendations expressed in this material are those of the author(s) and do not necessarily reflect the views of the DARPA.

References

[1] Joan Bruna, Wojciech Zaremba, Arthur Szlam, and Yann LeCun. Spectral networks and locally connected networks on graphs. In *ICLR*, 2014.

[2] Pau Panareda Busto, Ahsan Iqbal, and Juergen Gall. Open set domain adaptation for image and action recognition. *TPAMI*, 42(2):413–429, 2020.

[3] João Carreira and Andrew Zisserman. Quo vadis, action recognition? A new model and the kinetics dataset. In *CVPR*, pages 4724–4733, 2017.

[4] L. Carroll and J. Tenniel. *Through the Looking Glass: And what Alice Found There*. Altemus' illustrated young people's library. Henry Altemus Company, 1897.

[5] Christoph Feichtenhofer. X3D: expanding architectures for efficient video recognition. In *CVPR*, pages 200–210, 2020.

[6] Christoph Feichtenhofer, Haoqi Fan, Jitendra Malik, and Kaiming He. Slowfast networks for video recognition. In *ICCV*, pages 6201–6210, 2019.

[7] Christoph Feichtenhofer, Axel Pinz, and Andrew Zisserman. Convolutional two-stream network fusion for video action recognition. In *CVPR*, pages 1933–1941, 2016.

[8] Chuanxing Geng, Sheng-Jun Huang, and Songcan Chen. Recent advances in open set recognition: A survey. *TPAMI*, 43(10):3614–3631, 2021.

[9] Justin Gilmer, Samuel S. Schoenholz, Patrick F. Riley, Oriol Vinyals, and George E. Dahl. Neural message passing for quantum chemistry. In *ICML*, volume 70, pages 1263–1272, 2017.

[10] Dong Gong, Lingqiao Liu, Vuong Le, Budhaditya Saha, Moussa Reda Mansour, Svetha Venkatesh, and Anton van den Hengel. Memorizing normality to detect anomaly: Memory-augmented deep autoencoder for unsupervised anomaly detection. In *ICCV*, pages 1705–1714, 2019.

[11] Omkar Gune, Amit More, Biplab Banerjee, and Subhasis Chaudhuri. Generalized zero-shot learning using open set recognition. In *BMVC*, page 213, 2019.

[12] Kensho Hara, Hirokatsu Kataoka, and Yutaka Satoh. Can spatiotemporal 3d cnns retrace the history of 2d cnns and imagenet? In *CVPR*, pages 6546–6555, 2018.

[13] Mahmudul Hasan, Jonghyun Choi, Jan Neumann, Amit K. Roy-Chowdhury, and Larry S. Davis. Learning temporal regularity in video sequences. In *CVPR*, pages 733–742, 2016.

[14] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *CVPR*, pages 770–778, 2016.

[15] Dan Hendrycks and Kevin Gimpel. A baseline for detecting misclassified and out-of-distribution examples in neural networks. In *ICLR*, 2017.

[16] Lalit P Jain, Walter J Scheirer, and Terrance E Boult. Multi-class open set recognition using probability of inclusion. In *ECCV*, pages 393–409, 2014.

[17] Shuiwang Ji, Wei Xu, Ming Yang, and Kai Yu. 3d convolutional neural networks for human action recognition. *TPAMI*, 35(1):221–231, 2012.

[18] Thomas N. Kipf and Max Welling. Variational graph auto-encoders. *CoRR*, abs/1611.07308, 2016.

[19] Hildegard Kuehne, Hueihan Jhuang, Estíbaliz Garrote, Tomaso A. Poggio, and Thomas Serre. HMDB: A large video database for human motion recognition. In *ICCV*, pages 2556–2563, 2011.

[20] Ivan Laptev and Tony Lindeberg. Space-time interest points. In *ICCV*, pages 432–439, 2003.

[21] Guohao Li, Matthias Müller, Ali K. Thabet, and Bernard Ghanem. Deepgcns: Can gcns go as deep as cnns? In *ICCV*, pages 9266–9275, 2019.

[22] Marc Masana, Idoia Ruiz, Joan Serrat, Joost van de Weijer, and Antonio M. López. Metric learning for novelty and anomaly detection. In *BMVC*, page 64, 2018.

[23] Martin Mundt, Iuliia Pliushch, Sagnik Majumder, and Visvanathan Ramesh. Open set recognition through deep neural network uncertainty: Does out-of-distribution detection require generative classifiers? In *ICCVW*, pages 753–757, 2019.

[24] Xi Ouyang, Shuangjie Xu, Chaoyun Zhang, Pan Zhou, Yang Yang, Guanghui Liu, and Xuelong Li. A 3d-cnn and lstm based multi-task learning architecture for action recognition. *Access*, 7:40757–40770, 2019.

[25] Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, et al. Pytorch: An imperative style, high-performance deep learning library. In *NeurIPS*, pages 8026–8037, 2019.

[26] Pramuditha Perera, Vlad I. Morariu, Rajiv Jain, Varun Manjunatha, Curtis Wigington, Vicente Ordonez, and Vishal M. Patel. Generative-discriminative feature representations for open-set recognition. In *CVPR*, pages 11811–11820, 2020.

[27] Pramuditha Perera and Vishal M. Patel. Deep transfer learning for multiple class novelty detection. In *CVPR*, pages 11544–11552, 2019.

[28] Marco A. F. Pimentel, David A. Clifton, Lei A. Clifton, and Lionel Tarassenko. A review of novelty detection. *Signal Process.*, 99:215–249, 2014.

[29] Sebastian Ramos, Stefan Gehrig, Peter Pinggera, Uwe Franke, and Carsten Rother. Detecting unexpected obstacles for self-driving cars: Fusing deep learning and geometric modeling. In *Intelligent Vehicles Symposium*, pages 1025–1032, 2017.

[30] Alina Roitberg, Ziad Al-Halah, and Rainer Stiefelhagen. Informed democracy: Voting-based novelty detection for action recognition. In *BMVC*, page 52, 2018.

[31] Ethan M. Rudd, Lalit P. Jain, Walter J. Scheirer, and Terrance E. Boult. The extreme value machine. *TPAMI*, 40(3):762–768, 2018.

[32] Sreemanananth Sadanand and Jason J. Corso. Action bank: A high-level representation of activity in video. In *CVPR*, pages 1234–1241, 2012.

[33] Amin Salehi and Hasan Davulcu. Graph attention auto-encoders. *CoRR*, abs/1905.10715, 2019.

[34] Walter J Scheirer, Anderson de Rezende Rocha, Archana Sapkota, and Terrance E Boult. Toward open set recognition. *TPAMI*, 35(7):1757–1772, 2012.

[35] Bernhard Schölkopf, Robert C. Williamson, Alexander J. Smola, John Shawe-Taylor, and John C. Platt. Support vector

method for novelty detection. In *NeurIPS*, pages 582–588, 1999.

[36] Yemin Shi, Yaowei Wang, Yixiong Zou, Qingsheng Yuan, Yonghong Tian, and Yu Shu. ODN: opening the deep network for open-set action recognition. In *ICME*, pages 1–6, 2018.

[37] Karen Simonyan and Andrew Zisserman. Two-stream convolutional networks for action recognition in videos. In *NeurIPS*, pages 568–576, 2014.

[38] Khurram Soomro and Amir R Zamir. Action recognition in realistic sports videos. In *Computer vision in sports*, pages 181–208. 2014.

[39] Khurram Soomro, Amir Roshan Zamir, and Mubarak Shah. UCF101: A dataset of 101 human actions classes from videos in the wild. *CoRR*, abs/1212.0402, 2012.

[40] Du Tran, Lubomir D. Bourdev, Rob Fergus, Lorenzo Torresani, and Manohar Paluri. Learning spatiotemporal features with 3d convolutional networks. In *ICCV*, pages 4489–4497, 2015.

[41] Du Tran, Heng Wang, Lorenzo Torresani, Jamie Ray, Yann LeCun, and Manohar Paluri. A closer look at spatiotemporal convolutions for action recognition. In *CVPR*, pages 6450–6459, 2018.

[42] Petar Velickovic, Guillem Cucurull, Arantxa Casanova, Adriana Romero, Pietro Liò, and Yoshua Bengio. Graph attention networks. *CoRR*, abs/1710.10903, 2017.

[43] Heng Wang, Alexander Kläser, Cordelia Schmid, and Cheng-Lin Liu. Action recognition by dense trajectories. In *CVPR*, pages 3169–3176, 2011.

[44] Limin Wang, Yuanjun Xiong, Zhe Wang, Yu Qiao, Dahua Lin, Xiaoou Tang, and Luc Van Gool. Temporal segment networks: Towards good practices for deep action recognition. In *ECCV*, volume 9912, pages 20–36, 2016.

[45] Xiaolong Wang, Ross Girshick, Abhinav Gupta, and Kaiming He. Non-local neural networks. In *CVPR*, pages 7794–7803, 2018.

[46] Yue Wang, Yongbin Sun, Ziwei Liu, Sanjay E. Sarma, Michael M. Bronstein, and Justin M. Solomon. Dynamic graph CNN for learning on point clouds. *ACM Trans. Graph.*, 38(5):146:1–146:12, 2019.

[47] Jim Winkens, Rudy Bunel, Abhijit Guha Roy, Robert Stanforth, Vivek Natarajan, Joseph R. Ledsam, Patricia MacWilliams, Pushmeet Kohli, Alan Karthikesalingam, Simon Kohl, A. Taylan Cemgil, S. M. Ali Eslami, and Olaf Ronneberger. Contrastive training for improved out-of-distribution detection. *CoRR*, abs/2007.05566, 2020.

[48] Kelvin Wong, Shenlong Wang, Mengye Ren, Ming Liang, and Raquel Urtasun. Identifying unknown instances for autonomous driving. In *CoRL*, volume 100, pages 384–393, 2019.

[49] Lu Xia, Chia-Chih Chen, and Jake K Aggarwal. View invariant human action recognition using histograms of 3d joints. In *CVPRW*, pages 20–27, 2012.

[50] Zhe Xue, Junping Du, Dawei Du, and Siwei Lyu. Deep low-rank subspace ensemble for multi-view clustering. *Inf. Sci.*, 482:210–227, 2019.

[51] He Zhang and Vishal M. Patel. Sparse representation-based open set recognition. *TPAMI*, 39(8):1690–1696, 2017.

[52] Yiru Zhao, Bing Deng, Chen Shen, Yao Liu, Hongtao Lu, and Xian-Sheng Hua. Spatio-temporal autoencoder for video anomaly detection. In *MM*, pages 1933–1941, 2017.