Received: 5 June 2021 | Revised: 29 September 2021 | Accepted: 8 November 2021

DOI: 10.1002/ail2.59

WILEY

LETTER

Explainable activity recognition in videos: Lessons learned

Chiradeep Roy[1] | **Mahsan Nourani**[2] | **Donald R. Honeycutt**[2] | **Jeremy E. Block**[2] | **Tahrima Rahman**[1] | **Eric D. Ragan**[2] | **Nicholas Ruozzi**[1] | **Vibhav Gogate**[1]

[1]Department of Computer Science, The University of Texas at Dallas, Richardson, Texas, USA

[2]Department of Computer and Information Science and Engineering, University of Florida, Gainesville, Florida, USA

Correspondence
Chiradeep Roy, Department of Computer Science, The University of Texas at Dallas, Richardson, TX, USA.
Email: chiradeep.roy@utdallas.edu

Funding information
DARPA XAI Program, Grant/Award Number: N66001-17-2-4032; National Science Foundation, Grant/Award Numbers: IIS-1528037, IIS-1652835

Abstract

We consider the following activity recognition task: given a video, infer the set of activities being performed in the video and assign each frame to an activity. This task can be solved using modern deep learning architectures based on neural networks or conventional classifiers such as linear models and decision trees. While neural networks exhibit superior predictive performance as compared with decision trees and linear models, they are also uninterpretable and less explainable. We address this *accuracy-explanability gap* using a novel framework that feeds the output of a deep neural network to an interpretable, tractable probabilistic model called dynamic cutset networks, and performs joint reasoning over the two to answer questions. The neural network helps achieve high accuracy while dynamic cutset networks because of their polytime probabilistic reasoning capabilities make the system more explainable. We demonstrate the efficacy of our approach by using it to build three prototype systems that solve human-machine tasks having varying levels of difficulty using cooking videos as an accessible domain. We describe high-level technical details and key lessons learned in our human subjects evaluations of these systems.

KEYWORDS

cutset networks, debugging machine learning models, human-computer interaction, temporal and time-series models, tractable probabilistic models, video activity recognition

1 | INTRODUCTION

Activity recognition in video, the task of inferring and assigning a predefined set of activities to frames or segments of a given video, is an important subtask in *video content analysis* with a myriad of applications including video surveillance, video summarization, improving situational awareness (detecting dangerous situations), and home security. While this task is hard in general, today, with advances in deep learning, especially given the remarkable predictive performance of deep models, the task can be solved accurately when ample labeled (videos) training data is available. Unfortunately, despite high accuracy, deep neural models are black boxes: it is difficult to explain the reasoning behind their decisions and answers. This lack of explanability is often undesirable, especially for solving interactive human-machine tasks[1] where the user makes decisions with the aid of a machine learning (ML) system. In such cases, the users need to trust

the predictions of the system and be able to easily determine when the ML system is (typically) correct and when it is not. To facilitate this, the system should be such that both its functioning and the reasoning behind its actions are clear to the users, that is, the system should be explainable.

The purpose of this paper is to describe a probabilistic modeling framework for activity recognition that is accurate yet interpretable and explainable, and to show that it helps improve the users' trust and understanding of the system. We define "explainable" here as the ability of our system to be able to compute the answers to probabilistic queries posed over videos and provide justifications to its answers in the form of explanations (similar to the notion of explainability in the probabilistic programming literature[2-4]). Since the explanations are generated directly from our joint model, they have high fidelity and can be used by end-users to build a mental model of the system's functioning by posing multiple queries to it.

At a high level, our model (referred to as "the model" in the remainder of the paper) has two layers. The top layer, with which a user interacts, is a tractable, interpretable probabilistic model, specifically a cutset network.[5] Unlike conventional graphical models such as Bayesian and Markov networks in which probabilistic inference is NP-hard in general and inaccurate in practice, cutset networks have two desirable properties. First, they are tractable in that they provide linear-time reasoning capabilities. Second, their predictive performance is substantially superior to Bayesian and Markov networks.[5-8] While all probabilistic models that represent a joint probability distribution over the variables of interest are capable of generating explanations, under the assumption that an explanation is a query over the model, the use of tractable cutset networks ensures that explanations can be computed both accurately and efficiently. The bottom layer of our model is a deep neural network that yields accurate estimates, which are fed into the cutset network layer. A possible interpretation of this model is that the deep learning layer provides noisy sensory inputs to the cutset network layer, which in turn, removes the noise, makes decisions and generates explanations (for the decisions) by performing fast, accurate inference. To model temporal aspects in video, we further refine this model, propose a novel temporal probabilistic modeling framework called dynamic cutset networks, and show that it improves the estimation accuracy.

We demonstrate the efficacy of our two-layer model by using it to build an *explainable activity recognition system* for an accessible domain—*cooking videos* given in the Textually Annotated Cooking Scenes (TACoS) dataset.[9] Our system is set up as a visual question-answering system where the user can ask the system a series of questions (eg, *"Did the person cut a carrot?"*) and the system provides answers to questions posed as probabilistic queries (eg, *"Yes"*) as well as explanations for these answers (eg, *"the activity took place between 00:12 to 00:45 with a probability of 0.72"*). The system provides three types of explanations that are relevant to the question and its answer: (a) visual explanations which show video segments; (b) ranked explanations which show top-3 combinations of objects, locations and actions; and (c) marginal explanations which show the confidence in the detected objects, locations, and actions. Note that the goal of this framework is *not* to provide low-level pixel explanations for the deep learning model; rather, it is to build an activity recognition system that is explainable *as a whole* by generating high-level explanations for each probabilistic query at the level of label probabilities. To this end, we hypothesize that the three kinds of explanations that can be generated from our two-layer model for each probabilistic query should give end users a good understanding of the strengths and the weaknesses of the *entire* system. For example, end users might notice that the system is able to accurately predict an orange only when a knife and cutting board are present in the given frame. These kinds of insights are either absent or very difficult to extract from pixel-level explanations.

We evaluated our system by using it to solve three human-machine tasks having varying levels of difficulty. The three tasks were: (a) answering whether an activity was performed in a video by watching the video (yes/no questions); (b) answering whether a set of policies were followed or not in a collection of videos; and (c) debugging the model to find its error patterns. We built novel interactive visual interfaces for each task and conducted human subjects studies. Our studies showed that when the underlying (human-machine) task is relatively easy (yes/no questions) and the model is accurate, explanations helped in improving time to completion, accuracy and trust in the system. However, on more involved tasks such as verifying policies in a set of videos, we found that the effects of explanations are not universally beneficial. Specifically, if first impressions of system outputs are skewed toward either system strengths or system weaknesses it can significantly affect users' trust, reliance, and (mis-)understanding of the ML model, regardless of the presence of explanations. In addition, we show how the explanations generated by our system can be used in a visual analytics tool to support model debugging through identification of both instance-level problems as well as global error patterns.

2 | RELATED WORK

2.1 | Hidden Markov models and dynamic Bayesian networks

There has been significant work in the past that have used state-space models such as hidden Markov models (HMMs)[10-13] and dynamic Bayesian networks (DBNs)[14-17] on top of low-level feature extractors to model persistence and dynamics. However, HMMs typically make a lot of restrictive assumptions and inference complexity increases exponentially in the size of the state space. The expressive power of DBNs, on the other hand, are offset by exact inference being NP-hard[18] in these networks. The model we use in this paper aims to bridge this expressiveness-tractability gap by extending cutset networks[5] into the temporal domain (much like DBNs extend Bayesian networks into the temporal domain). This formulation will help us efficiently compute posterior probabilities which can be used as explanations for queries (which we explain in the later sections).

2.2 | Models that use black boxes as sensors

There has been research in the past that has attempted to combine black-box models with probabilistic models in various ways. Some of the earlier attempts were the work of Pei et al,[19] Morariu et al,[20] and Brendel et al.[21] The work by Pei et al is a grammar-based approach[22-25] that uses an AND/OR graph to model the semantics of the video.[26] While this approach has the advantage of providing a global overview of the functioning of the model (ie, interpretability), it is unable to efficiently compute action-level conditional probabilities over time such as the probability that a cutting activity over a carrot would take place in the sixth interval given that a picking up knife activity took place in the first. It can only do this after enumerating all possible combination of missing activities (ie, inference is NP-hard). The latter works by Morariu et al and Brendel et al use probabilistic relational logic to refine the probabilities of candidate event hypotheses. Although probabilistic relational models work very well in situations where prior knowledge about the domain is known a priori, learning these relations from data is typically NP-hard.[27] Furthermore, inference is also typically intractable without making some very restrictive assumptions (eg, the work by Morariu restricts the treewidth of the MLN to make exact inference tractable). The model we propose in this paper addresses all of these problems by formulating as a temporal multi-label classification problem where the relations between labels are encoded into a compact representation of the joint probability distribution. Furthermore, using tractable prior and transition distributions in our dynamic conditional cutset network (DCCN) framework allows for both efficient and accurate posterior estimates using particle filtering.

2.3 | Explainable systems and trust

There have been a number of studies on how trust influences interactions between humans and automated systems, for example, Muir,[28] Muir et al,[29] Lee et al,[30] and Hoffman et al.[31] These studies examine factors that might affect the trust of the user in the system, such as showing the past performance of the system and making the working of the system more understandable.[30] The work by Hoffman et al[32] provides a more detailed taxonomy of such factors and explains how trust is context-specific and dynamic. In other words, trust might vary with respect to specific contexts of automation and must also be maintained over time. Our aim is to be able to measure and control user trust with respect to these systems in order to better understand what kind of explanations influence the trust variable.

2.4 | Activity recognition and NLP

This work is closely related to the work of Rohrbach[33] and Donahue[34] on generating a semantic representation from videos at an activity level using deep learning architectures. Instead of generating sentences in natural language, however, we assign a number of predefined labels divided into categories. Related efforts have considered the task of dense captioning,[35] that is, generating summaries of texts from particular segments. Song et al[36] attempted to create

captioning methods that require minimum supervision on the TACoS dataset. Duan et al[37] attempted to combine caption generation and sentence localization to feed off of each other to create a weakly supervised training model. While these works focus on constructing text summaries, our work is different in that it aims to create a semantic representation for activities in each frame that can be used to both answer queries easily as well as generate explanations (via probabilistic inference) that justify these answers.

3 | XAI SYSTEM DESCRIPTION

3.1 | The explainable activity recognition task for cooking videos

We evaluated and tailored our system to the TACoS dataset.[33] Each frame in each video in this dataset is labeled with an (*action, object, location*) triple; this triple defines an *activity*. The *action* component forms the core part of the activity. These are usually verbs like wash, cut, slice, open, and so on. The *object* component denotes the entities over which the activity is performed. These are generally nouns such as apples, refrigerator, cutting board, knife, and so on. Finally, the *location* component tells us *where* the activity is taking place. These are generally location nouns such as kitchen, counter top, sink, and so on, but can also overlap with the nouns we use as objects. The dataset has 28 labels (our vocabulary) which includes 12 actions, 7 objects, 8 locations and a special label called "Nothing" or "None."

Users interact with our system by posing so-called *selection questions*: "Did a particular activity defined by the triple (*action*, *object*, *location*) happen in the video?" where object and location can be "None," but action is not allowed to be "None." Examples of selection questions include: (a) "Did the person slice an orange on the counter?" where slice, orange, and counter denote the action, object, and location, respectively; and (b) "Did the person take out grapes from the refrigerator?" where take out, grapes, and refrigerator denote the action, object, and location, respectively.

Our goal is to build an explainable system that provides three types of explanations following an answer to a selection question:

1. *Video explanations*: When the system answers "yes," we want the system to highlight (possibly more than one) segments of the video where the activity happened. For "no" answers, we want the system to highlight segments where a related activity happened, for example, carrots were cut in the video but not oranges. If no related activity is found in case of a "no answer," we want the system to output the most likely activity in the video.
2. *Ranked (action, object, location) triples*: We want the system to display the top- k predicted activity triples in the video that are relevant to the query.
3. *Most probable entities*: We want the system to display the most probable actions, objects and locations, along with their likelihood that are relevant to the query.

3.2 | System architecture

Figure 1 shows a high-level overview of the components of the system and the processing pipeline. Roughly speaking, the system comprises the following two layers of models: (a) *video classification layer* which takes as input video frames and a vocabulary file and assigns a set of labels from the vocabulary to each frame; and (b) *explanation layer* which takes the predicted labels from the video classification layer as input, corrects them using a probabilistic model, and outputs (potentially more accurate) labels and explanations.

3.3 | Video classification layer

For this layer, we used GoogLeNet,[38] a 22-layer neural network that is pretrained on the ImageNet dataset.[39] To tailor GoogLeNet to our video dataset which has 28 labels, we replaced the topmost softmax layer in GoogLeNet with a fully connected layer with 28 nodes that uses sigmoid cross-entropy loss. The latter is used because it is a standard loss function for solving multilabel classification problems; note that we are solving a multilabel classification task since each frame can have multiple objects and locations. We used the backpropagation algorithm with stochastic gradient descent

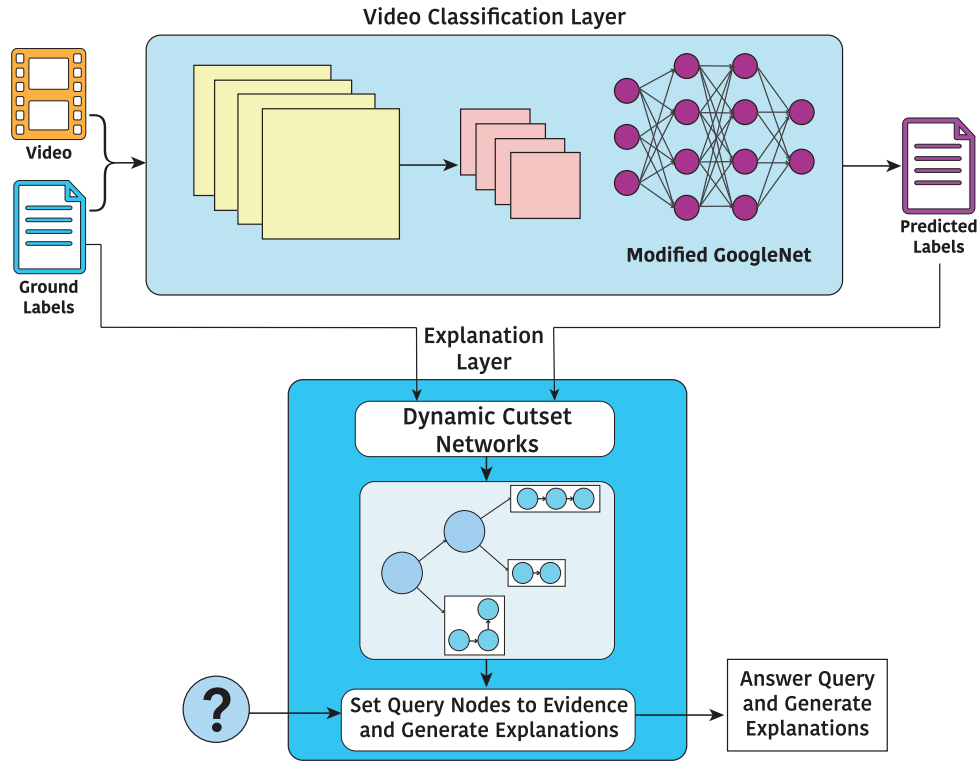


FIGURE 1 High-level architecture and data processing pipeline. Our system has two layers: a *video classification layer* based on a deep learning model whose output is fed to an *explanation layer* which is based on cutset networks,[5] an interpretable, tractable probabilistic model. During the learning phase, the classification layer uses the video frames and the ground truth activities (labels) as input and learns a mapping from frames to object, action, and location. On the other hand, during the learning phase, the explanation layer uses the labels predicted by the classification layer and ground truth as input and learns a mapping from predicted labels to the ground truth. During the query phase, the system answers questions by performing inference over the cutset network (in the explanation layer)

and Adaptive Moment Estimation (Adam) optimizers to further train the pretrained model on the TaCOS dataset and found that Adam yields the best performance.

3.4 | Explanation layer

In this section, we present DCCNs, a new tractable temporal probabilistic representation. We will use DCCNs in the explanation layer to: (a) correct errors in the labels predicted by the GoogLeNet at each frame; (b) model the dynamics as well as persistence (activities do not change rapidly between frames) in the video; and (c) provide explanations via polytime probabilistic inference.

3.4.1 | Conditional cutset networks

Tractable probabilistic models (TPMs)[40-42] are probabilistic models which admit polytime posterior marginal inference (MAR)—the task of computing marginal probability distribution over each variable given evidence which is defined as an assignment of values to a subset of variables—and maximum a posteriori (MAP) inference—the task of computing the most likely assignment to all nonevidence variables given evidence. Examples of popular TPMs include cutset networks,[5] arithmetic circuits,[42] sum-product networks,[43] and probabilistic sentential decision diagrams.[44] Although, TPMs are less expressive than intractable (latent) probabilistic models and as a result have slightly poor generalization performance as compared to the latter, their accuracy at test time is often much higher than intractable models. This is because tractable models use exact inference at prediction time while one has to use inaccurate approximate inference algorithms in intractable models.

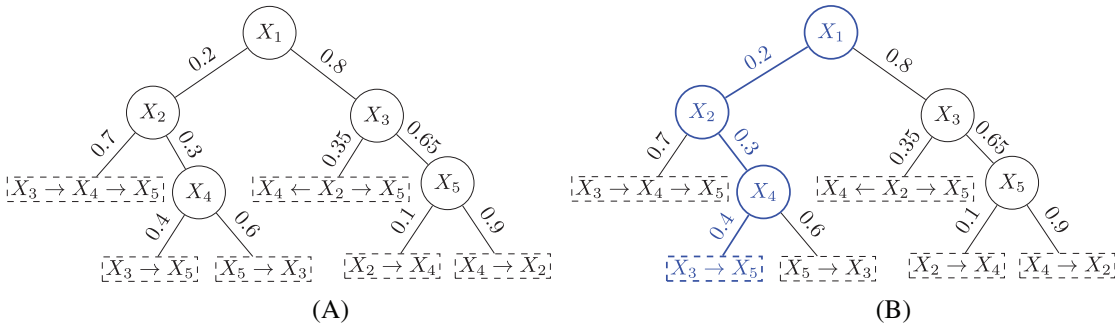


FIGURE 2 A, A cutset network over five variables $\{X_1,...,X_5\}$. Each variable takes a value from the binary domain $\{true,false\}$. OR nodes are denoted by circles. X_1 is the root node of the OR tree. Left and right arcs emanating from an OR node labeled by X_i indicate conditioning over *true* and *false* values of X_i, respectively. Arcs emanating from OR nodes are labeled with conditional probabilities. For example, the arc labeled with 0.4 denotes the conditional probability $P(X_4 = true|X_1 = true,X_2 = false)$. The leaf nodes are denoted by dashed rectangles and contain tree Bayesian networks over the remaining variables not appearing on the path to the leaf. B, The nodes, arcs and leaves activated during the computation of the query $P(X_1 = true,X_2 = false,X_3 = false,X_4 = true,X_5 = false)$

Cutset networks[5] are TPMs which represent multidimensional joint probability distributions using a rooted (directed) OR tree[45] with tree Bayesian networks at each leaf node of the OR tree (see Figure 2). Each OR node in the OR tree is labeled with a variable and just like in decision trees represents conditioning over the variable. Unlike decision trees, however, the arcs in the OR tree are labeled with conditional probability of the variable taking the corresponding value given an assignment of values from the root node to the OR node. Tree Bayesian networks at each leaf l represent the conditional distribution $P(\boldsymbol{X}|path(l))$* where $path(l)$ denotes the assignment from the root to l and $\boldsymbol{X}$ is the subset of variables not assigned in $path(l)$. MAR and MAP inference over cutset networks can be performed in linear time in the size of the network using cutset conditioning.[45,46] However, unlike conventional cutset conditioning algorithms, cutset networks take advantage of context-specific independence, dynamic variable orders and determinism. As a result, cutset networks can compactly represent and perform tractable inference in probability distributions that admit high treewidth probabilistic graphical models.[6,47,48]

Recently, Rahman et al[49] proposed a new framework called *conditional cutset networks* (CCNs) that extends the cutset networks framework to compactly represent and perform efficient reasoning over high-dimensional conditional probability distributions, namely, distributions of the form $P(\boldsymbol{Y}|\boldsymbol{X})$ where both $\boldsymbol{X}$ and $\boldsymbol{Y}$ are sets of random variables. To compactly represent the (exponentially many) conditional joint distributions over the variables $\boldsymbol{Y}$ given each assignment $\boldsymbol{X} = \boldsymbol{x}$, CCNs use a cutset network structure over $\boldsymbol{Y}$ whose conditional probability distributions $P(Y|path(n))$ at each OR node n as well as those attached to each variable in the Bayesian networks is replaced by calibrated probabilistic classifiers.[50] The latter takes an assignment $\boldsymbol{x}$ to $\boldsymbol{X}$ and $path(n)$ as input and outputs a conditional probability distribution over Y, namely, $P(Y|\boldsymbol{X} = \boldsymbol{x},path(n))$ by using only polynomial (in $|\boldsymbol{X}|$) number of parameters. For example, when we use logistic regression, we have $P(Y = 1|\boldsymbol{X} = \boldsymbol{x},path(n)) = \sigma\left(w_0 + \sum_{x_i \in \boldsymbol{x}} w_i x_i\right)$ where w_i's are the weights (parameters) and σ denotes the *sigmoid* function. We learn the parameters of the calibrated classifier (eg, logistic regression) using a subset of the data that is consistent with $path(n)$. CCNs are conditionally tractable in that given an assignment $\boldsymbol{X} = \boldsymbol{x}$, each (probabilistic) classifier yields a probability distribution over the (class) variable Y and thus given $\boldsymbol{X} = \boldsymbol{x}$, a CCN yields a (tractable) cutset network. (see Figure 3 for an example).

The structure of CCNs is learnt using the top-down induction algorithm detailed in Rahman et al.'s[49] paper. The base case occurs when the dataset contains either (a) a very small number of examples/records or (b) a very small number of variables. In such a situation, a simple tree-structured Bayesian network is powerful enough to represent the data distribution and can be learned using the Chow-Liu algorithm.[51] If the base condition is not satisfied, then the algorithm will recursively and heuristically select a single variable from the set of all currently available variables and then condition on it. For example, in a dataset defined over variables $\boldsymbol{Y} = \{Y_1,Y_2,Y_3,Y_4,Y_5\}$, the algorithm might find that Y_1 leads to the largest information gain in the data and choose to condition over it (such as in the CCN in fig.??). It will keep recursively doing this until it reaches the base case. After the structure is learned, the branch probability functions at each OR node in the CCN as well as each tree-structured Bayesian network at the leaves is learned using calibrated classifiers such as logistic regression and neural networks with a sigmoid layer on top (see Figure 3). The best calibrated classifier is chosen via cross-validation.

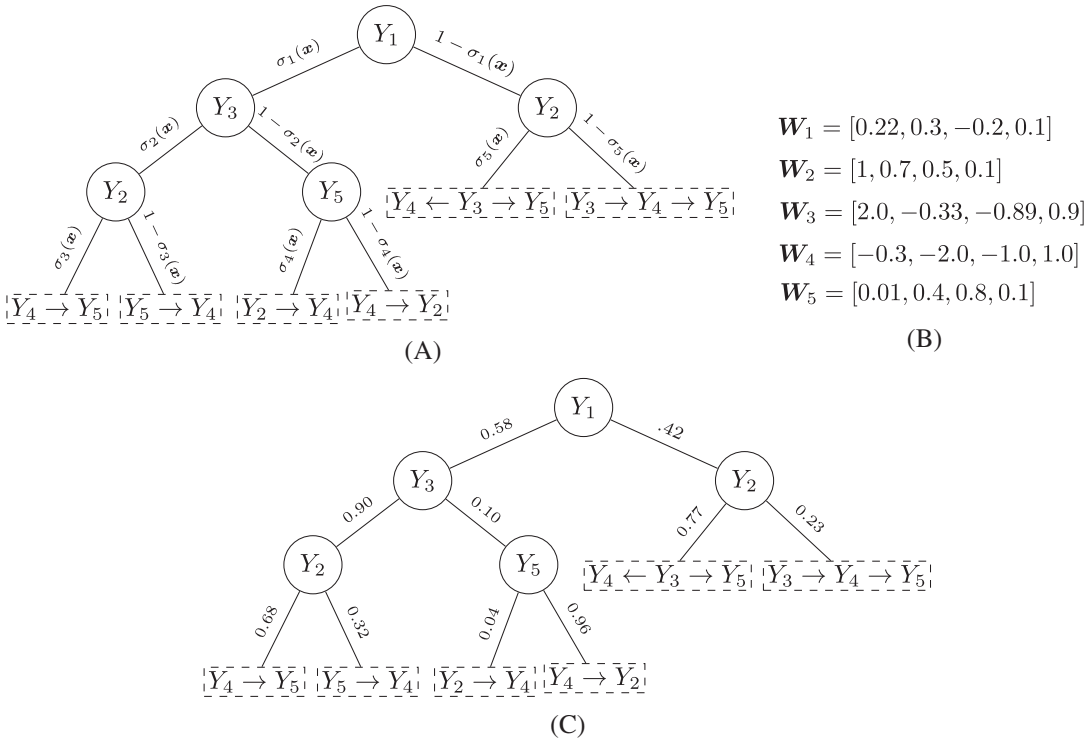


FIGURE 3 A, A conditional cutset network (CCN) representing $P(Y_1, ..., Y_5|X_1, X_2, X_3)$. Arcs emanating from OR nodes are labeled with sigmoid functions σ_1 through σ_5. For brevity, we omit showing sigmoid functions for the conditional probability distributions in the tree Bayesian networks at the leaves. B, Each $\boldsymbol{W}_i$ is the parameter vector of the corresponding sigmoid function $\sigma_i(\boldsymbol{x})$. C, Given the assignment $(X_1 = true, X_2 = true, X_3 = false)$, the CCN yields a cutset network having the same structure as the one given in (A) except that the parameters will be computed using the classifiers denoted by $\sigma_i(\boldsymbol{x})$

To use CCNs in our framework, we feed the output of GoogLeNet to the CCN. More formally, let $\boldsymbol{Y}$ denote the set of output nodes of GoogLeNet and $\boldsymbol{X}$ denote the set of true labels at a frame. We use the CCN to model $P(\boldsymbol{X}|\boldsymbol{Y})$ and learn its structure and parameters using a dataset constructed as follows. Each frame in each video is a training example and is composed of true labels ($\boldsymbol{X}$) and labels predicted by GoogleNet ($\boldsymbol{Y}$) with the pixels in the frame as input. At test time, at each frame, we instantiate all the classifiers in the CCN using the predicted labels to yield a cutset network and then perform inference over the cutset network to yield the final set of labels. In other words, the CCN treats the output of the neural network as a noisy sensor (see Figure 1) and computes a conditional joint probability distribution over the true labels given the predicted (noisy) labels.

We now describe how the compactness and tractability of CCNs can be leveraged to model the temporal dynamics in videos.

3.4.2 | Dynamic conditional cutset networks

An issue with CCNs is that they are static and do not explicitly model temporal aspects of video. For instance, we can use persistence, namely, objects do not change their position rapidly between subsequent frames to correct prediction errors at a frame by using data from neighboring frames. To address this issue, we propose a novel framework called DCCNs. Formally, let a video consist of n frames, let $\boldsymbol{Y}_t$ and $\boldsymbol{X}_t$ be the set of true labels and predicted labels (evidence) at frame t. Then, the DCCN represents the following probability distribution:

$$P(\boldsymbol{y}_{1:n}|\boldsymbol{x}_{1:n}) = P(\boldsymbol{y}_1|\boldsymbol{x}_1) \prod_{i=2}^{n} P(\boldsymbol{y}_i|\boldsymbol{y}_{1:i}, \boldsymbol{x}_{1:i-1}), \tag{1}$$

where the notation $\boldsymbol{x}_{1:n}$ (similarly $\boldsymbol{y}_{1:n}$) denotes an assignment of values to all predicted (true) labels in frames 1 to n. We will use the notation $\boldsymbol{X}_{1:n}$ to denote the set $\cup_{i=1}^{n}\boldsymbol{X}_i$.

The representation given in Equation (1) is not compact as n increases. To circumvent this issue, we use two standard assumptions widely used in temporal or dynamic probabilistic models—the 1-Markov and stationarity assumptions.[52] Specifically, we assume that each frame is conditionally independent of all frames before it given the previous frame (1-Markov) and all conditional distributions are identical (stationarity). With these assumptions, we can represent $P(\boldsymbol{y}_{1:n}|\boldsymbol{x}_{1:n})$ using

$$P(\boldsymbol{y}_{1:n}|\boldsymbol{x}_{1:n}) = P(\boldsymbol{y}_1|\boldsymbol{x}_1)\prod_{i=2}^{n} P(\boldsymbol{y}_i|\boldsymbol{x}_i,\boldsymbol{y}_{i-1}), \tag{2}$$

where $P(\boldsymbol{Y}_1|\boldsymbol{X}_1)$ and $P(\boldsymbol{Y}_i|\boldsymbol{X}_i,\boldsymbol{Y}_{i-1})$ are CNNs and $P(\boldsymbol{Y}_i|\boldsymbol{X}_i,\boldsymbol{Y}_{i-1})$ is the same for all i.

We learn DCCNs using the following approach. The prior model $P(\boldsymbol{Y}_1|\boldsymbol{X}_1)$ is the same as the CCN described in the previous section. To learn the structure and parameters of $P(\boldsymbol{Y}_i|\boldsymbol{X}_i,\boldsymbol{Y}_{i-1})$, we construct the dataset as follows. Each frame in each video is a training example and is composed of true labels at frame i ($\boldsymbol{Y}_i$), true labels at frame $i-1$ ($\boldsymbol{Y}_{i-1}$) and labels predicted by GoogleNet at frame i ($\boldsymbol{X}_i$) using the pixels in the frame as input.

Inference over DCCNs can be performed using sequential sampling approaches such as particle filtering and smoothing.[53] Here, we generate k assignments $\left(\boldsymbol{y}_1^{(1)},...,\boldsymbol{y}_1^{(k)}\right)$ uniformly at random from $P(\boldsymbol{Y}_1|\boldsymbol{x}_1)$, then for each assignment $\boldsymbol{y}_1^{(i)}$ we sample one assignment from $P\left(\boldsymbol{Y}_2|\boldsymbol{x}_2,\boldsymbol{y}_1^{(i)}\right)$, and so on. At the end of the sampling process, we will have k particles from $P(\boldsymbol{Y}_{1:n}|\boldsymbol{x}_{1:n})$. The main virtue of DCCNs is that unlike widely used temporal models such as DBNs,[54] the particles in DCCNs are generated from the posterior distribution $P(\boldsymbol{y}_i|\boldsymbol{x}_i,\boldsymbol{y}_{i-1})$ at each frame. As a result, issues such as particle degeneracy—particles vanish because their weights become too low as i increases—that typical sequential sampling algorithms suffer from will be less severe in DCCNs.

The three explanation types (see Section 3.1) can be computed by performing MAP and MAR inference in CCNs and DCCNs. To compute video explanations, we use an ontology that models relationships between activities and objects (eg, "chef's knife" is related to "kitchen knife," "slice" is related to "cut," etc.) and display video segments in which the marginal probability of the queried activity or activities related to it (recall that activity is a (*action*, *object*, *location*) triple) is larger than a threshold. Ranked triples over each segment in video explanations are computed by performing k-best MAP inference. Again, the key advantage of CCNs and DCCNs is that k-best MAP inference is linear in k and the number of parameters at each frame. Most probable entities in each highlighted segment are derived as follows. We compute the marginal probability of each entity in each highlighted segment given evidence (via MAR inference) and display the top k most likely ones according to the marginal probability. Note that these inferences are not possible on GoogLeNet or recurrent deep architectures[34] unless we treat the labels as independent entities.

3.5 | Video compilation and query processing

In this section, we explain how each video is individually processed and compiled when it becomes available to the system and how this compiled knowledge is later used for answering queries and providing explanations. We call this the compilation and query processing pipeline (for an example, see Figure 4). The pipeline has the following two phases:

- *Compilation phase*: When a new video becomes available to the system, relevant information about the video is compiled and stored in a database. The database is then used to answer queries in real-time. More specifically:

1. Each frame t of the video is passed through GoogLeNet which infers a set of noisy labels $\boldsymbol{x}_t$.
2. All the frames are then passed to the dynamic cutset network which finds the top-k activities in each frame t by constructing and inferring over the posterior distribution $P(\boldsymbol{y}_t|\boldsymbol{x}_{1:t})$ where $\boldsymbol{y}_t$ denotes the true labels associated with frame t and $\boldsymbol{x}_{1:t}$ denotes the noisy labels from frame 1 to frame t.
3. All consecutive frames with the *same* top explanation are then grouped into a *segment*. In addition, the dynamic cutset network also computes the component-wise marginal probabilities and these are averaged over each segment.

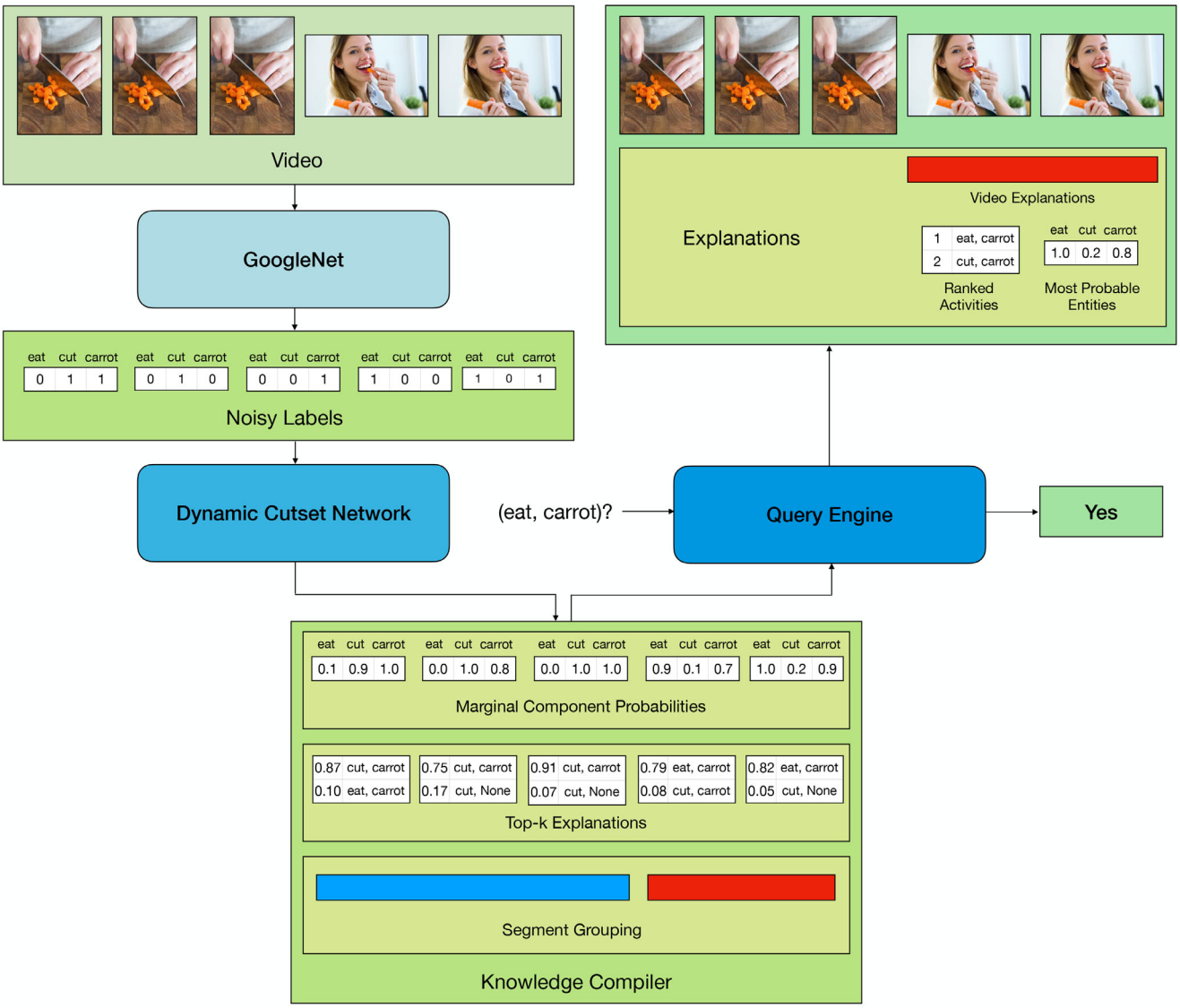


FIGURE 4 An example of a dummy video with five frames being passed through the compilation and query pipeline over three labels—*eat*, *cut* and *carrot*. For brevity, an activity is treated as a *pair* instead of a *triple* and comprises of only *action* and *object*. The frames are first individually passed through GoogLeNet which assigns noisy labels to them. They are then passed through the dynamic cutset network that groups the video into segments based on the top most likely activity given the noisy labels. In addition, it also computes the marginal component probabilities and other top-k explanations and stores them. Finally, when a query is posed to the system, the query engine searches for a segment that *completely* matches the query on *all* components. If such a segment exists, then it answers "yes" (otherwise it answers "no" and the partial matches are shown as explanations). It then fetches the explanations and shows them to the user by highlighting the segment of the video that is matched and showing the ranked activities and most probable entities averaged over the entire segment

4. Finally, the explanations and marginal probabilities are stored in a database for fast retrieval.

- *Query phase*: In this phase, the user poses a selection query to the system. These queries are in the form of whether a given combination of *action*, *object*, and *location* exists in the video. The system then searches over all the compiled segments (stored in a database) for matches on the most probable activity for each segment. If at least one *complete* match is found, then the system answers "yes" and returns *all* the segments that match the query, along with their explanations. Otherwise, the system answers "no" and as explanations displays all segments with partial matches. For example, if the query is asking if the person cuts a carrot on a cutting board (*cut*, *carrot*, *cutting board*) and there are no exact matches, then the system might return segments where the person is cutting a carrot, but on a plate (*cut*, *carrot*, *plate*) or washing a carrot in the sink (*wash*, *carrot*, *sink*).

3.6 | Interactive explanatory interface

We originally sought after an interactive interface that allowed users to load videos, ask queries, and review the model output along with the explanations. The goal for the interface design was to limit the amount of model information presented to the users in order to avoid overwhelming them with information. To this end, we first designed the interface with a predefined list of queries which users were able to select and explore the model outputs for, and later, sought after a new design where users were allowed to build their own queries. Figures 5 and 6 show the resulting interfaces, respectively.

Initially, the queries were in form of yes/no questions that consisted at least two of the three combinations of particular *actions*, *objects*, and *locations* that defined an activity. An example query (as seen in Figure 5) is "Does the person *peel* an *onion*?" However, later-on, we opted for a query building tool where users could choose an *action*, an *object*, and a *location* from a list of potential vocabulary. Unlike the first approach, this method allowed users to search by selecting as little as one activity component. With this approach, people could form they query implicitly in their minds and by looking at the selected activity components. For example, one possible query is (*peel*, *any object*, *cutting board*), which can be mapped to the natural-language question: "Does the person *peel anything* on the *cutting board*?"

The model then attempted to answer the query. For the initial interface, it was a simple response of yes or no. This was under the assumption that the video was already loaded, that is, the person could see and select a query from the list of predefined queries for the loaded video. However, with the query building tool, we wanted to allow a user to show the model response to all the videos before deciding which video to inspect. Because of this, we designed the interface to organize the videos into two groups, based on whether model found a video was a match for the query or not.

With both of the designs, we showed the same interface elements for the explanations. These explanations comprised the video segments that were most relevant to finding the answer to the query, and for each of these segments, the top three activities found in the query as well as the model's confidence in observing individual activity components (ie, *actions*, *objects*, or *locations*). This information was always visible per video, that is, they were local explanations. Thus, for the initial interface (as seen in Figure 5), users always accessed the information while for the second interface, this information would pop-up when a video was selected (in interface shown in Figure 6).

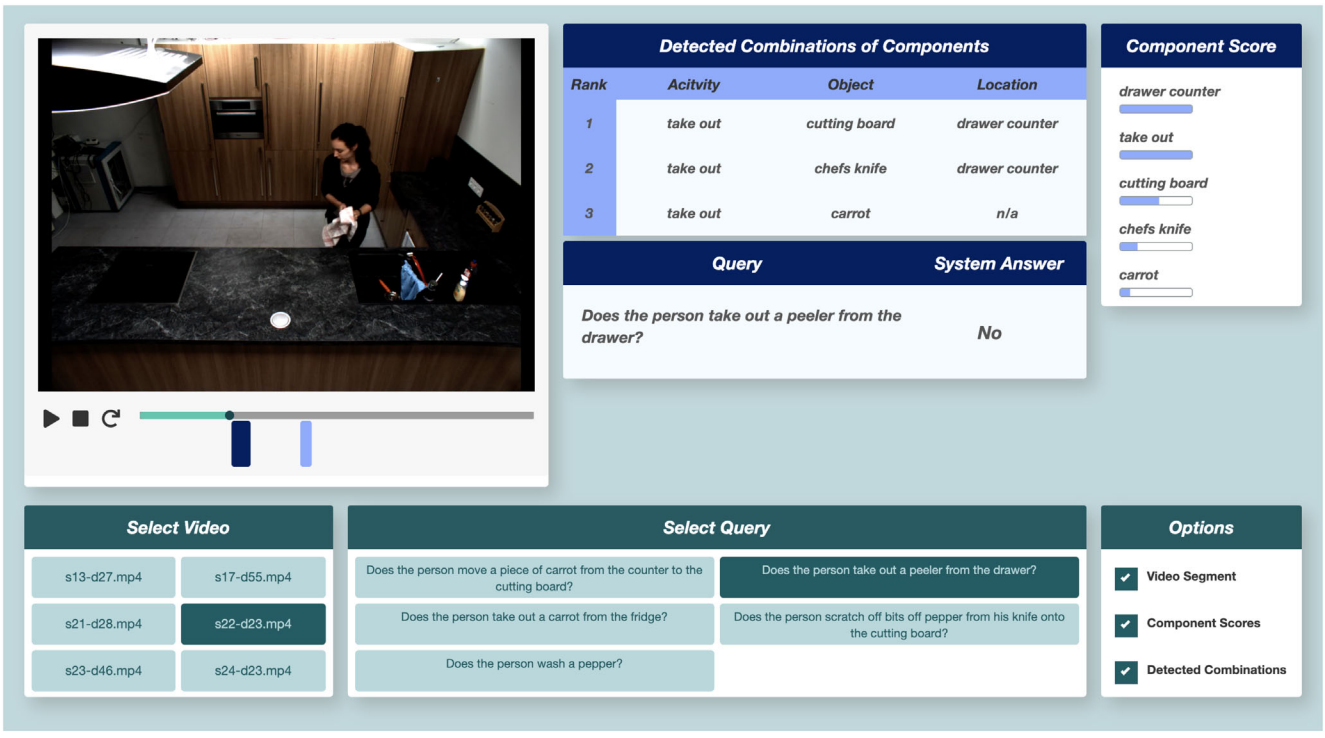


FIGURE 5 The interactive visual interface allows users to load videos and ask queries. The interface shows the ML system's answer along with explanatory elements for the output. The most relevant portions of the video play time are shown by colored bars beneath the video, and the right side shows detected video components and combinations of components relevant to the video and query. A similar version of this system was used in one of our previous user studies[55]

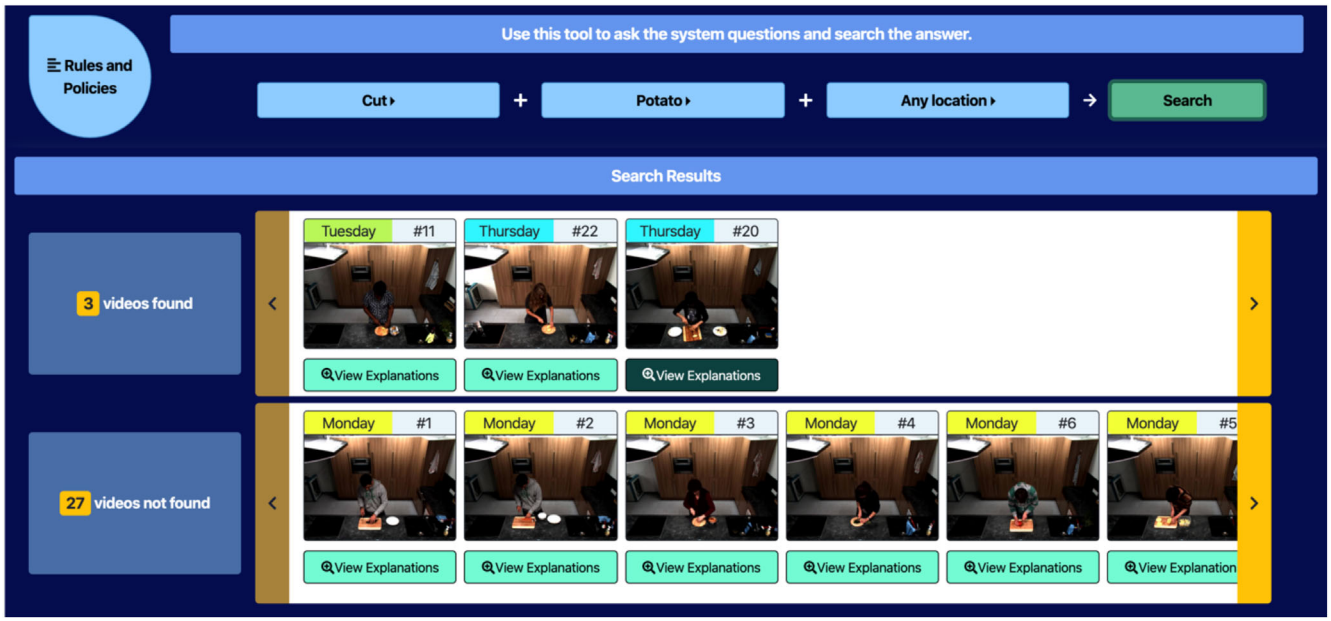


FIGURE 6 The overview of the query selection interface—as discussed in more detail in our previous work[56]—used in the policy-verification task, where users were able to query the model and find all the relevant videos. Upon clicking each video (or the view explanations bottom), a similar interface to Figure 5 would pop-up

Upon selecting a query (and choosing a video in the second interface), the video player highlights the most relevant segments of the video through visual annotations added under the video play bar (shown as light and dark blue under the video in Figure 5). The video player will also automatically jump to the appropriate segment to help users see the video frames most important for determining the output. For each selected video segment, the detected combination of components showed the top three detected activities within this segment (in form of a table), as well as the individually detected activity components. To help users to quickly judge these component scores, graphical bars are shown underneath detected components to visually represent the values of the component scores. Users can select different video segments to view the corresponding component scores and combinations from different portions of the video. The selected segment was represented with a darker color.

One of our main goals when designing these interfaces (particularly, second design) was supporting users to build better mental models of the model strengths and weaknesses (ie, errors). However, both of the interfaces we described gave greater attention to false positives (FPs) while giving less attention to false negative (FN) errors. In other words, it was easier/faster to discover and pay attention to FP errors as opposed to their counterparts. Since it is important to understand both these types of errors to (a) build a more accurate mental model of the system and (b) be able to fix the errors with the model when designing an algorithm, we aimed to design yet another exploratory interface to provide more support for both of the tasks.

We hypothesized that the addition of holistic explanations to provide broader model-level information will increase user's ability to equally pay attention to and identify FN and FP errors, as well as higher-level model-based problems across multiple videos. We refer to these more holistic representations as "global explanations" as they provide insights about the model's logic by displaying multiple possible outputs at once with overall system performance indicators (ie, accuracy, precision, recall, etc.). This view was designed to aid user understanding of higher-level model-based problems across multiple videos rather than focusing on problem identification on a per video basis.

Figure 7 shows the visual design for the global view of the interface. In the video selection panel, each video shows a list of the top five components the system detected alongside an estimated rate of FPs and FNs for the video. With each video, a set of temporal heat maps are provided, representing the model's detection confidence for each component in the system's vocabulary over the video's duration. In addition, a global information panel is included to provide general system performance information. This includes an estimation of the overall system detection accuracy, FP rate, and FN rate. Finally, this panel contains a bar chart for both FP and FN rates per each object in the system.

FIGURE 7 The overview of the interface that included the global representations of the model as well as instance-level outputs. This interactive interface is not dependent on specific queries like those demonstrated in Figures 5 and 6

TABLE 1 Accuracy for activity recognition on test videos

Metric	GoogLeNet	CCNs	Dynamic CCNs
K-1	0.9335	0.9677	**0.9687**
K-2	0.8557	0.8998	**0.9197**
K-3	0.7918	0.7962	**0.8168**
Jaccard index	0.8608	0.8559	**0.8674**
Hamming loss	0.1392	0.1286	**0.1160**

Note: Bold results indicate the best performing model.
Abbreviation: CNN, conditional cutset network.

4 | EVALUATION

4.1 | ML evaluation

We selected 60 313 frames for training and 9355 frames for testing distributed over 17 videos in the TACoS dataset. For each set, we selected a set of ground labels and used the video classification layer to generate the predicted labels. We performed exact inference over CCNs and used particle filtering with 100 particles for inference in DCCNs. We performed the following ablation study: (a) our system in which the explanation layer is removed (GoogLeNet); (b) our system which uses (static) CNNs in the explanation layer (CCNs); and (c) the full system (DCCNs).

Table 1 outlines the accuracy scores for correct activity recognition according to various evaluation metrics. Since predicting each activity correctly is a multilabel classification task, we use K-Group measures to calculate the overall percentage of instances where K labels out of the total number of labels were predicted correctly. We report K-1, K-2, and K-3 since each activity comprises of *action*, *object*, and *location*. In addition, we also use standard measures such as the Hamming loss and the Jaccard index. We notice that the full system that uses the DCCNs yields the best scores. This is expected, since the DCCN encodes both the error distribution of the labels predicted by the neural network at each given frame as well as the transition distribution of how labels evolve over time.

4.2 | Human evaluations

To evaluate the effectiveness of the explanations with user interactions, we conducted a series of studies that focused on human performance. These experiments aimed to study various human factors with explainable AI systems, specifically, in the context of decision-support systems, human-AI collaboration, and video activity recognition. In this section, we rely on a brief description for these studies and refer the audience to the full manuscripts to learn more about the details of each study.[55-57]

4.2.1 | Evaluating human-machine query verification

A primary goal in this study was to measure the degree to which the explanations generated by our system would benefit the end users with little to no understanding of how ML systems work. We hypothesized that the presence of explanations would improve both the speed and the accuracy of decision-making. We also hypothesized that user agreement with the system's outputs would significantly increase with explanations. A user's answer is said to "agree" with the system's when they are the same.

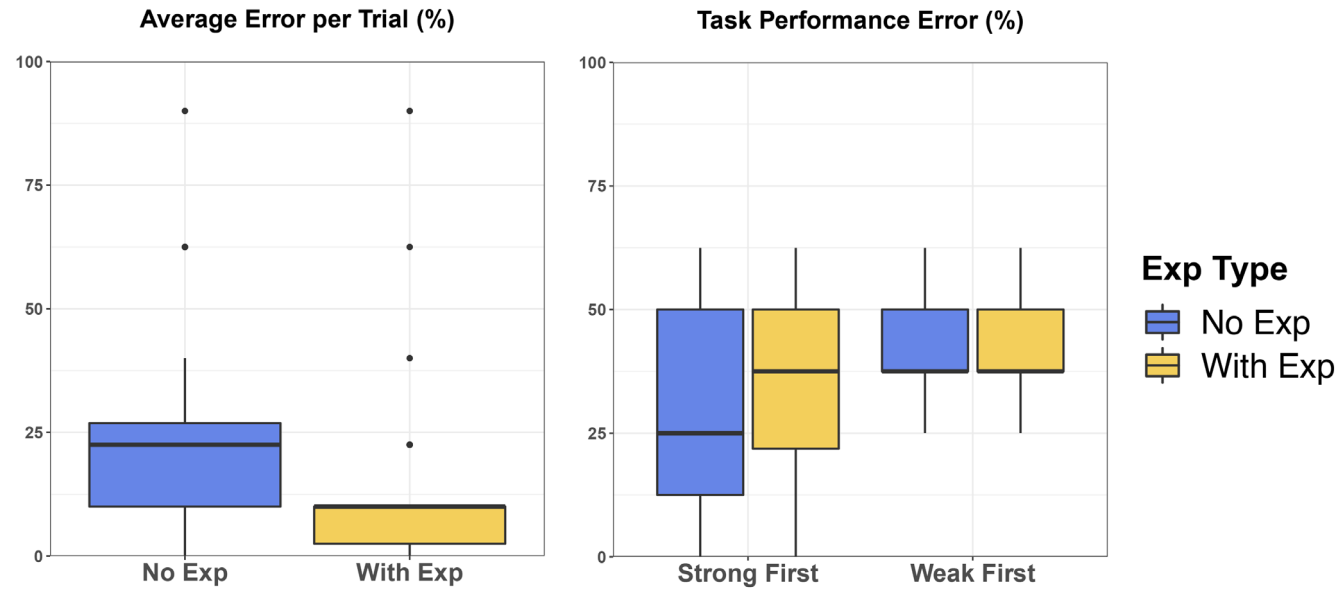
To test our hypotheses, we designed a user study where participants were set to review the model's responses and explanations to a set of queries about videos, using a similar design from interface shown in Figure 5. More detailed study description, other findings and results, and discussions from this study is available in our previous paper.[55] The query-review task was designed to assess the participants' ability to accurately determine the correct answer to queries with the aid of the system. To clarify, we define a "correct" answer here as the actual answer (or ground truth) of a given query. This is different from "agreement" (defined above) where the user's answer matches the one from the system. For example, there could be cases where both the user and the system agree on the same (incorrect) answer; however, this might be different from the actual, "correct" answer. Through a between-subjects user study, we divided the participants into two groups: *with* and *without* explanations. Participants from both groups had access to the video player and the system's answer to each question. However, while those *with explanations* were provided with the interface seen in Figure 5, their counterparts did not see any explanations (ie, they were only able to view the system's answers and not the video segments, the detected combination of components,and the component scores). Overall, each participant reviewed 20 unique queries with a ratio of 16-4 correct-incorrect system answers.

The experiment was completed online by 80 AMT workers. Of these participants, 40 of them were shown explanations while the other 40 were not. After preprocessing the data and removing outliers that did not fall within 1.5 × IQR, we analyzed results from 38 participants for the *with explanations* category and 40 for the *without explanations* category. We analyzed the results using the Kruskal-Wallis nonparametric test to measure the difference between the two groups. We observed a significant difference on error per trial ($\chi^2(1,76) = 5.63, P < .05$), showing that the participants *with explanations* had significantly less error than those *without explanations*. Our experiment also detected a significant difference on average time per trial ($\chi^2(1,76) = 28.1, P < .001$). Participants *with explanations* were significantly faster. In addition, the results from the user agreement with the system show that participants *with explanations* significantly agreed with the system more than their counterparts ($\chi^2(1,76) = 8.00, P < .01$). Figure 8A shows average participant error per trial.

Overall, these results support our hypothesis that the addition of these explanations significantly improves user-task performance in our system. Through this study, we learned that providing more information (through post hoc explanations) can support user understanding of the system and judge when it is correct. It would seem that the explanations encouraged the users to *correctly* trust its output. Since the *with explanations* category also had significantly better performance results, this suggests that the higher rate of agreement was not simply blind trust or *automation bias*,[58] where humans tend to trust an intelligent system by virtue of its "intelligence" alone. However, it is to be noted that our study was not designed to specifically focus on the potential effects of explanations on automation bias.

4.2.2 | Evaluating open-ended human-machine video review

In the first evaluation, we designed a task where users reviewed and answered queries about activities and objects in designated videos. Building on these results, we used the same underlying algorithm, to explore how users attempt to understand model competencies and weaknesses when given the freedom to explore. In addition, we also wanted to



(A) **Experiment 1**: Human-Machine Query Verification (B) **Experiment 2**: Open-ended Video Review

FIGURE 8 A, Participant error on the policy task (percentage) from Experiment 1 (see Section 4.2.1). B, Average participant error per trial (percentage) from Experiment 2 (see Section 4.2.2)

understand the role of first impressions on users' mental model formation. We performed a user study based on the interface described in Section 3.6 and seen in Figure 6, and the results of this work were recently published.[56,57] We will therefore briefly describe the study and key findings here and advice those who seek to learn about the study in more detail to refer to our prior work.

We designed a policy-verification task, where participants were asked to verify whether a set of kitchen guidelines and policies are being followed by the people performing cooking activities by utilizing the query-building system. The policies were designed such that half were in reference to known system weaknesses while the others exposed system competencies. Assuming that most users began from the top of the list and worked their way down, we changed the order of the policy sets and compared how participants experienced the system. As an additional variable, half of the participants were provided explanations while others were not. After a brief video tutorial, users freely explored how the tool classified different combination of detected components as they tried to verify system policies before being asked about their impressions of the system. We ultimately asked 110 participants to review the system, 54 observed explanations: 28 of whom saw policies that exposed model capabilities first and another 26 observed those that exposed model weaknesses early-on. Of those provided no explanations, the number of participants were 29 and 27, with the respective order.

First, we measured the proportion of policies that were answered correctly (user-task error). Our results indicate that participants who saw system strengths early-on made significantly more errors in the policy-verification task compared to those who encountered weaknesses first, with $F(1,106) = 6.55$, $P < .05$, $\eta_p^2 = 0.058$. This indicates that encountering strengths earlier can lead to overreliance on the model outcomes (ie, automation bias), while seeing weaknesses in the beginning can prevent this problem. Figure 8B shows the distribution of the task error results across the conditions.

After completing the policy-review task, participants were asked to estimate the model's detection accuracy (percentage) for several activity components in the system's vocabulary that corresponded to both system strengths and weaknesses. For each participant, we calculated the error in their estimated accuracy for that component. For components that corresponded to system strengths, participants who observed weaknesses first significantly underestimated the model's detection accuracy compared to their counterparts, with $F(1,106) = 6.24$, $P < .05$, $\eta_p^2 = 0.056$. In addition, those who observed weaknesses early-on were significantly less confident about their estimations, with $F(1,106) = 3.94$, $P < .05$, $\eta_p^2 = 0.036$.

This shows that participants who observed system weaknesses first had problems forming their mental models of the system competencies and strengths. They significantly underestimated the system capabilities while also having less

confidence in their estimations. These users were skeptical of system strengths yet showed hesitancy in their skepticism because their earlier negative observations obscured their judgment of the system capabilities. With a negative first impressions of the system capabilities, users tended to rely more on their own abilities and focused less on how the model performed. Since they were more focused on completing the task themselves, questions about their impressions of the system capabilities may have been unexpected—leading to the confusion reported in our results. On the other hand, users who experienced system strengths first tended to exhibit behaviors related to overconfidence. Their performance on the policy review task was significantly worse than their weakness first counterparts. They appear to develop a false sense of security (ie, automation bias) early on, showing that when they looked at the later policies, they generally continued to rely on the system, even when it made mistakes. Having overconfidence in system capabilities had no influence on completion time. Yet, if users took about the same amount of time to complete the policy review task, it appears that being overconfident had the undesired effect of decreasing user's critical review of system outputs.

Overall, these results suggest an additional nuance to the findings from the previous user study described in Section 4.2.1. While the prior study found the addition of explanations significantly improved human verification performance, the second study demonstrates that explanations cannot be assumed to be universally beneficial. Even more notable is the knowledge that explanations might increase the likelihood that users will develop unfounded knowledge of the model (ie, they think they understand how the model works when, in fact, they do not). In safety critical applications, these impression effects can have disastrous consequences. In this study, we found evidence to support that early impressions of a ML model is fundamental for users to establish their assessment of model capabilities. When introducing end-users to XAI systems, attention should be given to how to expose users to both system weaknesses and strengths in a balanced way early on to help ensure accurate mental models are developed and maintained by users.

4.2.3 | Using global explanations to debug the model

While the previously described user studies focused on human-machine performance for video review and inspection user tasks, the third evaluation—using the interface described in Section 3.6 and seen in Figure 7—addresses a different use case involving developers debugging a model. To demonstrate the utility of the tool for deeper, more detailed model inspection, we present case studies using two datasets in which an experienced ML designer from the research team used the global explanation view to identify various types of errors and problem patterns in the underlying model for debugging purposes. The first case study was based on the activity recognition model created with the previously described TACoS video corpus[9] of kitchen activities. For the second case study, we chose a set of six videos from the Wet Lab dataset,[59] which consists of videos of laboratory experiments involving chemical and biological testing. While we performed and included a comprehensive case study in a prior work,[60] we will only briefly touch upon the case studies in the current paper and refer those interested to learn more about the details on the system design goals and choices, usability user evaluation, and the two case studies to read our past paper.[60]

Our interface from Figure 7 provides possibilities to pay attention to both FP and FN errors, equally, as demonstrated by both of our case studies. To achieve this, the subject examined which video to explore by referring to the objects that have the highest FP and FN rates using the bar charts in the global information panel. Once he decided to explore a specific object, for example, object X, he could use the top five components to identify which video highly includes object X. By comparing the heat maps based on their colors and (ie, the model's detection confidence per object, per second) using the blue bar, he could explore many scenarios where X is a FP or FN *individually*, or may compare various heat maps against X to find cases where X (or other components) are FP or FN in the same *activity*. This information can then be used to penalize the model parameters proportional to the observed probability of the error to ensure the final model learnt will make fewer errors of each type. This is one approach to use this exploratory tool, and allows a model designer to navigate and find various types of errors (with or without combinations) or even find the sources of a problem by comparing similar combinations or individual components with errors across multiple videos to find similar global error patterns. For example, a debugger might notice that a given video has a high number of FPs. After clicking on the video, the global information panel might show that the FP rate for "plate" is 20%. The debugger can then browse through the sections in the video that have high confidence scores for plate by looking at the heat map (see Figure 7) and notice that the probability for plate goes up whenever there is a frying pan in the frame. More technical details of these findings and a more detailed walkthrough of the case studies are available in our prior paper.[60]

Through our case studies, we learnt that providing higher-level explanations that are not limited to a query or specific activity can support identification of various specific types of model errors and allows more experienced model architects to explore the model to find high-level, global error patterns as well as instance-level problems. This can be used to alleviate the bias toward focusing primarily on FP errors that we observed with AI-assisted querying.

5 | CONCLUSION

In this paper, we presented the technical details of our proposed approach that seeks to address the following accuracy-explanability gap in existing ML technology: deep neural networks yield accurate predictions but are not explainable while conventional classifiers such as linear models and decision trees are explainable but substantially less accurate. The key idea in our approach is to *compose* neural networks with explainable, tractable probabilistic logic representations. We presented a version of this general approach in which we first used a neural network to yield accurate estimates over the labels (or classes), then used these estimates as *soft evidence* in a probabilistic model called *dynamic cutset networks*, and finally derived decisions and explanations by performing reasoning over the latter. The main virtue of dynamic cutset networks is that they are tractable. Therefore, they can answer reasoning queries—both decision and explanation queries—accurately and often in linear time in the size of the model. We showed that for the task of activity recognition in cooking videos derived from the TACoS dataset, an XAI system based on our approach not only has better predictive performance than the one based on neural networks alone but also has superior explanatory power. The two main lessons learned from building this explainable system were:

- *Lesson 1*: To build models that are both accurate and explainable, *compose* a neural network defined over low-level features with probabilistic models that offer superior reasoning capabilities (and thus better explanations via reasoning).
- *Lesson 2*: In order to ensure that explanations are accurate, generated in real-time and faithful to the model used to make decisions, use *tractable* models.

We evaluated our explainable video activity recognition system by using it to solve three human-machine tasks: (a) answering yes/no questions posed over a given video; (b) answering whether a set of policies were followed or not in a given collection of videos; and (c) debugging the XAI model. We built novel interactive visual interfaces for each task and conducted human subjects studies. The lessons learned from these studies were:

- *Lesson 3*: If the XAI system for solving a human-machine task is highly accurate and the task is relatively easy, provide detailed explanations. Explanations significantly and justifiably improve the task completion time and the user's trust and reliance in the system, and reduce the gap between the user's perceived accuracy and the system's actual accuracy. On the other hand, if the XAI system for solving a human-machine task is not accurate, explanations wrongly amplify the automation bias in that they increase the gap between the user's perceived accuracy and the system's actual accuracy by increasing the former.
- *Lesson 4*: When the XAI system is used for solving relatively harder human-machine tasks (eg, in safety-critical applications) and has known weaknesses, provide a balanced, detailed view of the system's strengths and weaknesses when the user begins to use the system. These early impressions play a critical role in establishing the user's assessment of model capabilities and developing an accurate mental model.
- *Lesson 5*: For expert users of the XAI system, develop novel visual exploratory tools that clearly depict not only local explanations specific to a query but also more global, higher-level explanations that can support identification of various types of model errors. These global views can help alleviate various biases such as automation bias, anchoring bias, and bias toward focusing on FP errors, and help the expert user find high-level, global error patterns as well as instance-level problems.

ACKNOWLEDGMENT

This work was supported by the Defense Advanced Research Projects Agency Explainable Artificial Intelligence (XAI) Program under contract number N66001-17-2-4032.

DATA AVAILABILITY STATEMENT

The experiments were performed on the open-source TACoS dataset which is freely available online: https://www.mpi-inf.mpg.de/departments/computer-vision-and-machine-learning/research/vision-and-language/tacos-multi-level-corpus

ORCID

Chiradeep Roy https://orcid.org/0000-0001-5565-1247
Jeremy E. Block https://orcid.org/0000-0003-1626-9074
Vibhav Gogate https://orcid.org/0000-0002-6459-7358

ENDNOTE

* We denote random variables as uppercase letters, for example, X, Y, and so on, sets of random variables as bold uppercase letters, for example, $\boldsymbol{X}, \boldsymbol{Y}$, and so on, and assignment of values to all the variables in a set as bold lowercase letters, for example, $\boldsymbol{x}, \boldsymbol{y}$, and so on.

REFERENCES

1. Kulesza T, Burnett M, Wong W, Stumpf S. Principles of explanatory debugging to personalize interactive machine learning. Paper presented at: In Proceedings of the Twentieth International Conference on Intelligent User Interfaces (IUI). 2015.
2. Poole D. Probabilistic horn abduction and bayesian networks. *Artif Intell*. 1993;64(1):81-129.
3. Kimmig A, Demoen B, De Raedt L, Costa VS, Rocha R. On the implementation of the probabilistic logic programming language problog. *Theory Pract Log Program*. 2011;11(2–3):235-262.
4. Roy S, Suciu D. A formal approach to finding explanations for database queries. Paper presented at: ACM SIGMOD International Conference on Management of Data (MOD). 2014:1579-1590.
5. Rahman T, Kothalkar P, Gogate V. Cutset networks: a simple, tractable, and scalable approach for improving the accuracy of Chow-Liu trees. Paper presented at: Proceedings of the 2014 Joint European Conference on Machine Learning and Knowledge Discovery in Databases (ECML PKDD). 2014:630–645.
6. Rahman T, Gogate V. Learning ensembles of cutset networks. Paper presented at: Proceedings of the Thirtieth AAAI Conference on Artificial Intelligence. 2016:3301-3307.
7. Rooshenas A, Lowd D. Learning sum-product networks with direct and indirect variable interactions. Paper presented at: Proceedings of the Thirty-First International Conference on Machine Learning (ICML). 2014:710-718.
8. Liang Y, Bekker J, Van den Broeck G. Learning the structure of probabilistic sentential decision diagrams. Paper presented at: Proceedings of the Thirty-Third Conference on Uncertainty in Artificial Intelligence (UAI). 2017.
9. Regneri M, Rohrbach M, Wetzel D, Thater S, Schiele B, Pinkal M. Grounding action descriptions in videos. *Trans Assoc Comput Linguist*. 2013;1:25-36.
10. Starner T, Pentland A. Real-time american sign language recognition from video using hidden markov models. In: Shah M, Jain R, eds. *Motion-Based Recognition*. Dordrecht: Springer; 1997:227-243.
11. Bobick AF, Wilson AD. A state-based approach to the representation and recognition of gesture. *IEEE Trans Pattern Anal Mach Intell*. 1997;19(12):1325-1337.
12. Natarajan P, Nevatia R. Coupled hidden semi markov models for activity recognition. Paper presented at: In IEEE Workshop on Motion and Video Computing (WMVC). 2007:1–10.
13. Oliver NM, Rosario B, Pentland AP. A Bayesian computer vision system for modeling human interactions. *IEEE Trans Pattern Anal Mach Intell*. 2000;22(8):831-843.
14. Buxton H, Gong S. Visual surveillance in a dynamic and uncertain world. *Artif Intell*. 1995;78(1–2):431-459.
15. Gong S, Xiang T. Recognition of group activities using dynamic probabilistic networks. Paper presented at: IEEE International Conference on Computer Vision (ICCV). 2003:742-749.
16. Al-Hames M, Rigoll G. A multi-modal mixed-state dynamic Bayesian network for robust meeting event recognition from disturbed data. Paper presented at: IEEE International Conference on Multimedia and Expo (ICME). 2005:45-48.
17. Otsuka K, Yamato J, Takemae Y, Murase H. Conversation scene analysis with dynamic Bayesian network basedon visual head tracking. Paper presented at: IEEE International Conference on Multimedia and Expo (ICME). 2006:949-952.
18. Cooper GF. The computational complexity of probabilistic inference using bayesian belief networks. *Artif Intell*. 1990;42(2–3):393-405.
19. Pei M, Jia Y, Zhu SC. Parsing video events with goal inference and intent prediction. Paper presented at: IEEE International Conference on Computer Vision (ICCV). 2011:487-494.
20. Morariu VI, Davis LS. Multi-agent event recognition in structured scenarios. Paper presented at: IEEE Conference on Computer Vision and Pattern Recognition (CVPR). 2011:3289-3296.
21. Brendel W, Fern A, Todorovic S. Probabilistic event logic for interval-based event recognition. Paper presented at: IEEE Conference on Computer Vision and Pattern Recognition (CVPR). 2011:3329-3336.
22. Ivanov YA, Bobick AF. Recognition of visual activities and interactions by stochastic parsing. *IEEE Trans Pattern Anal Mach Intell*. 2000;22(8):852-872.

23. Ryoo MS, Aggarwal JK. Recognition of composite human activities through context-free grammar based representation. Paper presented at: IEEE Conference on Computer Vision and Pattern Recognition (CVPR). 2006:1709-1718.
24. Joo SW, Chellappa R. Recognition of multi-object events using attribute grammars. Paper presented at: IEEE International Conference on Image Processing (ICIP). 2006:2897-2900.
25. Zhang Z, Tan T, Huang K. An extended grammar system for learning and recognizing complex visual events. *IEEE Trans Pattern Anal Mach Intell*. 2010;33(2):240-255.
26. Si Z, Pei M, Yao B, Zhu SC. Unsupervised learning of event and-or grammar and semantics from video. Paper presented at: In IEEE International Conference on Computer Vision (ICCV). 2011:41–48.
27. Friedman N, Getoor L, Koller D, Pfeffer A. Learning probabilistic relational models. Paper presented at: Proceedings of the Sixteenth International Joint Conference on Artificial Intelligence (IJCAI). 1999:1300-1309.
28. Muir BM. Trust in automation: part i. theoretical issues in the study of trust and human intervention in automated systems. *Ergonomics*. 1994;37(11):1905-1922.
29. Muir BM, Moray N. Trust in automation. Part ii. Experimental studies of trust and human intervention in a process control simulation. *Ergonomics*. 1996;39(3):429-460.
30. Lee JD, See KA. Trust in automation: designing for appropriate reliance. *Hum Factors*. 2004;46(1):50-80.
31. Hoff KA, Bashir M. Trust in automation: integrating empirical evidence on factors that influence trust. *Hum Factors*. 2015;57(3): 407-434.
32. Hoffman RR. A taxonomy of emergent trusting in the human–machine relationship. In: Smith PJ, Hoffman RR, eds. *Cognitive Systems Engineering: The Future for a Changing World*. Boca Raton, FL: CRC Press; 2017:137-164.
33. Rohrbach A, Rohrbach M, Qiu W, Friedrich A, Pinkal M, Schiele B. Coherent multi-sentence video description with variable level of detail. Paper presented at: The Thirty-Sixth German Conference on Pattern Recognition. 2014:184–195.
34. Donahue J, Anne Hendricks L, Guadarrama S, et al. Long-term recurrent convolutional networks for visual recognition and description. Paper presented at: IEEE Conference on Computer Vision and Pattern Recognition (CVPR). 2015:2625-2634.
35. Krishna R, Hata K, Ren F, Fei-Fei L, Niebles JC. Dense-captioning events in videos. Paper presented at: IEEE International Conference on Computer Vision (ICCV). 2017:706-715.
36. Song YC, Naim I, Al Mamun A, et al. Unsupervised alignment of actions in video with text descriptions. Paper presented at: Proceedings of the Twenty-Fifth International Joint Conference on Artificial Intelligence (IJCAI). 2016:2025-2031.
37. Duan X, Huang W, Gan C, Wang J, Zhu W, Huang J. Weakly supervised dense event captioning in videos. Paper presented at: Advances in Neural Information Processing Systems (NeurIPS). 2018:3063-3073.
38. Szegedy C, Liu W, Jia Y, et al. Going deeper with convolutions. Paper presented at: IEEE Conference on Computer Vision and Pattern Recognition (CVPR). 2015:1–9.
39. Russakovsky O, Deng J, Su H, et al. ImageNet large scale visual recognition challenge. *Int J Comput Vis*. 2015;115(3):211-252.
40. Bach FR, Jordan MI. Thin junction trees. Paper presented at: Advances in Neural Information Processing Systems (NeurIPS). 2002:569-576.
41. Lowd D, Domingos PM. Learning arithmetic circuits. Paper presented at: Proceedings of the Twenty-Fourth Conference in Uncertainty in Artificial Intelligence (UAI). 2008:383-392.
42. Darwiche A. A differential approach to inference in Bayesian networks. Paper presented at: Proceedings of the Sixteenth Conference on Uncertainty in Artifical Intelligence (UAI). 2000:123-132.
43. Poon H, Domingos P. Sum-product networks: a new deep architecture. Paper presented at: Proceedings of the Twenty-Seventh Conference on Uncertainty in Artificial Intelligence (UAI). 2011:337-346.
44. Bekker J, Davis J, Choi A, Darwiche A, Van den Broeck G. Tractable learning for complex probability queries. Paper presented at: Advances in Neural Information Processing Systems (NeurIPS). 2015:2242-2250.
45. Dechter R, Mateescu R. AND/OR search spaces for graphical models. *Artif Intell*. 2007;171:73-106.
46. Pearl J. Bayesian inference. *Probabilistic Reasoning in Intelligent Systems: Networks of Plausible Inference*. San Francisco, CA: Morgan Kaufmann; 1988.
47. Di Mauro N, Vergari A, Basile TMA. Learning Bayesian random cutset forests. Paper presented at: Foundations of Intelligent Systems - 22nd International Symposium. 2015:122-132.
48. Di Mauro N, Vergari A, Esposito F. Learning accurate cutset networks by exploiting decomposability. Paper presented at: The Fourteenth International Conference of the Italian Association for Artificial Intelligence. 2015:221-232.
49. Rahman T, Jin S, Gogate V. Cutset Bayesian networks: a new representation for learning Rao-Blackwellised graphical models. Paper presented at: Proceedings of the Twenty-Eighth International Joint Conference on Artificial Intelligence (IJCAI). 2019:5751–5757.
50. Niculescu-Mizil A, Caruana R. Predicting good probabilities with supervised learning. Paper presented at: Proceedings of the Twenty-Second International Conference on Machine Learning (ICML). 2005:625–632.
51. Chow CK, Liu CN. Approximating discrete probability distributions with dependence trees. *IEEE Trans Inf Theory*. 1968;14:462-467.
52. Rabiner LR. A tutorial on hidden Markov models and selected applications in speech recognition. *Proc IEEE*. 1989;77(2):257-286.
53. Doucet A, de Freitas N, Murphy KP, Russell SJ. Rao-Blackwellised particle filtering for dynamic Bayesian networks. Paper presented at: Proceedings of the Sixteenth Conference on Uncertainty in Artificial Intelligence (UAI). 2000:176-183.
54. Murphy KP. Dynamic Bayesian Networks: Representation, Inference and Learning [PhD thesis]. Berkeley, CA: University of California, Berkeley. 2002.

55. Nourani M, Roy C, Rahman T, et al. Don't explain without verifying veracity: an evaluation of explainable ai with video activity recognition. *arXiv Preprint arXiv:2005.02335*, 2020.
56. Nourani M, Roy C, Block JE, et al. Anchoring bias affects mental models and user reliance in explainable ai systems. Paper presented at: Proceedings of the Twenty-Sixth International Conference on Intelligent User Interfaces (IUI). 2021.
57. Nourani M, Honeycutt DR, Block JE, et al. Investigating the importance of first impressions and explainable ai with interactive video analysis. Paper presented at: Extended Abstracts of the 2020 CHI Conference on Human Factors in Computing Systems Extended Abstracts. 2020:1–8.
58. Goddard K, Roudsari A, Wyatt JC. Automation bias: a systematic review of frequency, effect mediators, and mitigators. *J Am Med Inform Assoc*. 2012;19(1):121-127.
59. Naim I, Song Y, Liu Q, Kautz H, Luo J, Gildea D. Unsupervised alignment of natural language instructions with video segments. Paper presented at: Proceedings of the Twenty-Eighth AAAI Conference on Artificial Intelligence. 2014:1558–1564.
60. Nourani M, Roy C, Honeycutt DR, Ragan ED, Gogate V. Detoxer: a visual debugging tool with multi-scope explanations for temporal multi-label classification models. 2021.

How to cite this article: Roy C, Nourani M, Honeycutt DR, et al. Explainable activity recognition in videos: Lessons learned. *Applied AI Letters*. 2021;2(4):e59. doi:10.1002/ail2.59