Complex Video Action Reasoning via Learnable Markov Logic Network

Yang Jin\textsuperscript{1,2}, Linchao Zhu\textsuperscript{3}, Yadong Mu\textsuperscript{*}
\textsuperscript{1}Peking University, \textsuperscript{2}Baidu Research, \textsuperscript{3}ReLER Lab, AAll, University of Technology Sydney
jiny@stu.pku.edu.cn, linchao.zhu@uts.edu.au, myd@pku.edu.cn

Abstract

Profiting from the advance of deep convolutional networks, current state-of-the-art video action recognition models have achieved remarkable progress. Nevertheless, most of existing models suffer from low interpretability of the predicted actions. Inspired by the observation that temporally-configured human-object interactions often serve as a key indicator of many actions, this work crafts an action reasoning framework that performs Markov Logic Network (MLN) based probabilistic logical inference. Crucially, we propose to encode an action by first-order logical rules that correspond to the temporal changes of visual relationships in videos. The main contributions of this work are two-fold: 1) Different from existing black-box models, the proposed model simultaneously implements the localization of temporal boundaries and the recognition of action categories by grounding the logical rules of MLN in videos. The weight associated with each such rule further provides an estimate of confidence. These collectively make our model more explainable and robust. 2) Instead of using hand-crafted logical rules in conventional MLN, we develop a data-driven instantiation of the MLN. In specific, a hybrid learning scheme is proposed. It combines MLN’s weight learning and reinforcement learning, using the former’s results as a self-critic for guiding the latter’s training. Additionally, by treating actions as logical predicates, the proposed framework can also be integrated with deep models for further performance boost. Comprehensive experiments on two complex video action datasets (Charades & CAD-120) clearly demonstrate the effectiveness and explainability of our proposed method.

1. Introduction

Action recognition is a fundamental task in video understanding and has garnered significant attention in the last few years. Recently, in virtue of the drastic development of deep learning, 3D convolutional networks (3D CNNs) have revolutionized this research field \cite{4,7,8,10,23}. With various elaborately-designed neural architectures and end-to-end learning algorithms, it has emerged as a prominent paradigm for video action recognition. Compared to early works \cite{21,33,55,56} based on low-level features (e.g., trajectories, key points), the powerful representation capability of 3D CNNs enables them to better capture complex long-range semantic dependencies across video frames.

Though extensively adopted in modern video action understanding tasks, these deep neural networks still suffer from some inherent deficiencies. Typically, 3D CNNs are fed a video clip and output a score that indicates the confidence for each action category through multi-layer calculations. Such a black-box predicting mechanism does not explicitly provide compelling evidence regarding the actions, such as when / where / why the action occurred. The lack of interpretability also makes deep neural networks vulnerable to adversarial attacks \cite{16,36}, which limits its applications in many real-world scenarios \cite{2} with strict security requirements. Therefore, in recent years, an increasing research effort has been devoted to explainable deep learning \cite{45,62}. All afore-mentioned facts strongly spur us to pursue an action reasoning framework with both accurate performance and convincing interpretability.

Figure 1. An illustration example from Action Genome \cite{22}. It demonstrates that actions can usually be decomposed into evolving spatio-temporal scene graphs (i.e., how a person interacts with surrounding objects over time such as \textit{person-lying-on-bed} to \textit{person-siting-on-bed}). Inspired by this, we propose to use a data-driven Markov Logic Network to model this evolving pattern.
Our motivation is also built upon some discovery from cognitive science and neuroscience [43,49] that people usually represent visual events as a composition of prototypical atomic unit. The research in [22] reveals that a complex action can be decomposed into spatio-temporal scene graphs, which depict how a person interacts with surrounding objects over time. Take action “awakening in bed” shown in Figure 1 as an example. To accomplish this action, a person may be initially lying on the bed, then wake up and sit on the bed. The procedure can be described by the temporal evolution of the human-object relationship, namely from \( \langle \text{person, lying on, bed} \rangle \) to \( \langle \text{person, sitting on, bed} \rangle \). This allows the model to explicitly recognize the occurrence of actions through detecting the transition of visual relationships, thereby its interpretability and robustness can be significantly improved. To implement this idea, we need to address two key challenges: automatically learning the temporally-evolving patterns from data instead of using hand-crafted rules, and conducting high-confidence inference under the noisy information in the real data that contaminates the aforementioned learned patterns.

To address the aforementioned issues, a novel explainable action reasoning framework is introduced to recognize actions in untrimmed videos. Specifically, we adopt first-order logic [1] for encoding the semantic-level state change of a complex action. At each logical rule, the visual relationships serve as atomic predicates. These rules contain adequate information and can be generated by a recurrent policy network from scratch. This procedure proceeds by progressively adding the action-related relationship predicates. Since these rules are generated in a data-driven fashion rather than by domain experts, they are prone to errors. To tackle this problem, we resort to Markov Logic Network (MLN) [44], a statistical relational model that combines first-order logic and probabilistic graphical models [30]. It associates a weight to each logical rule to soundly handle its uncertainty: the larger the weight, the more reliable the rule is. Hence, assigning lower (even negative) weights to noisy ones will alleviate their deficiency. Eventually, the probability of occurrence for each action is determined via conducting probabilistic logical reasoning on MLN.

The overall training scheme of our framework consists of two stages: rule exploration and weight learning. The first stage is accomplished by leveraging reinforcement learning. As for the second stage, the weight belonging to each rule can be updated via supervised learning (i.e., maximizing the likelihood of actions in the videos). Notably, the evaluation result from weight learning can be exploited as a critic criterion for guiding the rule exploration. The technical contributions of this work can be summarized as follows:

(1) Compared to the prevalent deep 3D convolutional networks, the proposed framework enjoys remarkable interpretability since the weighted logical rules can convey clear evidence regarding specific action. Moreover, our framework naturally supports simultaneously recognizing the categories of actions and localizing their temporal boundaries, benefiting from the learned temporal-evolution patterns.

(2) The logical rules for encoding complex actions can be automatically exploited from data via our proposed rule exploration mechanism, which is superior to some earlier approaches [3,35,54,70] that relied on manually-designed rules to perform action reasoning.

(3) Comprehensive experiments on two challenging video benchmarks (Charades [47] and CAD-120 [31]) show that our method obtained excellent performance. In addition, it can furthermore boost the accuracy when being integrated with deep models. Surprisingly, our framework are still capable of achieving outstanding performance only leveraging limited number of training examples.

2. Related Work

Video action recognition. Human action understanding and analysis has been an active research area over the past decades. Thanks to the emergence of deep learning, especially for the engineering tailoring of convolutional neural networks (CNNs) [32], significant development has been made in action recognition. For example, two-stream approaches like [12,48] read the RGB and optical flows as input and process them separately in different branches of networks, which surpasses previous works by a large margin. The prevalence of 3D-CNNs [4,17,52] makes them become the mainstream paradigm in this area. A majority of works [5,9,11] mainly focus on designing effective neural architectures to extract rich spatio-temporal information from videos. One related work in [60] also adopts a graphical structure to exploit the implicit relationships between object region proposals in videos and perform reasoning on it via Graph Convolutional Networks [29]. Unlike them, we adopt the weighted logical formulae to explicitly encode the visual relationships and leverage MLN to handle the uncertainty, which contributes to remedying the low interpretability of deep models.

Probabilistic logic reasoning. This research field [6,13], aims to integrate probabilistic reasoning with first-order logic and machine learning. First-order logic rules can systematically generalize the domain knowledge and thus have been widely adopted for reasoning, such as expert systems [61]. Due to the hard constraint of logic, researchers attempted to integrate it with probability, which led to the development of approaches based on graphical models in recent years, including Bayesian logic programs [27], Markov logic networks [44] and others. They have been utilized for human activity recognition in early works [35,54]. For example, Liao et al. [34] performed probabilistic inference on an unrolled Markov network based on the information about locations provided by GPS sensors. In [37], the
authors incorporated the pre-defined knowledge (e.g., the trajectories of players and objects) into the Markov logic network and performed multi-agent event recognition.

Although these related works take advantage of rule-based knowledge to recognize video events, their explainability is still limited due to the inadequate representation capacity of low-level features. In addition, the rules for encoding actions need elaborate labeling by domain experts. Instead, our proposed method adopts high-order visual relationship as the prototypical unit and automatically mines rules from the video data, which enables to subtly capture the semantic information of complex events without burdensome manual labor.

**Scene graph generation.** Scene graph [25] is a structural representation for understanding visual content in static images, where each unique object defines a node, and the relationship between two objects corresponds to an edge. Owning to the potential of enhancing many down-stream visual reasoning tasks [24,65], this task has attracted tremendous attention from researchers. By harnessing the message passing mechanism [63], recent methods [51,64,67] are capable of fully exploiting the global visual context and predicting satisfactory scene graphs. In this work, we apply it to the video domain and generate a spatio-temporal scene graph in video segments to represent the semantic information of a complex event without burdensome manual labor.

### 3. Preliminary: Markov Logic Network

Markov Logic Network (MLN) [44] is a statistical relational model that utilizes first-order logic to define potential functions in the Markov random field. In MLN, each logic formula has an associated real-value weight, which indicates its confidence score. Higher weight is favored for accurate formula. Essentially, MLN softens the hard constraints of first-order logic, making states that violate some of the formulae less probable but not impossible.

Formally, let $\mathcal{F}$ be a logic formulae set, $\omega_i$ be the weight with respect to formula $f_i \in \mathcal{F}$ and $\mathcal{C} = \{c_1,c_2,...,c_{|\mathcal{C}|}\}$ be a finite set of constants. Then, MLN serves as a template for constructing a Markov network $M_{\mathcal{F},\mathcal{C}}$, where each possible grounding of an atomic predicate in $f_i$ can be seen as a binary node that takes value 1 if that grounded predicate is true, and 0 otherwise. Each possible grounding of formula $f_i$ is a potential function whose value is 1 if the ground formula is true and 0 otherwise. Hence there is an edge between two nodes in $M_{\mathcal{F},\mathcal{C}}$ if their grounded predicates appear simultaneously in one formula grounding. Given such formulation, the probability distribution over a world $x$ is given as below:

$$P(X = x) = \frac{1}{Z} \exp \left( \sum_{i \in \mathcal{F}} \omega_i n_i(x) \right),$$

where $n_i(x)$ is the number of true groundings of formula $f_i$ in $x$, $F$ is the size of formula set $\mathcal{F}$, and $Z$ is the normalizing partition function given by $\sum_x \exp \left( \sum_i \omega_i n_i(x) \right)$. See [44] for more details.

### 4. The Proposed Approach

In this section, we present the technical details of our methods. As previously mentioned, complex actions can usually be decomposed into temporal transitions of human-
object interactions across video frames. Inspired by this observation, we develop an explainable reasoning framework in accordance with the evolving pattern of visual relationships such as \( \langle \text{person-lying on-bed} \rangle \) to \( \langle \text{person-sitting on-bed} \rangle \) for complex action recognition.

As illustrated in Figure 2, the proposed approach consists of two main components. The first one is a rule policy network that aims to generate a near-optimal formulae set \( \mathcal{F} \), where each formula \( f \in \mathcal{F} \) explicitly represents a specific transition pattern. The other one is an action reasoning module that performs probabilistic logic reasoning for calculating the probability of each action through a Markov logic network [44], which is constructed according to the evolving pattern of visual relationships such as \( \{R_t\} \ni \mathcal{F} \) before. Next, we will expound the implementation details for each component and the corresponding training algorithm for the overall framework.

4.1. Rule Policy Network

Unlike early works with hand-crafted logical formulae [34, 37], we aim to automatically produce formulae tailored to each interested action without relying on any human labor. In this work, we specify the evolving pattern of human-object interaction pattern with the logic form: \( R_1 \land ... \land R_T \), where \( R_{i:T} \) denotes the relationship predicates in different frame and \( T \) denotes the total number of these predicates. Then the formula \( f \) with respect to a complex action \( a \) can be represented as:

\[
\bigwedge_{t=1}^{T} R_t \rightarrow A \quad \text{or} \quad \bigwedge_{t=1}^{T} R_t \Leftrightarrow A,
\]

(2)

where \( A \) is the predicate form of action \( a \). Given a specific action predicate \( A \), only the left part in \( f \) remains to be specified. Since \( \bigwedge_{t=1}^{T} R_t \) in Eq. 2 only contains the conjunction operation \( \land \), it can be further represented as a linear sequence \( l_f = \{R_t\}_{t=1}^T \).

Relying on the above transformation, the generation of \( f \) turns into a sequential decision process with the goal of predicting the most suitable \( l_f \) for each action. We model this process using a policy network \( \pi \), which is trained to approximate the probability distribution \( \pi(f|a; \theta) \) over all possible formula \( f \) with respect to \( a \). Here \( \theta \) is the distribution parameter. Once \( \theta \) is determined, we can accordingly draw several samples from \( \pi(f|a; \theta) \) to harvest the formulae set \( \mathcal{F} \). To this end, \( \pi \) is fulfilled by a Gated Recurrent Unit network which can be formulated as:

\[
h_t = \text{GRU}(x_t, h_{t-1}),
\]

(3)

where \( x_t \) is the embedding feature for predicate \( R_t \) at \( t \)-th step, \( h_{t-1} \) denotes the hidden state maintained within \( \pi \) that aggregates the information of all past predicates \( \{R_1, ..., R_{t-1}\} \). At the initial step, input the feature vector \( x_0 \) of action predicate \( A \) to \( \pi \), then the probability for generating each predicate \( R_t \) is computed by:

\[
p(R_t|R_1, ..., R_{t-1}, A) = \text{softmax}(W_p h_t), \tag{4}
\]

where \( W_p \) is the parameter to be learned from data. During training, we can obtain a formula \( f \) by sampling a corresponding sequence \( l_f = \{R_t\}_{t=1}^T \) in terms of the distribution in Eq. 4. Hence, the probability of the formula \( f \) is:

\[
p(f|A) = \prod_{R_t \in l_f} p(R_t|R_1, ..., R_{t-1}, A). \tag{5}
\]

After training the policy network \( \pi \), we leverage the beam search strategy to sample \( k \) best sequences from \( \pi(f|a; \theta) \) for each action \( a \) as the learned formulae set \( \mathcal{F} \).

4.2. Probabilistic Action Reasoning

This section presents the detailed probabilistic reasoning procedure for action recognition. The reasoning module mainly contains three steps (see Figure 2). Next, we will describe them respectively in the following.

Snippet generation with sliding window. Given an untrimmed video denoted by \( v \), a sliding window mechanism is first applied to \( v \) for generating several video snippets. In view of the fact that different actions often exhibit large variation in temporal duration, the kernel of our sliding window is set to multiple sizes. Moreover, for a sliding window with kernel size \( L \), each snippet has \( L/2 \) frames overlapped with its neighbours. The sampled snippet set, denoted as \( U \), serves as the temporal proposals for the underlying actions within the video \( v \).

Scene graph prediction. For each snippet \( u \in U \), we employ a pretrained scene graph predictor to exploit the high-level visual information belonging to a video frame. Specifically, the predictor extracts all the objects in a frame and forecasts their visual relationships with the actor. The generated scene graph can be denoted as \( G = (O, E) \). Here, \( O = \{o_1, o_2, ...\} \) is the set of objects interacting with an actor \( p \) and \( E = \{\{e_{11}, e_{12}, ...\}, \{e_{21}, e_{22}, ...\}\} \) denotes the relationships between them, where \( e_{ij} \) indicates the \( j \)-th relationship between the actor \( p \) and \( i \)-th object \( o_i \). There may exist multiple types of relationship between each actor and object due to the diversity of visual interaction. Note that each triplet \( r_{ij} = (p, e_{ij}, o) \) can be treated as a grounding of its corresponding relationship predicate on a video snippet. Moreover, the confidence score \( s_{r_{ij}} \) of the grounding \( r_{ij} \) is given by:

\[
s_{r_{ij}} = s_p \cdot s_{e_{ij}} \cdot s_{o_i}, \tag{6}
\]

Here, \( s_p, s_{o_i}, s_{e_{ij}} \) are respectively the confidence scores for the predicted actor \( p \), object \( o_i \) and their relationship \( e_{ij} \), which are given by the scene graph predictor.
Considering that visual relationships among objects barely change in a few consecutive frames, it would be redundant if we generate scene graph for every single frame in a snippet. Hence, only $M$ frames are uniformly sampled from a snippet $u \in U$ to perform the above prediction.

**Probability inference.** Given a trained Markov network \( \mathcal{M} = \{(f_i, \omega_i)\}_{i=1}^{P} \), the probability of each action \( a \) on a video can be accordingly inferred. To this end, following Eq. 1, the number of true grounding \( n_i(x) \) on snippet \( u \) with respect to formula \( f_i \) requires to be determined. Note that the logic formula in the MLN operates on binary predicates, which can only take a value of 0 or 1. However, the grounding of our relationship predicate takes a real value specified in Eq. 6 with the range of \([0,1]\). Such a property makes it difficult to determine whether a formula grounding is absolutely true.

To ensure compatibility with logical operations (e.g., \( \lor, \land, \neg \)) in first-order logic, we relax the operations on continuous variables using \( \check{\text{L}}\)ukasiewicz logic \([14]\). The relaxed conjunction (\( \land \)), disjunction (\( \lor \)) and negation (\( \neg \)) can be defined as: \( X \land Y = \max(0, X + Y - 1) \), \( X \lor Y = \min(1, X + Y) \) and \( \neg X = 1 - X \). With such a formulation, the \( n_i(x) \) in Eq. 1 can be effectively computed. Take the formula on the left part of Eq. 2 as an example. According to the transform criterion in first-order logic, such a formula can be firstly converted into a Horn Clause \([19]\):

\[
\bigwedge_{t=1}^{T} R_t \rightarrow A \iff \bigwedge_{t=1}^{T} \neg R_t \lor A, \tag{7}
\]

which are the disjunctions of positive or negated literals. Then, based on the predicted scene graph on \( u \), the value of each grounding \( f_i(x) \) is:

\[
f_i(x) = \min \left( \sum_{t=1}^{T} (1 - s_{r_t}) + x_a, 1 \right), \tag{8}
\]

where \( s_{r_t} \) is the confidence score obtained by Eq. 6. \( x_a \) is a binary variable with a value of 0 or 1 that indicates whether an action \( a \) occurs. \( n_i(x) \) is thus obtained by adding up the value \( f_i(x) \) of all groundings. After that, the probability of action \( a \) on a video snippet is given by:

\[
P(a = x_a | \text{MB}_x(a)) = \frac{\exp \left( \sum_{i=1}^{P} \omega_i n_i(x_{[a=x_a]}) \right)}{\exp \left( \sum_{i=1}^{P} \omega_i n_i(x_{[a=0]}) \right) + \exp \left( \sum_{i=1}^{P} \omega_i n_i(x_{[a=1]}) \right)}, \tag{9}
\]

where \( P \) is the number of formulae related to \( a \), \( \text{MB}_x(a) \) indicates the Markov blanket of \( a \), which are the triplets that appear together with \( a \) among all formulae. The final results of the whole video \( v \) are obtained by performing a max-pooling on its snippet set \( U \).

**4.3. Hybrid Training Algorithm**

Our goal is to learn the most suitable Markov network \( \mathcal{M} = \{(f_i, \omega_i)\}_{i=1}^{P} \) from the training data. For this purpose, the training scheme consists of two main stages: \textit{rule exploration} and \textit{weight learning}. Due to the discrete nature, one cannot directly learn the policy network \( \pi \) through back-propagation based on the end-task loss. Thus we propose to use a hybrid learning strategy in which the rule exploration stage is optimized by the policy gradient method in reinforcement learning, and the weights of generated rules are optimized via supervised learning.

Suppose we obtain a formula \( f \) by sampling from \( \pi(f|a; \theta) \), then we can train the rule policy network via maximizing the expected reward:

\[
J(\theta) = E_{f \sim \pi(f|a; \theta)} [H(f)]. \tag{10}
\]

Here \( H(f) \) is the recognition performance evaluation metric such as mAP. Then, the gradient \( \nabla_{\theta} J \) will be formulated as: \( E_{f \sim \pi(f|a; \theta)} [H(f) \nabla_{\theta} \log \pi(f|a; \theta)] \), which can be estimated by Monte-Carlo sampling:

\[
\nabla_{\theta} J \approx \frac{1}{K} \sum_{k=1}^{K} (H(f_k) \nabla_{\theta} \log \pi(f_k|a; \theta)), \tag{11}
\]

where \( K \) is the sampling times. Inspired by \([42]\), we introduce a baseline \( b \), which is the exponential moving average of recent \( H(f_k) \). The original reward in Eq. 11 is then replaced by \( H(f_k) - b \). Moreover, to encourage the diversity of rule exploration, we also add an entropy regularization over \( \pi(f|a; \theta) \) to the final loss.

The weight learning stage aims to learn the appropriate weight for the generated formula, which is fulfilled by maximizing the log-likelihood:

\[
\mathcal{L}(f) = \sum_{i=1}^{N} \log (P_i(a = x_a | \text{MB}_x(a))), \tag{12}
\]

where \( N \) is the size of a batch of videos, \( x_a = 1 \) if the action \( a \) exists in the \( i \)-th video \( v_i \) and 0 otherwise.

The overall training procedure will be alternatively executed between \textit{rule exploration} and \textit{weight learning}. Firstly, we perform \textit{weight training} for the formula set \( F \) generated via the initialized rule policy network \( \pi \), and then fix the weight to update the parameter of \( \pi \) based on the gradient estimated by Eq. 11. After that, we perform weight training for a fresh \( F \) generated by updated \( \pi \). These two stages will be performed alternatively for several times.

**4.4. Integration with Deep Model**

An untrimmed video usually involves multiple actions, among which some underlying relations may exist. Take a video instance in Charades \([47]\) as an example, there are
some reasonable connections among the actions holding a broom, putting a broom somewhere and tidying something on the floor: when a person is tidying something on the floor, he may be holding a broom and then put the broom back after the tidying. Therefore, our proposed framework can be incorporated as an inference layer after the output of deep models to enhance the prediction for hard-to-detect actions (e.g., tidying something on the floor), based on easy-to-detect actions (e.g., holding a broom). Specially, our framework can be leveraged to learn some logical formulae and the corresponding weights to represent the connections among actions. During inference, given the output confidence scores from a deep model, we consider the actions with high confidence as the observed evidence and perform probabilistic reasoning for other actions.

5. Experiment

5.1. Datasets and Metrics

Datasets. Two large-scale video datasets are utilized in the whole experiments. (1) Charades [47]. It is a large dataset composed of about 9.8k untrimmed videos, among which 7,985 are used for training and 1,863 for testing. These videos contain 157 complex daily activities about 267 people’s 15 types of indoor scenes. On average, each video contains 6.8 distinct action categories, often with multiple ones in the same frame, which makes the recognition extremely challenging. To train the scene graph predictor, we leverage the Action Genome [22], which provides frame-level relation annotations for videos in Charades. Overall, it includes 1.7M instances of 25 relationship classes. (2) CAD-120 [31]. This is an RGB-D dataset focusing on the human activity of daily life. It consists of 551 video clips with 32,327 frames about 10 different high-level activities (such as having meal, arranging objects). Here, we adopt a re-annotated version provided by [70], which includes detailed relationships and attributes for the video frames.

Evaluation protocol. For Charades, our goal is to recognize multiple complex actions in an untrimmed video. Due to the multi-label property, we calculate the Mean Average Precision (mAP) to evaluate the performance on all categories. While for CAD-120, the Mean Average Recall (mAR) metric is adopted as in [70] to measure whether the model successfully recognizes the performed actions.

5.2. Implementation Details

We firstly train a scene graph detector to generate the scene graph for video frames. To fulfill this, a Faster RCNN [41] detector with ResNet-101 [18] backbone is applied to extract a 2,048-dimensional RoI (region-of-interest) feature for each detected objects. Then, we employ Motifs [50, 66] to perform relation prediction, which is trained on Action Genome by following the train / validation splits as Charades. For the rule policy network, we utilize a gated recurrent unit (GRU) with 512 hidden units and project the logic predicate to a 200-dimensional vector by simply averaging its word embedding [38].

Before the hybrid training, we perform a warm-up pre-training for the policy network. It can be done by randomly sampling some relationship transition sequences from the training data and leveraging them as supervision to guide our rule policy. Through this procedure, it learned a suitable parameter initialization which serves as a frequency-related prior and enables our hybrid training to converge faster. The pre-training proceeds three epochs with a learning rate \( lr = 0.001 \) and uses a cross-entropy loss. After that, we conduct our hybrid training to update the policy network. In detail, we optimize it using an Adam optimizer [28] with \( \beta_1 = 0.9, \beta_2 = 0.999, lr = 0.0005 \) and set \( K = 5 \) in Eq. 11. The weight learning is fulfilled via maximizing the log-likelihood in mini-batch data, where the batch size is set to 256.

5.3. Main Results

To fully demonstrate the superiority of our proposed model, we design two key experimental settings on aforementioned video datasets, including action recognition and action temporal localization.

5.3.1 Complex action Recognition

This task requires the model to predict a video-level action labels as the final recognition results. We adopt the ResNet-101 [18] as the backbone for our scene graph predictor and compare with several recent competitive methods (note that the mAP scores using K400-pretrained backbone are supposedly over-rated since the action categories in Kinetics and Charades are partially overlapped). Table 1 summarizes the results on Charades. It can be seen that our model achieves 38.4% mAP and surpasses the powerful 3D CNN
models, which demonstrate that our model can fully exploit temporal information via the generated formula and their MLN weight, on the basis of just utilizing 2D scene graph on individual video frames (rather than more informative short snippets as in I3D). Benifiting from the pre-training on large video benchmark Kinetics [26], the state-of-the-art 3D models (e.g., X3D) achieves higher performance than our model, but our method exceeds deep models only pre-trained on ImageNet (38.4 % v.s. 21.0 % in [15]). Due to the limitations of scene graph predictor, we follow [22] and design an Oracle version of our method, which leverages the ground-truth of relationships on a frame. As presented in the bottom line of Table 1, our Oracle version achieves a significant improvement (24%) on mAP performance and outstrips all the deep models by a large margin, which demonstrates the powerful potential of our method. We also evaluate the model integration (Section 4.4) with SlowFast (R-50). By exploiting the relation between different actions, our model can further boost the performance of deep models (1.3% higher mAP in Table 1).

For CAD-120 dataset, we follow the same setting as in [70] to divide the long video sequences into small clips, each of which only contains one action, and evaluate the Average Recall metric for each action. As shown in Table 2, our model achieves the best results in terms of mAR. Although [70] also adopted an explainable framework, they performed action reasoning just by observing specific state transitions between two consecutive frames defined by domain expert. Our model leverages MLN learned from real data, which is more general and excellent (0.83 v.s. 0.80).

5.3.2 Action Temporal localization

Our model recognizes complex actions by relying on explainable formula, and thus provides convincing evidence that shows the reason to make such prediction. Therefore, by knowing the timestamp where these evidence appears, one can localize the temporal boundary of the action. It is fulfilled by firstly leveraging the sliding window mechanism described before to generate several video snippets from the whole video. Then we perform action reasoning on each snippet and choose the one with highest probability as the temporal location of the corresponding action.

We compare with several advanced deep models on the Charades. It can be seen in Table 3 that our model achieves a prominent action localization results. Compared with models [46, 57] that also are pre-trained only on ImageNet, we have the best performance (20.9% mAP v.s. 14.2% mAP by [57]). In addition, we still achieve a comparable results with ones pre-trained on Kinetics (e.g., [39]). Despite slightly weaker than [10] in mAP performance, our localization prediction is more explainable. Since no ground truth is provided in CAD-120, we did not report results on it.

<table>
<thead>
<tr>
<th>Methods Modality Pre-train</th>
<th>mAP(%)</th>
</tr>
</thead>
<tbody>
<tr>
<td>I3D [4] RGB K-400</td>
<td>15.6</td>
</tr>
<tr>
<td>3D ResNet-50 [53] RGB K-400</td>
<td>18.6</td>
</tr>
<tr>
<td>X3D [8] RGB K-400</td>
<td>18.9</td>
</tr>
<tr>
<td>I3D + SP [40] RGB + Flow K-400</td>
<td>19.4</td>
</tr>
<tr>
<td>X3D-L [8] RGB K-400</td>
<td>20.0</td>
</tr>
<tr>
<td>SlowFast det (X3D) [10] RGB K-400</td>
<td>22.3</td>
</tr>
<tr>
<td>Ours RGB ImageNet</td>
<td>20.9</td>
</tr>
</tbody>
</table>

5.4. Ablation Studies

Module combinations. To explore the effect of our rule policy network (RPN) and the weight learning (WL) in probabilistic action reasoning module, we conduct some related ablation studies. To be specific, we propose three adjustments, 1) replacing the rule policy network with a frequency-induced baseline that generate the formula according to the co-occurrence frequency of relationships in training set. 2) leveraging the formula produced by our rule policy network and directly adopt the probability in Eq. 5 as the final weight for MLN reasoning. 3) directly using the formula generated from the frequency-induced baseline and treating the frequency value as the weight without additional learning. The quantitative results are shown in Table 4. One can observe that cancelling any of our two key modules will weaken the recognition performance. Besides, the rule policy network contributes more to the whole performance compared with weight learning (5.3% mAP decrease v.s. 8.6% mAP decrease on Charades), which demonstrates the effectiveness of exploiting suitable formulæ from real video data.

Different amounts of training data. Intuitively, the human-object interaction pattern belongs to the same action should be similar among different videos. Therefore, one can learn this specific pattern via just several examples. To validate this assumption, we conduct an experiment on Charades to explore the recognition performance under different numbers of training examples. To be specific, we train our model with only k positive examples of each action category. The results are reported in Table 5. As expected, our
Figure 3. Some examples of the learned formula and corresponding weights by the proposed hybrid training algorithm. Each formula can be drawn from $\pi(f | a; \theta)$ with the corresponding weight. We highlight positively-supporting formula in green, otherwise in yellow.

Figure 4. Statistics of user study regarding the explainability (i.e., being human-friendly) of the learned formulae. The candidates are categories as good, neutral or bad according to their weights. Each row depicts the confusion with other categories aggregated from all sampled rules.

6. Concluding Remarks

We propose an explainable action reasoning framework for complex video action recognition. Inspired by the fact that complex actions can be decomposed into prototypical atomic unit like scene graph, we perform the probabilistic logical inference based on Markov Logic Network (MLN). The formulae used for reasoning are all learned automatically from the data. Different from existing approaches based on black-box deep convolutional networks, our model is capable of explaining when / where / why an action occurs in the video. Extensive experiments and visualization both confirm the effectiveness and interpretability.

Acknowledgement: This work is supported by National Key R&D Program of China (2018AAA0100702) and Beijing Natural Science Foundation (Z190001).
References


