Explainable Activity Recognition in Videos using Deep Learning and Tractable Probabilistic Models

CHIRADEEP ROY*, The University of Texas at Dallas, USA
MAHSAN NOURANI*, University of Florida, USA
SHIVVRAT ARYA*, The University of Texas at Dallas, USA
MAHESH SHANBHAG, The University of Texas at Dallas, USA
TAHRIMA RAHMAN, The University of Texas at Dallas, USA
ERIC D. RAGAN, University of Florida, USA
NICHOLAS RUOZZI, The University of Texas at Dallas, USA
VIBHAV GOGATE, The University of Texas at Dallas, USA

We consider the following video activity recognition (VAR) task: given a video, infer the set of activities being performed in the video and assign each frame to an activity. Although VAR can be solved accurately using existing deep learning techniques, deep networks are neither interpretable nor explainable and as a result their use is problematic in high stakes decision-making applications (e.g., in healthcare, experimental Biology, aviation, law, etc.). In such applications, failure may lead to disastrous consequences and therefore it is necessary that the user is able to either understand the inner workings of the model or probe it to understand its reasoning patterns for a given decision. We address these limitations of deep networks by proposing a new approach that feeds the output of a deep model into a tractable, interpretable probabilistic model called a dynamic conditional cutset network that is defined over the explanatory and output variables and then performing joint inference over the combined model. The two key benefits of using cutset networks are: (a) they explicitly model the relationship between the output and explanatory variables and as a result the combined model is likely to be more accurate than the vanilla deep model and (b) they can answer reasoning queries in polynomial time and as a result they can derive meaningful explanations by efficiently answering explanation queries. We demonstrate the efficacy of our approach on two datasets, Textually Annotated Cooking Scenes (TACoS), and wet lab, using conventional evaluation measures such as the Jaccard Index and Hamming Loss, as well as a human-subjects study.

CCS Concepts: • **Computing methodologies** → **Activity recognition and understanding**; **Maximum a posteriori modeling**; • **Human-centered computing** → *User studies.*

Additional Key Words and Phrases: Temporal Models, Dynamic Bayesian Networks, Cutset Networks, Tractable Probabilistic Models

*These authors contributed equally to this work.

Authors' addresses: Chiradeep Roy, The University of Texas at Dallas, Richardson, Texas, USA, Chiradeep.Roy@utdallas.edu; Mahsan Nourani, University of Florida, Gainesville, Florida, USA, mahsannourani@ufl.edu; Shivvrat Arya, The University of Texas at Dallas, Richardson, Texas, USA, shivvrat.arya@utdallas.edu; Mahesh Shanbhag, The University of Texas at Dallas, Richardson, Texas, USA, rayashanbhag@gmail.com; Tahrima Rahman, The University of Texas at Dallas, Richardson, Texas, USA, Tahrima.Rahman@utdallas.edu; Eric D. Ragan, University of Florida, Gainesville, Florida, USA, eragan@ufl.edu; Nicholas Ruozzi, The University of Texas at Dallas, Richardson, Texas, USA, Nicholas.Ruozzi@utdallas.edu; Vibhav Gogate, The University of Texas at Dallas, Richardson, Texas, USA, Vibhav.Gogate@utdallas.edu.

2160-6455/2023/10-ART
https://doi.org/10.1145/3626961

1 INTRODUCTION

Video activity recognition (VAR), which is the task of inferring high-level activities given a sequence of frames, has received increasing attention in recent years. This task is notoriously difficult especially when (1) the number of activities is large; (2) each frame is associated with multiple activities; and (3) activities in different frames depend on each other. Despite the high degree of difficulty, recent advances in deep learning architectures and algorithms have made it possible to accurately solve this task (see, for example, [41, 74, 111, 112]). In particular, we can solve the VAR task using the following approach: (1) predict activities happening in each frame using deep learning based image classification techniques; and (2) make high-level predictions over segments of a video by aggregating predictions over individual frames, leveraging prior knowledge and enforcing constraints to resolve discrepancies (e.g., using a constraint that activities do not change rapidly).

Unfortunately, a problem with the aforementioned approach is that most deep learning models are not explainable; they lack the required probabilistic reasoning tools (see, for example, [45]) that can derive meaningful and faithful explanations for their decisions—for example, see Rudin [102] for a discussion. As a result, in many instances, users are unable to understand why a particular decision was made (by a deep learning model) as compared to a plausible alternative. For example, given a set of *explanatory variables* each of which have a real-world interpretation (e.g., explanatory variable X_1 denotes whether an orange is present in a given frame or not), a possible human-understandable explanation is *the most likely assignment* to a subset of explanatory variables that caused the decision variables to take the predicted values. However, existing neural network technology is unable to accurately and reliably answer such *explanation queries* (see, for example, Gunning and Aha [30]). More specifically, the current technology uses another, possibly simpler (or interpretable) model M which replicates the neural network in order to explain its decisions, without having any guarantees on how closely M mimics the neural network [102]. The main virtue of the approach described in the paper is that it learns a tractable, interpretable probabilistic representation over the explanatory and output variables and derives explanations (via probabilistic reasoning) that are *faithful* to the model.

The lack of explainability is especially problematic in high-stakes applications of VAR and machine learning in general (see, for example [114, 117]). For instance, consider a AI-based video surveillance system at a supermarket that has labeled a particular activity by a shopper as suspicious. Without a human-understandable explanation, the shopper could be wrongly detained if the activity was benign, thus exposing the supermarket to possible lawsuits. On the other hand, in the event that the shopper did commit a crime, the criminal activity may not be evident in the video to a casual observer without a verifiable proof/explanation provided by the system.

To address this issue, we focus on *Explainable Activity Recognition (XAR)*, which we loosely define as the task of inferring high-level activities from videos (or in general, from low-level sensors) along with an explanation of why the activities were chosen in lieu of other activities [30]. An XAR system can benefit and enable a wide variety of real-world applications. For instance, a video surveillance system such as the one described above would greatly benefit from human-understandable explanations that describe why the system flagged predicted activities (happening in specific video segments) as suspicious or benign. These explanations can either be short or detailed. A short explanation, for example, would tag the most important sub-segments in each video segment where the specific (e.g., when a package is stolen from a porch) or a relevant activity (e.g., when a package is touched but not stolen after seeing the surveillance equipment) happened. A detailed explanation, for example, would describe alternate or competing hypotheses along with confidence in each hypothesis and its various components (e.g., the system believes that with a probability greater than 70% that the package on the porch was touched at a later time by the delivery person because he/she forgot to scan the package when it was delivered). Finally, an indirect advantage of explanations is that they help the user better understand how the system works which can potentially improve the user's trust and reliance on the system [32, 33].

The main purpose of this paper is to describe a general approach for XAR and then apply and evaluate it for activity recognition in cooking videos as well as videos from a wet lab (lab dealing with potentially hazardous substances). We also evaluate the explanation quality in cooking videos using human subjects studies.[1] Specifically, we build an XAR system that can perform the following three tasks: (1) parse a video into a set of pre-defined activity labels, namely divide each video into segments and associate activity labels with each segment; (2) use this information to answer Yes/No queries posed by the user; and (3) provide three different types of explanations to add context to the system's answers. The third task, in particular, can only be performed by an explainable machine learning system and is of particular interest to us.

Our system consists of two parts: (1) an explainable machine learning model, which forms the nuts and bolts of our system; and (2) a visual interface which provides answers to user's queries and displays the explanations. Our model, in turn, has two layers, a video classification layer and an explanation layer (see Fig. 1). The video classification (top) layer is a deep neural network that takes video frames as input and predicts an activity label for each frame. The predicted labels are then fed into the explanation (bottom) layer. The latter aggregates the predictions made by the neural network and improves the accuracy using a probabilistic model that represents and reasons about relationships between different activities as well as temporal constraints. The explanation layer provides answers to the queries posed by the user as well as explanations; both tasks are solved by performing inference over the probabilistic model.

The explanation layer consists of a tractable, interpretable probabilistic graphical model [45], specifically a cutset network [90]. 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 are desirable in that they admit accurate linear-time inference and often have the same generalization performance as Bayesian and Markov networks [18, 58, 65, 87, 88, 90, 97]. In other words, inference over cutset networks is always fast and accurate, and as a result they often yield significantly better quality predictions and explanations than Bayesian and Markov networks.

A possible interpretation of our explainable machine learning model is that the deep learning layer provides noisy sensory inputs to the cutset network layer, which in turn removes the noise and provides explanations. The neural network, by itself, is unable to provide explanations because it does not model, and therefore is unable to reason about, the relationships between the predicted labels. On the other hand, a cutset network explicitly models relationships between various activities and can provide fast, high quality explanations by performing (abductive) probabilistic inference over the network. 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.

A key feature of our explainable model is that it is declarative in nature (see, for example, Koller and Friedman [45]) and represents a joint distribution over the explanatory and decision variables given evidence. At a high level, a declarative representation means that the model expresses knowledge or information about a particular domain and for answering queries, either decision or explanation queries, one only needs to change the reasoning algorithm (and not the model). In other words, if the query changes—for instance, if a different type of explanation is sought—our proposed explainable model can use a new general-purpose reasoning method without the need to relearn or modify the underlying model. In contrast, deep learning or discriminative models require learning a distinct representation for each decision or explanation query, which can be computationally expensive. Moreover, as mentioned earlier, because a separate discriminative model is used to generate explanations, which can be quite different from the model used to make decisions, the explanations may not align with the decision (i.e., the explanations may not be faithful to the model that made the decision).

[1]Unlike cooking videos, the wet lab domain is less accessible and requires a domain expert. Therefore, we performed human subjects studies on the cooking domain only.

The second part of our system consists of a browser-based user interface that can be used to pose Yes/No questions to the system (see Fig. 5). The user first chooses a video and can then choose from a list of questions of the form "Did activity *X* take place?" The system then uses the explanation layer to search for frames where there is an activity match and displays explanations in the form of video segments, ranked triples, and component-wise contributions to the explanations. If no perfect matches are found, then the system answers *No* to the query and uses partial component-wise matches to explain its decision.

We evaluated the machine learning part of our system using standard information retrieval metrics such as the Jaccard Index and Hamming Loss (see, for example Leskovec et al. [55]). Specifically, we compared our two-layer architecture, which contains a video classification layer and an explanation layer, with a one-layer architecture that contains only the video classification layer. We observed that the two-layer architecture is more accurate than the one-layer architecture. We evaluated the interface using human-subjects studies where each user was shown a set of videos and presented with questions that she/he had to answer using the explanations provided by the system. Our results clearly demonstrate that the users who used the two-layer system (having the explanation layer) completed the human-machine task faster and more accurately than users who used the one-layer system that did not provide explanations.

In summary, this article makes the following contributions beyond what has been already published by a subset of the authors at archival venues[2] (see [88, 99, 100]):

- We propose a novel two-layer architecture which combines a tractable, interpretable model called dynamic conditional cutset network (DCCN) with deep neural networks for solving the multi-label activity recognition task in video (where the DCCN is defined over the explanatory and output variables). The main virtue of this new architecture is that once learned from data, it is able to generate meaningful, real-time explanations for its decisions via efficient probabilistic inference.
- Via experimental evaluation on the TACoS (third-person videos of a person cooking in a kitchen) and wet lab (third person videos of a person performing lab experiments) datasets, we showed that the two-layered architecture is superior to the one-layered architecture in terms of accuracy.
- We propose and describe a novel interactive visual interface which allows users to load videos, ask queries and visualize the system's answer as well as its explanations.
- Via a human subjects evaluation with 80 users for the task of activity recognition in single-actor cooking videos from the TACoS dataset, we show that the explainable model helps the user answer queries faster and more accurately as compared to the non-explainable model.

The rest of the paper is organized as follows. In the next section, we describe related work. In section 3, we describe the desiderata of an XAR system for both cooking and wet lab videos and show how to build the system using machine learning representations and algorithms in section 4. We empirically evaluate the machine learning models in section 5 and describe the results of a human-subjects study for measuring explanation effectiveness in section 6. Finally, we summarize our contributions and present avenues for future work in section 7.

2 RELATED WORK

2.1 Traditional Space-Time Approaches

Traditional activity recognition has been typically concerned with isolating segments of video and mapping them to particular actions. These traditional space-time approaches treat each video as a 3D volume along the X, Y (spatial) and T (temporal) axes. This 3D volume is then either (a) matched against smaller template volumes for each action [9, 42, 95, 106]; or (b) used to track the trajectories of the agent inside the volume in order to detect specific actions [13, 92, 107, 122]; or (c) used to extract spatio-temporal features or points of interest (eg.

[2]This paper is an extended version of a non-archival paper presented at the 2019 ICML Workshop on Tractable Probabilistic Models [101].

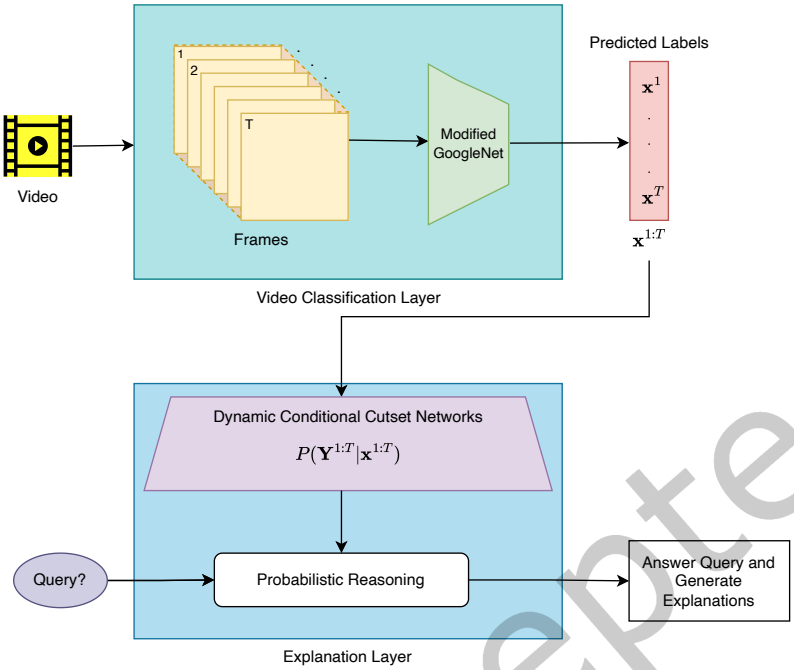


Fig. 1. High-level Architecture and Data Processing Pipeline. Our system has two layers: a video classification layer (see section 4.1) based on deep learning whose output is fed to an explanation layer which is based on dynamic conditional cutset networks (see section 4.2), which are temporal versions of a tractable interpretable probabilistic model called cutset networks [90]. During the learning phase, the classification layer uses the video and the ground truth (labels) as input and learns a mapping from frames to object, action and location. 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 classification layer uses the frames (from video) to provide us with predicted labels for each frame and then these labels are provided to the explanation layer to answer the query by performing marginal and MAP inference over the dynamic cutset network.

changes in directions, collisions, etc.) and use these features to assign actions [8, 15, 19, 50, 51, 77, 105, 118, 124]. A detailed description of these methods can be found in the survey paper by Aggarwal and Ryoo [2]. Unlike previous work, our model encodes additional context about each activity by also modeling the object and location associated with each action.

2.2 State-Based Sequential Approaches

These approaches use Hidden Markov Models (HMMs) and Dynamic Bayesian Networks (DBNs) to model the problem as a sequence of states and then use these sequences to determine the action in question. There exists a large body of work that uses these approaches [10, 12, 29, 35, 63, 73, 78, 81, 110, 121]; however, similar to the traditional space-time approaches, most of these approaches are concerned with single actions only and although they can be used to generate the explanations we use in our system, HMMs suffer from a lack of expressivity while DBNs are impractical either because exact inference over them is intractable or approximate inference algorithms such as particle filtering are inaccurate in practice. The Dynamic Conditional Cutset Network (DCCN)

model we use in this paper circumvents both of these issues and also allows for multiple action/object/location labels per time slice by framing the problem as a temporal multi-label classification (TMLC) problem.

2.3 Deep Learning Approaches

More recently, there have been a number of works that use Convolutional Neural Networks (CNNs) [53], which have already found extraordinary success in large scale image classification and object detection, e.g. [48, 111]. Most of these works use CNNs as spatial feature extractors and then train a recurrent model such as an HMM or an RNN/LSTM on top of these spatial feature vectors to model the sequential nature of activities [20, 26, 27, 31, 37, 52, 59, 61, 67, 75, 79, 108, 119, 120, 123]. Unfortunately, as is typical of deep-learning approaches, these systems are not explainable. The approach used in this paper aims to rectify this problem by training our tractable, dynamic and probabilistic framework to create a transparent and robust decision-making process that can be used to answer any probabilistic query accurately and efficiently as well as to generate explanations to justify its decisions.

There has been another line of work from the variational autoencoder (VAE) community that compose graphical models with neural networks and perform joint learning and inference [38, 40, 47]. The key difference between the formulation used by these models and our formulation is the fact that the graphical models in the former are defined over latent variables while in our model, they are defined over the explanatory variables themselves. This makes our model better suited towards computing the answers to probabilistic queries posed over the explanatory variables that can be presented to the end-user as explanations.

2.4 Event Recognition Models that use Black-box Models as Sensors

In addition to training HMMs and DBNs on top of low-level feature extractors, there have also been a number of grammar-based approaches [36, 39, 104, 125] that use stochastic context-free grammars to model the hierarchical nature of activities as well as the temporal relations between them. The work by Pei et al. [83], for instance, uses a black-box random forest model to predict noisy activities which, in turn, are fed into an AND/OR graph that represents the hierarchical semantic structure of each video. While such approaches are globally interpretable (since they provide a good, hierarchical overview of the event structure in each video), they are unable to efficiently compute conditional probabilities of actions over time e.g. the probability of a slicing activity taking place over an orange in the seventh interval given that the orange was taken out of the fridge in the second interval.

The second class of approaches [11, 68] involve generating candidate interval hypotheses using black-box models and then using a probabilistic relational model to refine these probabilities. 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 [24]. In fact, inference is also intractable in these models unless some very restrictive assumptions are made. For example, the work by Morariu et al [68] restricts the treewidth of the underlying Markov Logic Network (MLN) in order to guarantee tractability of exact inference. Similarly, the work by Brendel et al. [11] defines their own probabilistic event logic; however, computation of the maximum a-posteriori (MAP) is required for inference which is intractable in their model and is approximated using a local search-based algorithm. Our model addresses all these problems since it treats each action, object and location as individual labels and compactly encodes a joint distribution over them. Solving this temporal multi-label classification (TMLC) problem using our dynamic and tractable framework allows for both (a) refining the results of the black-box outputs and (b) generating efficient and accurate posterior probabilities as explanations using particle filtering.

2.5 Semantic Representation of Videos

This effort is also closely related to the work of Rohrbach et al. [96] and Donahue et al. [20] on generating a semantic representation from videos at an activity level using deep learning architectures. Instead of generating sentences in natural language, we assign a number of pre-defined labels divided into categories. Related efforts have considered the task of dense captioning [46], i.e., generating summaries of texts from particular segments. Song et al. [109] attempted to create captioning methods that require minimum supervision on the TaCOS dataset. Duan et al. [22] attempted to combine caption generation and sentence localization to feed off of each other to create a weakly supervised training model. These works focus on creating text summaries for video segments, and as is typical of deep learning approaches, they are essentially black-boxes. Our approach, on the other hand, 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.

2.6 Explainable AI and Machine Learning

Most modern machine learning models use deep neural networks or ensembles as key components to achieve state-of-the-art results. However, due to the opaque nature of these black-box models, decision makers have been reluctant to adopt them in mission-critical applications where the cost of failure is high and where it is imperative to have at least some understanding of the underlying logic used by these models to arrive at their decisions (cf. Veale et al. [116]). There are many types of explanation techniques that are currently in use. The survey by Adadi and Berrada [1] list some of these including visualization techniques such as surrogate models and partial dependency plots (PDPs); knowledge-extraction techniques such as rule extraction and model distillation; influence methods such as feature importance; as well as example-based techniques such as counterfactuals. While these techniques are both powerful and versatile, they are not specifically tailored to the task of activity recognition. There has been some work in this area regarding the usage of association rules [3] and computing feature relevance scores using interpretable models [7] as explanations; however, these systems primarily use sensor data as input (as opposed to video data that is used in this paper). In addition, there have also been some successful attempts in the visual question-answering (VQA) community to generate meaningful visual explanations [57, 80]; however these use still images instead of videos. Very recently, Chen et al. [14] proposed a method that extended the visual question-answering task to videos. However, the aim of most of these VQA methods is to generate natural-language answers to questions posed in natural-language which is significantly different from our aim of building a context-rich semantic representation that can be easily queried. Further, these methods do not incorporate uncertainty very well (for example, they cannot tell the user how confident the model is about certain components of the answer).

2.7 Explainable Activity Recognition (XAR) in Videos

Technically, we can use image attribution approaches (see, for example Grad-CAM [53] and LIME [94]) to perform XAR because each video is a sequence of images. However, these approaches do not model or reason about spatiotemporal relationships and often yield coarse-grained, unsatisfactory explanations [56]. To circumvent this issue, recently, researchers have begun investigating video attribution approaches [5, 56] which output important regions in each relevant frame of a video as explanations. The issue with the above approaches is that they use a separate explainable model or an external process (e.g., loss functions or gradient computations) to explain the original model's decision. These external methods are unable to guarantee that the provided explanation will be close to the optimal explanation. In contrast, the method described in this paper derives optimal or close to optimal explanations by reasoning over a tractable, interpretable probabilistic model defined over the explanatory and output variables.

Our work is also closely related to the work of Zhuo et al. [126] which uses logical reasoning over scene graphs to derive explanations. Our work is different from Zhou et al.'s approach in that we use tractable probabilistic models and reasoning.

2.8 AI and Trust

There have been a number of studies on how trust influences interactions between humans and automated systems, e.g., [32, 54, 69, 70]. These studies examine factors that might affect the trust of the user in the system, such as the past performance of the system and how understandable the system is to the user [54]. Hoffman [33] 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 compare the *demonstrated trust* [66] between two systems (models), one that uses explanations and another that does not by conducting human subjects studies. More specifically, we first divide the participants into two groups and allow participants in the first and second group to observe and interact with a model that uses explanations and a model that does not (use explanations) respectively. Then, we have participants in each group use the system they interacted with to answer yes/no queries about activities performed in the video. Finally, we *assume* that the participant has deemed a system more trustworthy than the other if they are able to answer the query in less time, more accurately and agree with the system's answer in more cases than the other system. An interesting future work is to measure participants' self-reported perception of their trust (via questionaires and surveys) using the trust scale for explainable AI proposed by Hoffmann et al. [34] and compare the results with the three metrics (accuracy, speed and agreement) used in this paper.

3 ACTIVITY RECOGNITION WITH EXPLANATIONS

In this section, we describe the desiderata and assumptions for our proposed explainable activity recognition system. We will describe how we can build such a system using a combination of dynamic conditional cutset networks and neural networks in the next section.

3.1 Desiderata and Assumptions for the Activity Recognition System

We assume that the we are given a collection of third person videos where each video contains exactly one actor performing simple activities such as "open door", "take bottle out of fridge", etc., but not complicated ones such as "cut a banana into ten 1-inch pieces with a knife on a cutting board". Under these assumptions, we define an activity (performed by an actor) as a triple (*action*, *object*, *location*) where *object* and *location* can be "None" but *action* cannot be "None". We assume that the domains of (namely the values that can be assigned to) *action*, *object* and *location* are provided to the system as input.

The *action* component forms the core part of the activity. These are usually verbs such as wash, cut, slice, open, etc. The *object* component denotes the entities over which the activity is performed. These are generally nouns such as apples, refrigerator, cutting board, knife, etc. Finally, the *location* component tells us *where* the activity is taking place. These are generally location nouns such as kitchen, bathroom, counter top, sink, etc. but can also overlap with the nouns we use as objects. For example, when a person "kicks open a door," the activity is "kick" and the object is "door," but the same entity might play a different semantic role in a different activity such as if a person "paints a picture on the door." Here "paint" is the activity, "picture" is the object, and "door" is the location. For reflexive actions, such as "walking," the object and location are "None."

Users interact with our system by posing so-called *selection questions*: "Did a particular (simple) activity defined by the triple (*action*, *object*, *location*) happen in the video?" Examples of selection questions include: (1) "Did the person slice an orange on the counter?" where slice, orange, and counter denote the action, object, and location respectively; (2) "Did the person take out grapes from the refrigerator?" where take out, grapes, and refrigerator

denote the action, object, and location respectively; and (3) "Did the person open the refrigerator?" where open and refrigerator denote action and object respectively and location is None.

3.2 Desiderata for Explanations Provided by the Activity Recognition system

In addition to answering queries posed by the user, we also want the system to provide explanations to justify its answers. We seek three different types of explanations:

(1) **Video Explanations:** When the system answers "yes," we want the system to highlight segments (possibly more than one) of the video where the activity happened. For "no" answers, we want the system to highlight segments where a related activity happened. For example, for the question "did the person in the video wash their hands?", there might be two segments in the video from say, 01:00 to 01:10 and from 04:15 to 04:25 where the person washes their hands. We want our system to detect these segments and use them as explanations to justify its answer to the question ("yes" in this case). If the person does not wash their hands, we would expect the system to answer "no" and explain its answer by highlighting a section of the video (say from 00:20 to 00:35) where the person performs a similar activity such as washing a knife or washing a peeler. The system is expected to therefore justify "no" answers by saying that similar activities were detected but not the specific activity (or activities) that the user was querying about.

(2) **Ranked (action, object, location) Triples:** Given a video explanation (namely a segment of the video), we want the system to display the top-*s* predicted activity triples in the segment that are relevant to the query. For example, for the question "does the person cut a carrot?", the system might answer "yes" and display a list of three possible explanations: *(cut, carrot, cutting-board)*, *(cut, carrot, plate)* and *(cut, orange, plate)*. We want these explanations to be *ordered* in descending order of likelihood (or confidence). In this case, we know that the system believes that the activity taking place was *(cut, carrot, cutting-board)* with the highest degree of confidence, followed by *(cut, carrot, plate)* and *(cut, orange, plate*). These explanations not only provide more context to the answers but also help the user decode patterns in the behavior of the system. For instance, the user might notice that the system frequently generates "orange" as an alternative explanation for "carrot" presumably because they have the same color.

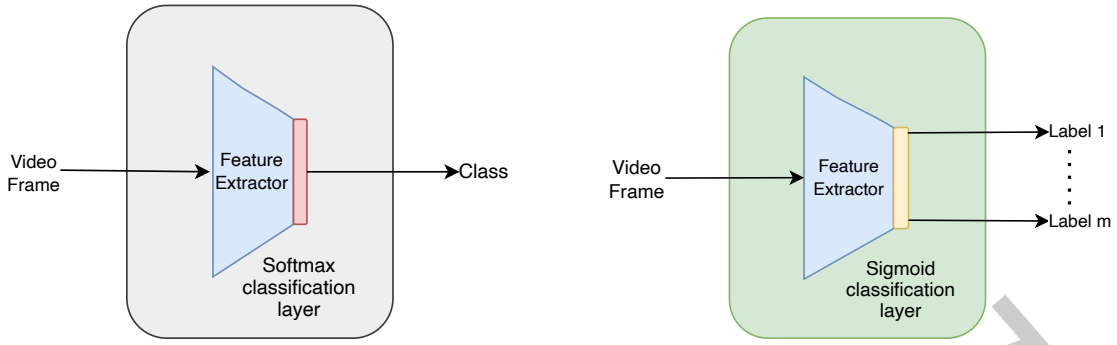
(3) **Most Probable Entities:** Given a video explanation, we want the system to display the most probable actions, objects and locations (along with their likelihood) that are relevant to the query in the video segment. Using the same example as above, we want the system to give us a component-wise score for the components *cut* (action), *carrot* (object), *orange* (object), *cutting-board* (location) and *plate* (location). The system might have a 100% score for *cut* because it is very confident that the action cut happened in the video but only a 60% score for *carrot* because it is not sure if the object being cut is a carrot or an orange.

4 SYSTEM DESCRIPTION

Fig. 1 shows a high-level overview of the components of the system and the processing pipeline. The system can be roughly organized into the following two layers: (a) the *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) the *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.

4.1 Video Classification Layer

The input to this layer is a set of frames (i.e., a video), and the output is a set of predicted activity labels, one for each frame. Each frame is passed through a deep neural network and the latter generates the output activity labels for the frame.



(a) Pre-Trained GoogleNet (b) Modified GoogleNet

Fig. 2. Changes to the GoogleNet architecture. (a) The original GoogleNet schematic which contains a feature extraction module (based on convolutional neural networks) and a softmax classification layer that takes the output of the feature extraction module as input and classifies the input video frame into one of the roughly 1000 possible object categories. (b) Our proposed modification which uses the same feature extraction module as the original GoogleNet but has a fully-connected sigmoid layer having m nodes (labels) that takes the output of the feature extraction module as input and assigns either a 0 (false) or a 1 (true) to each of the m labels.

In our study, we used GoogleNet [111], a 22-layer Convolutional Neural Network (CNN) (see, for example Dumoulin and Visin [23]) that is pre-trained on the ImageNet dataset [103]. Note that any deep neural network can be used instead of GoogleNet; the accuracy of our system will only (likely) improve with more advanced deep learning algorithms and architectures.

We modified the GoogleNet architecture slightly to accommodate our problem. The original GoogleNet, which is trained on the ImageNet dataset, solves a *multi-class classification task* in that it takes an image as input and classifies it into one of the roughly 1000 object categories in ImageNet. In this work, we are interested in solving a *multi-label classification* task where each image can be associated with multiple labels. Specifically, the TACoS and wet lab datasets used in our experiments have 28 and 57 labels respectively. To this end, we replaced the softmax classification layer of GoogLeNet which yields a probability distribution over roughly 1000 object categories with a fully connected sigmoid layer having m nodes, one for each class label (where m equals 28 and 57 for the TACoS and wet lab datasets respectively). Fig. 2 depicts the difference between the original GoogleNet and our modification.

Training: As mentioned earlier, we used a feature extractor that was pre-trained on the ImageNet dataset [103]. Then, we retrained our modified architecture (see Fig. 2(b)) for a fixed number of iterations using the ADAM gradient optimization technique [44] on the TACoS and wet lab datasets respectively.

4.2 Explanation Layer

In this section, we present dynamic conditional cutset networks (DCCNs), our new tractable, temporal probabilistic representation. We use DCCNs in the explanation layer to: (a) correct errors in the labels predicted by 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 abductive poly-time probabilistic inference.

4.2.1 Cutset Networks. Probabilistic Graphical Models (PGMs) such as Bayesian and Markov networks (see, for example Koller and Friedman [45]) are widely used in practice to represent and reason about uncertainty.

At a high level, they are a compact representation of the joint probability distribution over a large number of random variables. Once learned from data, they can be used to answer any query posed over the joint distribution via probabilistic inference. The two main types of inference (queries) tasks are posterior marginal (MAR) and maximum-a-posteriori (MAP) inference. In MAR inference, we are interested in computing the marginal probability distribution over each query variable given evidence where evidence (or observation) is an assignment of values to a subset of random variables. In MAP inference, we are interested in computing the most likely assignment to all query variables given evidence. Both tasks are notoriously difficult to solve in many practical networks, and theoretically they are NP-hard in general [98]. As a result, in practice, one has to often use approximate inference algorithms to solve these problems (approximately). Unfortunately, these algorithms are unreliable and often yield inaccurate query answers [89].

Tractable probabilistic models (TPMs) [4, 17, 62] are special types of probabilistic models which admit poly-time MAR and MAP inference and thus circumvent the problem of unreliability of approximate inference in Bayesian and Markov networks. Although TPMs are less expressive than intractable (latent) probabilistic models such as Bayesian and Markov networks, their prediction 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 Bayesian and Markov networks. Examples of popular TPMs include cutset networks [86, 87, 90], arithmetic circuits [17, 62], sum-product networks [84] and probabilistic sentential decision diagrams [6].

Cutset networks [90] are a class of TPMs that use recursive cutset conditioning [64, 82] to build a rooted OR tree where each non-leaf node corresponds to a conditioned variable and each leaf node corresponds to a tree-structured Bayesian Network defined over all variables not appearing on the path from the root to the leaf. Formally, given a set of variables $\boldsymbol{X} = \{X_1, \ldots, X_n\}$, a cutset network C is a pair (O, T) where O represents an OR tree and T represents a set of tree-structured Bayesian Networks, one for each leaf node in O (see Fig. 3(a) for an example). Assuming that all the variables in $\boldsymbol{X}$ are binary, each non-leaf node in O will have two branches. We will assume that the left and right branches of an OR node labeled by X_i in O correspond to the values $\overline{x}_i$ and x_i respectively where $\overline{x}_i$ (similarly x_i) denotes an assignment of value 0 to X_i (similarly 1). Each directed edge between an OR node labeled by X_i and its child node in O is labeled with the conditional probability of the variable X_i taking the corresponding value given the assignment on the path from the root to X_i. For example, in Fig. 3a, 0.92 equals the conditional probability $P(\overline{y}_4|y_1)$. Every non-leaf node partitions the probability space into data points that agree with $\overline{x}_i$ in the left sub-tree and those that agree with x_i in the right sub-tree. The probability of a full assignment x w.r.t. the cutset network C is given by

$$P_C(\boldsymbol{x}) = T_{l(\boldsymbol{x})}\left(\boldsymbol{x}_{V\left(T_{l(\boldsymbol{x})}\right)}\right) \cdot \prod_{p \in O(\boldsymbol{x})} p \tag{1}$$

where $l(\boldsymbol{x})$ denotes the leaf node in C corresponding to the assignment $\boldsymbol{x}$, $T_{l(\boldsymbol{x})}$ denotes the tree Bayesian network at $l(\boldsymbol{x})$, $V(T_{l(\boldsymbol{x})})$ denotes the set of variables over which $T_{l(\boldsymbol{x})}$ is defined, $x_{V\left(T_{l(\boldsymbol{x})}\right)}$ denotes the projection of the assignment $\boldsymbol{x}$ on the variables $V\left(T_{l(\boldsymbol{x})}\right)$ (where $V\left(T_{l(\boldsymbol{x})}\right) \subseteq \boldsymbol{X}$) and $O(\boldsymbol{x})$ is the set of conditional probabilities on the path from root to the leaf node $l(\boldsymbol{x})$ in the OR tree O.

The time complexity of posterior marginal estimation (MAR) and full maximum-a-posteriori estimation (MAP) is linear in the size of the cutset network and can be solved by making two passes over the network [90]. The fact that most prediction tasks can be reduced to these two types of inference queries makes these models an attractive choice for applications that rely heavily on exact inference at test time.

The structure and parameters of cutset networks can be learned from data using the top-down, recursive induction approach described in Algorithm 1. The algorithm has two main steps: base case and conditioning step. In the base case, the algorithm returns a tree Bayesian network if a pre-defined termination condition (a

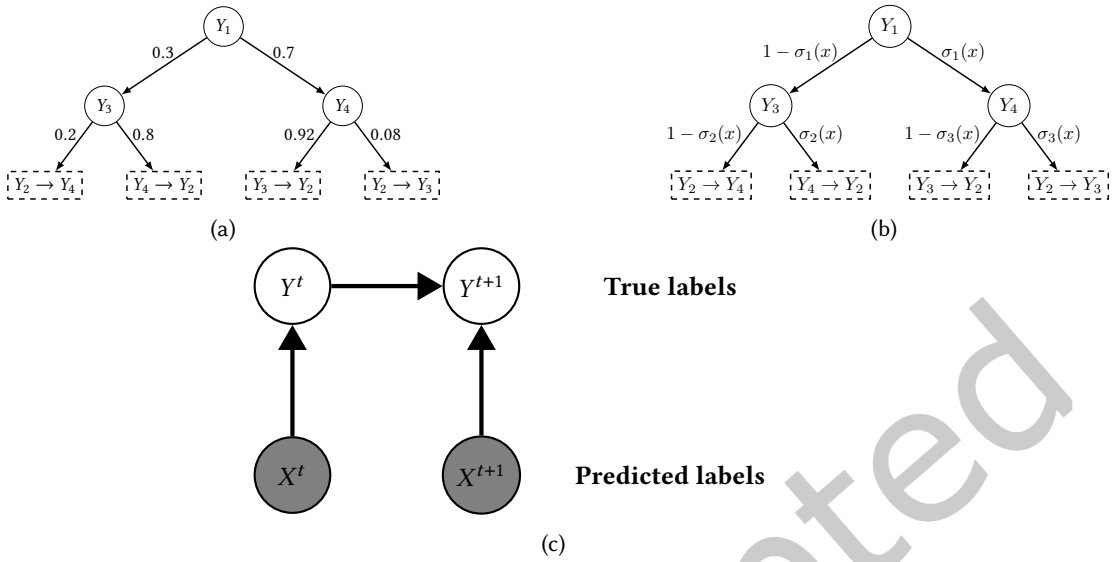


Fig. 3. (a) A cutset network over 4 variables $\{Y_1, \ldots, Y_4\}$. OR nodes are denoted by circles. Y_1 is the root node of the OR tree. Left and right arcs emanating from an OR node labeled by Y_i indicate conditioning over false (assignment of 0) and true (assignment of 1) values of Y_i respectively. Arcs emanating from OR nodes are labeled with conditional probabilities. For example, the arc labeled with 0.08 denotes the conditional probability $P(Y_4 = 1|Y_1 = 1)$. The leaf nodes of the OR tree are tree Bayesian networks. (b) A conditional cutset network (CCN) representing $P(Y_1, \ldots, Y_4|\boldsymbol{X})$. Arcs emanating from OR nodes are labeled with (calibrated) classifier functions. For example, the arc from the OR node Y_1 to the OR node Y_3 is labeled with a logistic regression classifier $1 - \sigma_1(\boldsymbol{x})$. Given an assignment $\boldsymbol{X} = \boldsymbol{x}$ to all variables in $\boldsymbol{X}$, the CCN yields a cutset network having the same structure as the one given in (a) except that the parameters will be computed using σ_1, σ_2 and σ_3. (c) 2-slice dynamic conditional cutset network. The CCN at time slice t represents $P(\boldsymbol{Y}^t|\boldsymbol{X}^t)$ while the CCN at time slice $t+1$ represents $P(\boldsymbol{Y}^{t+1}|\boldsymbol{X}^{t+1}, \boldsymbol{Y}^t)$.

popular condition is described below) is satisfied. The tree Bayesian network is learned from data using the Chow-Liu algorithm [16]. This algorithm first constructs an undirected weighted complete graph in which each node corresponds to a variable X_i in X and each edge (X_i, X_j) is weighed using the mutual information score (MIScore) between X_i and X_j:

$$\text{MIScore}(X_i, X_j) = \sum_{i=0}^{1}\sum_{j=0}^{1} P_{\boldsymbol{D}}(X_i = i, X_j = j) \log \frac{P_{\boldsymbol{D}}(X_i = i, X_j = j)}{P_{\boldsymbol{D}}(X_i = i)P_{\boldsymbol{D}}(X_j = j)}$$

where $P_{\boldsymbol{D}}(X_i = i, X_j = j)$ is estimated from the dataset $\boldsymbol{D}$; the estimate equals the number of times the partial assignment $(X_i = i, X_j = j)$ appears in the data divided by the number of examples in $\boldsymbol{D}$, and $P_{\boldsymbol{D}}(X_i = i) = \sum_{j=0}^{1} P_{\boldsymbol{D}}(X_i = i, X_j = j)$ (similarly, $P_{\boldsymbol{D}}(X_j = j) = \sum_{i=0}^{1} P_{\boldsymbol{D}}(X_i = i, X_j = j)$). Then, the Chow-Liu algorithm finds a maximum spanning tree from the weighted complete graph and converts the tree to a directed tree K using depth-first search. The latter yields a tree Bayesian network which represents the following distribution:

$$T(\boldsymbol{x}) = \prod_{i=1}^{n} P_{\boldsymbol{D}}(x_{\{X_i\}}|x_{pa_K(X_i)})$$

where $pa_K(X_i)$ is the set of parents of X_i in K. Note that $pa_K(X_i) \leq 1$ for all i.

Algorithm 1 LearnCNet

Input: Dataset $\boldsymbol{D} = \{\boldsymbol{x}^{(1)}, \ldots, \boldsymbol{x}^{(m)}\}$ having m training examples
Variables $\boldsymbol{X} = \{X_1, \ldots, X_n\}$
Output: Cutset network C

1: **if** termination condition then
2: **return** ChowLiuTree(D)
3: Use a splitting heuristic to select a variable $X_i \in \boldsymbol{X}$
4: Create a new OR node o and label it by X_i
5: $\boldsymbol{D}_{\overline{x}_i} \leftarrow \{\boldsymbol{x} \in \boldsymbol{D} | X_i = 0\}$
6: $\boldsymbol{D}_{x_i} \leftarrow \{\boldsymbol{x} \in \boldsymbol{D} | X_i = 1\}$
7: Let l and r denote the left and right child nodes of o
8: $l \leftarrow$ LearnCNet($\boldsymbol{D}_{\overline{x}_i}, \boldsymbol{X} \setminus \{X_i\}$)
9: $r \leftarrow$ LearnCNet($\boldsymbol{D}_{x_i}, \boldsymbol{X} \setminus \{X_i\}$)
10: Let $p_{o,l}$ and $p_{o,r}$ denote the conditional probabilities on the edge between o and l and between o and r respectively.
11: $p_{o,l} \leftarrow \frac{|\boldsymbol{D}_{\overline{x}_i}|}{|\boldsymbol{D}|}$
12: $p_{o,r} \leftarrow \frac{|\boldsymbol{D}_{x_i}|}{|\boldsymbol{D}|}$
13: **return** o

The following termination condition is often used in practice [86, 87, 90]: stop growing the OR tree if any of the following conditions are satisfied

(1) The number of examples is smaller than k;
(2) The depth of the OR tree is larger than d (d is bounded by n)

Hyperparameters d and k are tuned using the validation set; namely we search over possible choices of d and k and choose the combination that gives the highest log-likelihood score on the validation set.

In the conditioning step, the algorithm heuristically selects a variable X_i to condition on. The following heuristic is often used in practice [86]. We select a variable having the following sum mutual information score with ties broken randomly:

$$\text{SumMIScore}(X_i) = \sum_{j:j \neq i} \text{MIScore}(X_i, X_j)$$

Once the variable (X_i) is selected, the algorithm induces an OR node o labeled by X_i (line 4). Then, the algorithm partitions the dataset $\boldsymbol{D}$ into two datasets, $\boldsymbol{D}_{\overline{x}_i}$ and $\boldsymbol{D}_{x_i}$ where the former contains only those examples in $\boldsymbol{D}$ which X_i is assigned the value 0 while the latter contains only those examples in $\boldsymbol{D}$ in which X_i is assigned the value 1. It then creates two child nodes l and r and recursively constructs a CN on l and r using $\boldsymbol{D}_{\overline{x}_i}$ and $\boldsymbol{D}_{x_i}$ respectively. Finally, the algorithm estimates the conditional probability on the edges between l and o and between r and o (lines 11 and 12) and returns the OR node o.

4.2.2 Conditional Cutset Networks. Conditional cutset networks (CCNs) are a new framework that was recently proposed by Rahman et al. [88]. As the name suggests, they generalize the cutset networks framework to compactly represent conditional distributions of the form $P(\boldsymbol{Y}|\boldsymbol{X})$ where $\boldsymbol{X}$ and $\boldsymbol{Y}$ are sets of variables. In CCNs, the OR tree and each tree Bayesian network is defined over variables in $\boldsymbol{Y}$. The conditional probabilities in the OR tree and tree Bayesian networks are given by a calibrated probabilistic classifier [76]. These classifiers take as input an assignment $\boldsymbol{x}$ to a set of variables $\boldsymbol{X}$ and output a probability distribution over the class label $Y_i \in \boldsymbol{Y}$. Tractability over each individual distribution is still maintained since the number of parameters for most

Algorithm 2 LearnCCN

Input: Dataset $\boldsymbol{D} = \{(\boldsymbol{x}^{(1)}, \boldsymbol{y}^{(1)}), \ldots, (\boldsymbol{x}^{(m)}, \boldsymbol{y}^{(m)})\}$, Sets of Variables $\boldsymbol{Y}$ and $\boldsymbol{X}$
Output: Conditional cutset network C

if termination condition **then**
  return ConditionalChowLiuTree($\boldsymbol{D}, \boldsymbol{Y}, \boldsymbol{X}$)
Use a splitting heuristic to select a variable $Y_i \in \boldsymbol{Y}$
Create new OR node o and label it by Y_i
Learn a calibrated classifier $\sigma(\boldsymbol{X})$ with class label Y_i and $\boldsymbol{X}$ as input using $\boldsymbol{D}$
$\boldsymbol{D}_{\overline{y}_i} \leftarrow \{(\boldsymbol{x}, \boldsymbol{y}) \in \boldsymbol{D} | Y_i = 0\}$
$\boldsymbol{D}_{y_i} \leftarrow \{(\boldsymbol{x}, \boldsymbol{y}) \in \boldsymbol{D} | Y_i = 1\}$
Let l and r denote the left and right child nodes of o
$l \leftarrow$ LearnCCN($\boldsymbol{D}_{\overline{y}_i}, \boldsymbol{Y} \setminus \{Y_i\}$)
$r \leftarrow$ LearnCCN($\boldsymbol{D}_{y_i}, \boldsymbol{Y} \setminus \{Y_i\}$)
Label the edge between l and o by $1 - \sigma(\boldsymbol{X})$
Label the edge between r and o by $\sigma(\boldsymbol{X})$
return o

calibrated classifiers scales polynomially with the number of input variables $\boldsymbol{X}$. For example, if we use the logistic regression classifier, $P(\boldsymbol{Y}_i = 1 | \boldsymbol{X} = \boldsymbol{x})$ equals $\sigma(w_0 + \sum_{X_i \in \boldsymbol{X}} w_i \boldsymbol{x}_{\{X_i\}})$ where w_i's are the weights (parameters) and σ denotes the sigmoid function.

Given an assignment $\boldsymbol{x}$ to all variables in $\boldsymbol{X}$, a CCN yields a cutset network because each calibrated classifier yields a marginal probability distribution over the class variable. Thus, given $\boldsymbol{x}$, CCNs yield a tractable probabilistic model over $\boldsymbol{X}$. Fig. 3(b) shows an example of a conditional cutset network.

Structure and parameters of a CCN can be learned (see Algorithm 2) using the same top-down induction approach used for cutset networks. The differences between the two algorithms are:

(1) In LearnCCN, we learn the parameters on the edges of the OR tree and the conditional distributions at each node in each tree Bayesian network using a calibrated classifier $\sigma(\boldsymbol{X})$ (e.g., logistic regression, neural networks, random forests, etc.). The best classifier is chosen using cross-validation.
(2) In LearnCCN, we learn the tree Bayesian networks at each leaf node of the OR tree using *conditional mutual information scores*. Similarly, the splitting heuristic in the LearnCCN algorithm uses sum conditional mutual information scores as compared with sum mutual information scores in the LearnCNet algorithm. See Rahman et al. [88] for details.

4.2.3 Using CCNs to Predict Activity Labels. To use CCNs in our video activity recognition framework, we feed the output of GoogLeNet to the CCN. More formally, let $\boldsymbol{X}$ denote the set of output nodes of GoogLeNet and $\boldsymbol{Y}$ denote the set of true labels at a frame. We use the CCN to model $P(\boldsymbol{Y}|\boldsymbol{X})$ and learn its structure and parameters using Algorithm 2. Given a set of videos $\boldsymbol{V}$, the training dataset $\boldsymbol{D}$ is constructed as follows. We have one training example in $\boldsymbol{D}$ for each frame in each video of $\boldsymbol{V}$. Each example is composed of true labels ($\boldsymbol{Y}$) and labels predicted by GoogleNet ($\boldsymbol{X}$) 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 MAP inference over the cutset network to yield the final set of labels. In other words, the CCN treats the output of GoogLeNet as a noisy sensor (see Fig. 3(c)) and computes a conditional joint probability distribution over the true labels given the predicted (noisy) labels. A second benefit of CCNs, apart from the improved accuracy, is that it can be used to generate high quality explanations.

4.2.4 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 dynamic conditional cutset networks (DCCNs). Formally, let a video consist of n frames, let $\boldsymbol{Y}^t = \{Y_1^t, \ldots, Y_m^t\}$ and $\boldsymbol{X}^t = \{X_1^t, \ldots, X_m^t\}$ be the set of true labels and predicted labels (evidence) respectively at frame t. Then, the DCCN represents the following probability distribution:

$$P(\boldsymbol{y}^{1:T}|\boldsymbol{x}^{1:T}) = P(\boldsymbol{y}^1|\boldsymbol{x}^1)\prod_{t=2}^{T} P(\boldsymbol{y}^t|\boldsymbol{x}^{1:t}, \boldsymbol{y}^{1:t-1}) \tag{2}$$

where the notation $\boldsymbol{y}^{1:T}$ (similarly $\boldsymbol{x}^{1:T}$) denotes an assignment of values to all true (predicted) labels in frames 1 to T. We will use the notation $\boldsymbol{Y}^{1:T}$ to denote the set $\bigcup_{t=1}^{T} \boldsymbol{Y}^t$.

The representation given in Eq. (2) is not compact as the number of frames in a video (n) increases. To circumvent this issue, we adopt two standard assumptions widely used in temporal or dynamic probabilistic models—the 1-Markov and stationarity assumptions [85]. 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 $P(\boldsymbol{y}^{1:T}|\boldsymbol{x}^{1:T})$ using

$$P(\boldsymbol{y}^{1:T}|\boldsymbol{x}^{1:T}) = P(\boldsymbol{y}^1|\boldsymbol{x}^1)\prod_{t=2}^{T} P(\boldsymbol{y}^t|\boldsymbol{x}^t, \boldsymbol{y}^{t-1}), \tag{3}$$

where $P(\boldsymbol{y}^1|\boldsymbol{x}^1)$ and $P(\boldsymbol{y}^t|\boldsymbol{x}^t, \boldsymbol{y}^{t-1})$ are conditional cutset networks and $P(\boldsymbol{y}^t|\boldsymbol{x}^t, \boldsymbol{y}^{t-1})$ has the same parameters and structure for $t = 2, \ldots, T$ (see Figure 3c).

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 $P(\boldsymbol{y}^t|\boldsymbol{x}^t, \boldsymbol{y}^{t-1})$, we construct the dataset as follows. Each frame in each video is a training example and is composed of (1) true labels at frame t, namely $\boldsymbol{Y}^t$); (2) true labels at frame $t-1$, namely $\boldsymbol{Y}^{t-1}$; and (3) labels predicted by GoogLeNet at frame t, namely $\boldsymbol{X}^t$ using the pixels in the frame as input.

Inference over DCCNs is intractable in general and can be approximately solved using the particle filtering algorithm [21, 60], a sequential importance sampling algorithm that generates samples from a proposal distribution and estimates the posterior distribution using a weighted average over the generated samples. The performance of the particle filtering algorithm is highly dependent on the quality of the proposal distribution; higher the quality better the estimate. It is known that the ideal proposal distribution is the posterior distribution $P(\boldsymbol{y}^{1:T}|\boldsymbol{x}^{1:T})$. Unfortunately, it is NP-hard to compute in DCCNs [71]. Therefore, we propose to approximate the ideal proposal using the following:

$$P(\boldsymbol{y}^{1:T}|\boldsymbol{x}^{1:T}) \approx P(\boldsymbol{y}^1|\boldsymbol{x}^1)\prod_{t=2}^{T} P(\boldsymbol{y}^t|\boldsymbol{y}^{1:t-1}, \boldsymbol{x}^{1:t}) \tag{4}$$

Thus (see Eq. (4)) we propose to use evidence up to the current time slice to approximate the ideal proposal, namely we use the filtering distribution $P(\boldsymbol{y}^t|\boldsymbol{y}^{1:t-1}, \boldsymbol{x}^{1:t})$ [71] to approximate the posterior distribution $P(\boldsymbol{y}^t|\boldsymbol{y}^{1:t-1}, \boldsymbol{x}^{1:T})$. It is easy to see that this approximation, because it uses evidence up to the current time slice, is likely to yield higher quality estimates as compared to the likelihood weighting approach which uses the prior distribution (namely a distribution that is not conditioned on evidence) as the proposal. Note that $P(\boldsymbol{y}^t|\boldsymbol{y}^{1:t-1}, \boldsymbol{x}^{1:t})$ can be computed in linear time in DCCNs because they use CCNs, which are conditionally tractable models to represent the transition distribution. Algorithm 3 uses this process to generate K particles that we will later use to generate the explanations.

Algorithm 3 GetPosteriorParticlesDCCN

Input: DCCN C, GoogleNet labels $\mathcal{X} = \{\boldsymbol{x}^1, \ldots, \boldsymbol{x}^T\}$ for T video frames, number of particles K
Output: Set of K particles from the filtering posterior distribution $\hat{P_C}(\boldsymbol{Y}^t|\boldsymbol{x}^1, \ldots, \boldsymbol{x}^n)$

for $k \in \{1, \ldots, K\}$ **do**
  $particles[1, k] \leftarrow$ Generate a sample from the prior distribution $P_C(\boldsymbol{Y}^1|\boldsymbol{x}^1)$
end for
for $t \in \{2, \ldots, T\}$ **do**
  for $k \in \{1, \ldots, K\}$ **do**
    $particles[t, k] \leftarrow$ Generate a Sample from the transition distribution

$$P_C(\boldsymbol{Y}^t|\boldsymbol{x}^{t-1}, old\_particles[t-1, k])$$

  end for
end for
return *particles*

4.2.5 Question-Answering and Explanation Generation. As mentioned earlier, the main virtue of CCNs and DCCNs is that unlike GoogleNet and other deep learning/discriminative methods, they can be used to efficiently generate the three different types of explanations described in Section 3.2. Fig. 4 gives a detailed overview of the knowledge compilation and query processing pipeline where each video is first compiled into a tractable probabilistic knowledge base which is later used by the system to answer queries posed to it. In this section, we describe in detail how the explanations for these queries are generated.

To recap, in our proposed system, the user poses a selection type query to the system such as "Does the person in the video cut anything?" *(cut,∗,∗)* or "Does the person do anything with an orange in the sink?" *(∗,orange,sink)*. The fields with ∗'s can be matched with anything. The system then tries to search the video for any activity tuples that match the conditions of the selection query. If a complete match is found, then the system answers *Yes*. If, however, no complete match is found then the system answers *No*. Re-using the previous notation, $\boldsymbol{Y^t} = \{Y_1^t, \ldots, Y_m^t\}$ and $\boldsymbol{X^t} = \{X_1^t, \ldots, X_m^t\}$ are two sets of m binary random variables that denote the set of true labels and noisy labels respectively at time slice t. $\boldsymbol{y^t} = \{y_1^t, \ldots, y_m^t\}$ and $\boldsymbol{x^t} = \{x_1^t, \ldots, x_m^t\}$ denote instantiations of the true and noisy labels respectively. The system then uses CCNs and DCCNs to generate the following types of explanations:

(1) **Video Explanations**: For each unlabeled video, we use Algorithm 3 to compute the joint probability distribution over all possible ground labels at time slice t. Then, we generate the most likely set of labels $\hat{\boldsymbol{y}}^t$ at t by choosing the particle having the highest filtering posterior probability, which is equivalent to performing MAP inference after computing the filtering posterior i.e. $\hat{\boldsymbol{y}}^t = \text{argmax}_{\boldsymbol{y}} P(\boldsymbol{y}^t|\boldsymbol{x}^{1:t})$. Once we have the most likely set of labels for all the time slices, we can cluster frames that have the same set of labels into a single segment. In the worst case, this process might exhibit high variance and result in segments spanning only a few frames; however, this issue can be somewhat circumvented by merging windows containing multiple frames instead of doing it on a frame-by-frame basis. Once this is done, we record the most probable activity triple associated with each segment. When the user submits a selection query, all segments whose activity triples completely match the parameters of the query are returned as video explanations. This helps the user to quickly navigate to the relevant segments of the video where the system believes that the activity took place.
  More interesting, however, are the explanations generated when the system answers *No*. In such a scenario, the system returns all video segments that have partial matches. For example, for the question "Does the

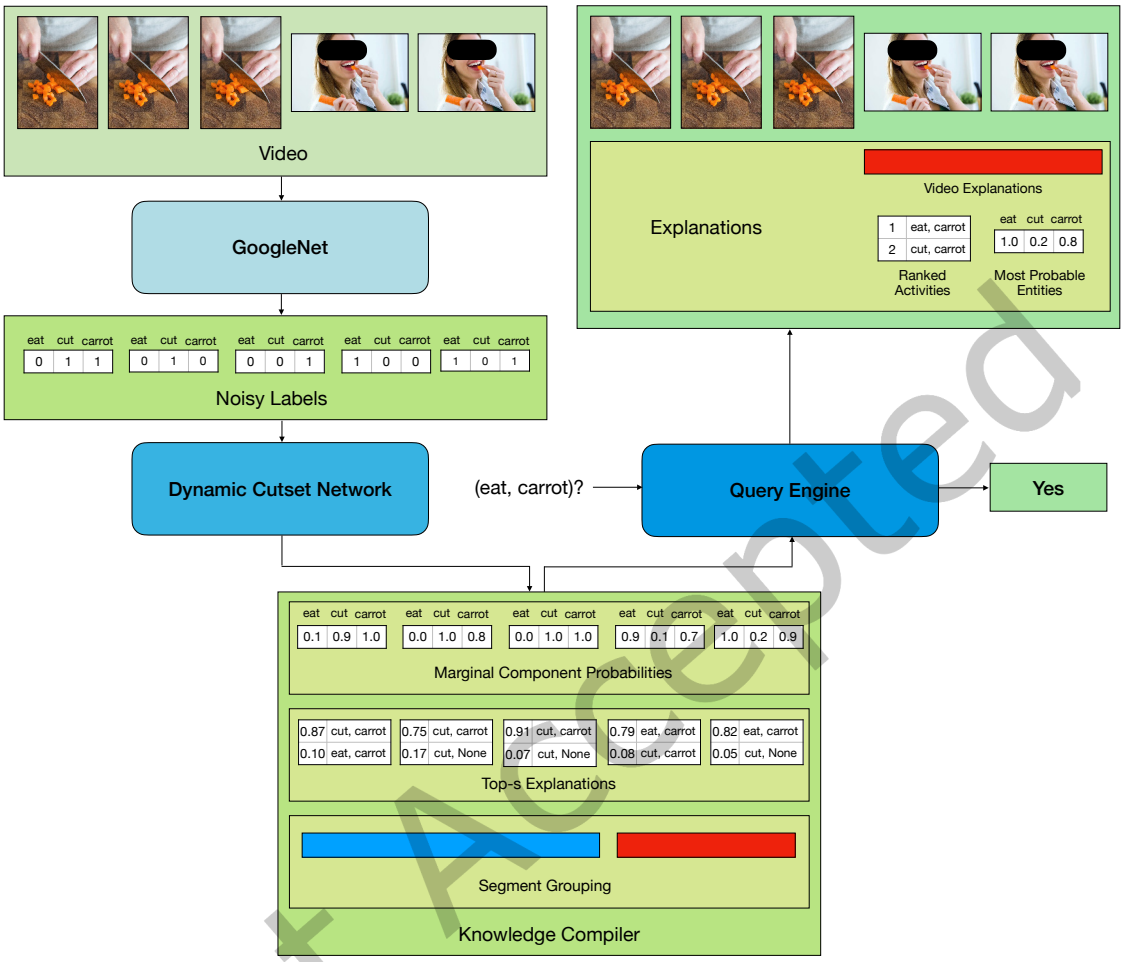


Fig. 4. A brief overview of the knowledge compilation and query processing pipeline w.r.t. a dummy video with five frames having two activities. For this particular example, we define three labels – *eat*, *cut* and *carrot* – and activities as *pairs* consisting of *action* and *object* instead of *triples*. During the *knowledge compilation phase*, each frame is first passed individually through GoogleNet in order to obtain the noisy labels. These noisy labels are then passed through the Dynamic Cutset Network that computes the information necessary for the query processing phrase such as computing the top-*s* activities and marginal label probabilities as well as grouping frames with the same activity into segments. During the *query processing phase*, the query engine searches through all segments that *completely* match the query parameters on *all* components. If such a segment is found, then it answers "Yes" and displays all such segments along with the ranked activities and most probable entities for each segment as explanations. Otherwise, it answers "No" and displays segments with partial matches as explanations.

person take out a knife from the drawer" *(take out, knife, drawer)*, in the absence of a complete match, the system will first try to search for partial matches where at least two of the three components match. For example, if there are segments where the person takes out a peeler from the drawer ***(take out**, peeler, **drawer**)* or takes out a knife from the cupboard ***(take out**, **knife**, cupboard)*, these segments will be returned

as explanations for the *No* answer. The rationale is that while the system found activities similar to the activity being queried, it did not find the exact activity being searched for and therefore answered *No*. If no such partial matches exist, the system then tries to match at least one component such as, say, *(wash, **knife**, sink)* or *(**take out**, carrot, fridge)*. If there are no single component matches either, then the system displays that no segments are found.

Note that video explanations can also be generated using video moment retrieval methods [26, 27, 31, 37, 59, 61, 67, 79, 119]. At a high level, the goal of video moment retrieval (VMR) is to take a video and a natural language sentence describing an activity as input and output a segment in the video that is relevant to the activity. However, as mentioned earlier a drawback of using these methods is that the output of the model that generates explanation may not align with the answer to the query. A second issue is that these discriminative VMR models do not have the representation and reasoning power to generate other types of explanations.

(2) **Ranked (action, object, location) Triples:** The particles/samples output by Algorithm 3 can also be used to generate ranked explanations for any given frame. In particular, if we want to compute the top-s most likely activities at time slice t, then we can select s particles having the highest likelihood at time slice t and display them to the user in descending order of likelihood scores; this is equivalent to performing s-MAP inference on the filtering posterior distribution. We can approximate this by sampling directly from $P(\boldsymbol{y}^t|\boldsymbol{x}^{1:t})$. Since video segments are returned as explanations to the system's answers and each video segment is associated with a single activity triple, we take the average of all the particles over a segment and return the top-s particles having the highest average likelihood as ranked triples.

(3) **Most Probable Entities:** Once again, these kinds of explanations can also be generated by Algorithm 3 using an approach similar to the one used for ranked triples. The only difference is that instead of generating s-MAP tuples, we wish to compute the marginal distribution $P(y_i^t|\boldsymbol{x}^{1:t})$ for each true label $y_i^t \in \boldsymbol{y}^t$ that will tell us how confident the system is about a particular label at a given time slice. Since the last step of particle filtering involves generating K instantiated cutset networks, we can simply perform exact inference on these networks (which can be done tractably and in fact, linearly) to compute the posterior marginals and then average out over all the K networks.

4.3 User Interface

Our prototype system uses an interactive visual interface that allows users to load videos, ask queries, and review the model output along with 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. For this reason, the system uses simple visual representations in the form of graphical annotations, textual component lists, and simple bar charts. Figure 5 shows the interface.

The interface allows the users to select a video and a relevant query based on that video. These would serve as inputs for the model. The interface would then provide two types of output: (1) the model's answer to that query and (2) the explanations for why the answer is provided. For this purpose, the interface includes a video player with the selected video that would allow the users to go over the video if they wanted to review and analyze the system's answer to the query or try to come up with their own answer for that query. The queries used in this system are in the form of yes/no questions, and hence, the answers are either *yes* or *no*.

The system also provides information to justify its answer to the selected query. For this purpose, it would first highlight the most relevant segments (a sequence of relevant and consecutive frames) in the video through visual annotations added under the video progress bar. These annotations are in the form of square-shaped blue buttons, as seen under the video in Fig. 3. These annotated segments are buttons, and when clicked, the video

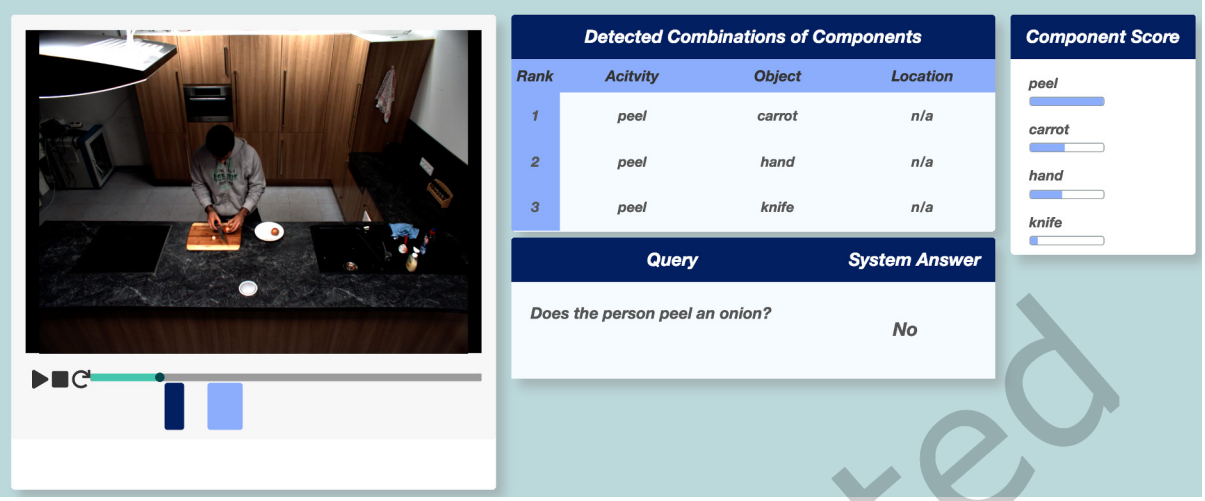


Fig. 5. The interactive visual interface allows users to load videos and ask queries. The interface shows the ML system's answer along with explanatory elements for the output. The most relevant portions of the video play time are shown by colored bars beneath the video, and the right side shows detected video components and combinations of components relevant to the video and query.

jumps to the start of that segment, and new information is loaded that explains why this segment is relevant to the query answer. By default, the first identified segment is selected.

Each segment includes detailed explanations as to why it is related to the system answer. We display this information on the right-side of the interface, as seen in Fig. 3. On the top right, the summary of detected video components (*action*, *objects*, and *locations*) for the given query is shown, which represent the highest ranking combinations of components the model detected in this segment. On the bottom right, the interface shows the component scores that summarize the marginal probabilities of single components in the selected segment. To help users to quickly judge 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.

5 MODEL EVALUATION

The model for the activity recognition system was trained on two datasets - the Textually Annotated Cooking Scenes (TACoS) dataset [93] and the Wetlab [72] dataset.

- **The TACoS dataset** consists of videos of several different cooking-related activities. For example, a typical video will have a person take out a vegetable from the refrigerator, wash it, cut it, and then cook it. The cooking context has the advantage of being easily understandable, even without particular domain expertise. The dataset includes hand-annotated labels of actions, objects, and locations for each frame of video. We isolate 28 such labels and use videos with only these labels for our experiments. Most videos are around 2 minutes in length (although videos as long as 15 minutes are also present). We used different sets of videos for training and testing; specifically 14 videos (60,313 frames) for training and 3 videos for testing (9,355 frames).
- **The Wetlab dataset** contains videos of technicians performing experiments in a wet laboratory with hazardous chemicals. In such settings, it is very important to ensure that correct protocol is being followed,

which is the motivation for using these videos for our system. The dataset consists of 12 videos spread out over the following three protocols: Cellobiose M9 Media (CELL), LB Liquid Growth Media (LLGM), and Yeast YPAD Media (YPAD). Out of the 12 videos, only 6 of them were annotated which is what we used for our experiments. A total of 57 labels were isolated and the training and test sets were comprised of 100,054 (5 videos) and 11,743 (1 video) frames respectively.

Training was carried out in two phases. In the first phase, we swapped out the softmax layer in GoogleNet with a fully-connected layer having sigmoid cross-entropy loss. We then trained it on the training labels by using backpropagation for around 50k iterations using the Adam optimizer [44]. In the second phase, we trained both (a) an intra-frame sensor model (CCN) that modeled the error between the noisy labels of GoogleNet and the ground labels; and (b) an inter-frame transition model (dynamic CCN) that modeled persistence between labels over multiple frames. During testing, we computed the MAP tuple for each frame in each video using particle filtering with 100 particles. We performed the following ablation study: (1) our system in which the explanation layer is removed (GoogLeNet); (2) our system which uses (static) conditional cutset networks in the explanation layer (CCNs); and (3) the full system (dynamic CCNs).

Table 1 outlines the accuracy scores for correct activity recognition according to various evaluation metrics. We use standard measures such as the Hamming Loss and the Jaccard Index (see, for example Leskovec et al. [55]). Hamming loss is defined as the average fraction of incorrect labels (smaller the better). Jaccard index is the ratio between the cardinality of the intersection of the predicted labels and the true labels and the cardinality of the union of the predicted and true labels (higher the better).

We observe that for both our datasets, the full dynamic model (dynamic CCNs) gives the best results. Even if we ignore the temporal component and use only the sensor (CCN) model for each time slice independently, we usually see an improvement in scores. As seen in Table 1, the differences are more apparent in the Wetlab dataset than they are on the TACoS dataset. We surmise that this is because of twoe reasons. First, the number of labels for the Wetlab dataset (57) is more than double that of the TACoS dataset (28). Secondly, the objects in the TACoS dataset are distinct and easily recognizable while those in the Wetlab dataset comprises mostly of test tubes and bottles of different shapes and sizes. This presumably makes it difficult for the neural network to properly distinguish between these objects and assign the correct labels to them.

Thus, our results clearly show that reasoning about relationships between the various labels via CCNs and temporal constraints via DCCNs improves the accuracy of deep neural networks. One thing that needs to be noted here is that higher scores in each of our evaluation metrics ensure greater quality of explanations. This is because the metrics are being computed w.r.t. the MAP tuple which denotes the most likely activity at each frame. Further, these activities are grouped together into frame segments which, in turn, are presented to the end users as explanations for a yes/no selection query. In other words, although video segment explanations can also be generated from GoogleNet by grouping similar multi-label assignments together (as alluded to in Section 4), our joint model would result in explanations of significantly higher quality as evidenced by its superior performance shown in Table 1.

Note that in contrast to the video moment retrieval (VMR) techniques described in the literature [26, 27, 31, 37, 59, 61, 67, 79, 119], which typically employ segment-level evaluation metrics (e.g., `Recall@n,IOU=m` which is defined as the percentage of at least one of the top n predicted moments which have Intersection over Union (IoU) with ground-truth moment larger than m [25]), we opted for frame-level evaluation metrics (e.g., Jaccard Index, Hamming loss, etc.). This choice was driven by the fundamental characteristics of our underlying model, a modified GoogLeNet, which operates at the frame level. To clarify, our model takes individual frames as input and generates predictions for actions, objects, and locations.

Furthermore, we want to point out that in contrast to state-of-the-art VMR techniques that leverage advanced deep learning architectures such as 3D CNNs [27], which can capture temporal relationships among video frames,

Table 1. Accuracy for Activity Recognition on Test Videos. Bold results indicate the best performing model.

Dataset	Metric	GoogLeNet	CCNs	Dynamic CCNs
TACoS	Jaccard Index	0.8608	0.8559	**0.8674**
	Hamming Loss	0.1392	0.1286	**0.1160**
Wetlab	Jaccard Index	0.6299	0.7564	**0.8308**
	Hamming Loss	0.0142	0.0118	**0.0083**

our study relies solely on a less advanced frame-level 2D CNN model derived from GoogLeNet. More specifically, our study's primary objective is not to demonstrate superiority over existing VMR techniques. Instead, our aim is to showcase that our explainable model, which can easily be integrated with other deep learning approaches, does not degrade performance. In fact, as our results indicate, especially when GoogLeNet serves as the base model, our approach can lead to improved performance. This improvement can be attributed to our model's capability to reason not only about time but also about the relationships between actions, locations, objects, and the query.

6 HUMAN SUBJECTS EVALUATION

To evaluate the overall effectiveness of the explanations in our video activity recognition system, we designed and conducted a human-centered experiment.

6.1 Goals and Hypotheses

Video activity recognition (VAR) systems are valuable and have many real-world applications, from fire detection [49] and airport security [113], to elderly care [43] and autonomous vehicles [91]. As alluded to in the introduction, many state-of-the-art models exist that yield high accuracy on the VAR task. However, no matter how accurate the model, these kinds of models typically suffer from false positives, which may be highly undesirable in mission-critical applications. At the very least, as users of these systems, we would like to predict the circumstances under which the system would be likely to generate erroneous results and if it does, what these results might look like. Thus, human-AI collaboration plays an important role in identifying the weaknesses of such systems. For this purpose, it is crucial that human users maintain a proper understanding of the systems and how they work in order to understand when to rely on them. This is the problem we expect to be addressed by the Explainable Activity Recognition (XAR) framework that we defined earlier.

The goal of this study was to measure the degree to which the explanations generated by our system would benefit non-expert end users with little to no understanding of how such systems work. We hypothesized that given a set of videos and several yes/no queries where each query asks whether an activity was performed in a video or not, the time required by the user to answer the query as well as the error rate over all queries would decrease significantly with the introduction of explanations. We also hypothesized that the user's level of agreement with the system's answers would vary significantly if explanations were shown. A user's answer is said to 'agree' with the system's when they are the same.

6.2 Experimental Design

The task used for evaluating user performance involved answering a series of questions (or queries) over a set of videos with the help of the system. The participants were divided into two groups: one *with* and the other *without* explanations. Participants from both groups had access to the video player and the system's answer to each question. Participants in the *with explanations* category used the interface with all the explanation components

available (i.e., the video segments, the detected combination of components, and the component scores), while participants in the *without explanations* category did not have these components shown (i.e. they were only able to view the system's answers but no explanations). The experiment was conducted between-subjects in order to allow the same set of questions and videos to be used for both groups and also to avoid learning effects about knowledge of the model performance.

The TACoS dataset that was used for training and evaluating our system was used since cooking videos typically do not require any domain-specific knowledge. Each user was presented with 20 queries spread out over 4 videos (i.e., 5 queries per video). Each query was a simple yes/no question about the video and had a single, unique answer (e.g., "Does the person cut a carrot?"). Participants of each group were presented with the system's answer to each of these queries and their task was to determine the true answer by watching the video and using the system's answers as point of reference.

Since the goal of the study was to evaluate whether the explanations helped the participants perform better than the model alone, it was necessary that the system made enough errors so that the explanations would actually be useful to determine the actual answer and that this improvement could be measured. It was, therefore, imperative that the task include multiple queries where the system provided incorrect answers so that participants would have opportunities to recognize system errors. However, since the actual model had a high accuracy score (The accuracy of our system was > 92%, also refer to Table 1 which show two other measures related to accuracy, the Jaccard Index and Hamming Loss), using a set of sample queries representative of the actual model accuracy would not have provided enough opportunities to view system errors since the system would have been too accurate for participants to see enough errors in the limited time of the study. To address this problem, we constrained the system accuracy to a constant 80% for this phase of the evaluation. This was done by controlling the composition of trials so that all participants experienced the system answering 80% of the queries correctly.

To assess task performance, we used the following three metrics: (1) error (2) time taken for task completion and (3) agreement. Error was calculated as the percentage of queries where the participant's answer was incorrect with respect to the (known) true answer. Agreement was measured as the percentage of queries where the participant's answer matched that of the system.

6.3 Procedure

The interface was a web-based application and the evaluation was conducted as an online study. We ran the experiment through the Amazon Mechanical Turk (AMT) crowdsourcing platform. The study was approved by our organization's Institutional Review Board (IRB), and participants were compensated at a fixed rate per hour. The experiment consisted of a single session with completion time of the participants varying from 25 minutes to 55 minutes (Note that the total session time includes time required to complete the questionnaires and demographic information).

The study opened with a consent form followed by a background questionnaire asking general information about participant demographics, education, and occupation. The participants were also provided with instructions on how to complete the task as well as a small tutorial prior to the main query review trials.

We slightly modified the interface described in Section 4.3 to (1) control the sequence of viewed videos and queries, and (2) include buttons for participants to provide their own answer for each query. Thus, instead of allowing participants to choose from the set of existing videos and queries, the interface only showed one video and one corresponding query at a time. For each trial, the system's answers were available, but the participants were asked to answer each query themselves. After providing their answer, the system showed participants the correct answer (which was sometimes different from the system's answer to simulate 80% accuracy as mentioned earlier) as feedback to help them estimate the system's simulated accuracy as well as build an understanding of

Table 2. Contingency tables for the (a) *with explanation* and (b) *no explanation* categories showing the proportion of instances where the user's responses were correct/incorrect when the system they had access to made correct/incorrect predictions.

		User Correct	
		Yes	No
System Correct	Yes	74.61 ± 3.37	5.39 ± 3.37
	No	15.66 ± 4.38	4.34 ± 4.38

(a) With Explanations

		User Correct	
		Yes	No
System Correct	Yes	70.625 ± 6.01	9.375 ± 6.01
	No	16.25 ± 4.63	3.75 ± 4.63

(b) No Explanations

how the system made its decisions which, in turn, would help them decide whether or not to rely on its answers. Participants were not allowed to change their response after submission.

6.4 Participants

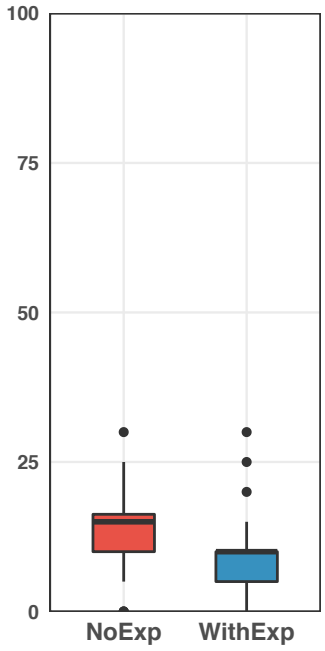
The experiment was completed online by 80 AMT workers. Of these participants, 40 of them were shown explanations while the other 40 were not. With the exception of a single participant who reported himself to be a programmer, all the others had occupations that were not related to data science, machine learning, and statistics; hence, the participants were non-experts with regards to machine learning knowledge. After pre-processing 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.

6.5 Results

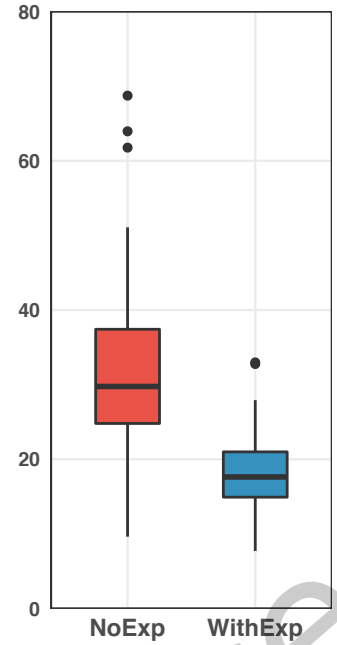
Because the collected data is not normally distributed—verified using the Shapiro-Wilk test and graphically via a histogram—we analyzed the results using the Kruskal-Wallis non-parametric test to measure the difference between the two groups. The plots are shown in Fig. 6. We observed a significant difference on error per trial $\left(\chi^2(1,76) = 5.63, p < 0.05\right)$, 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 $\left(\chi^2(1,76) = 28.1, p < 0.001\right)$. Participants *with explanations* were significantly faster. Together, these results support our hypothesis that the addition of our explanations significantly improves user task-performance in our system.

We also observed that participants *with explanations* agreed with the system significantly more than their counterparts $\left(\chi^2(1,76) = 8.00, p < 0.01\right)$. This observation is aside the fact that both groups showed high agreement levels with the system given its high accuracy (80%), as the mean and standard deviation is shown in Fig. 6 (d).

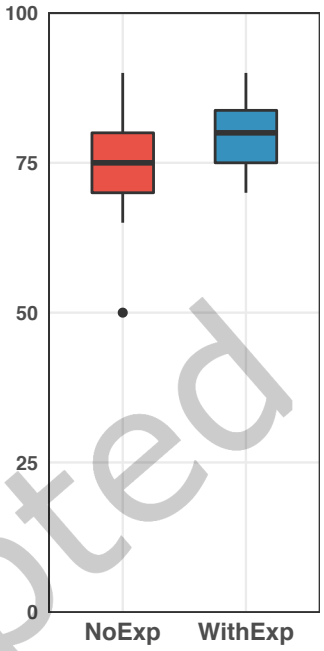
We can further analyze user agreement by using the contingency table (confusion matrix) presented in Table 2. In cases where the system is correct, we observe that the users in the *with explanations* group agreed with the system significantly $\left(\left(\chi^2(1,76) = 12.01, p < 0.001\right)\right)$ more than those in the *no explanations* group. This implies that providing more information helped participants understand the system and judge when it was correct; namely users *appropriately* placed their trust in the *with explanations* system, in contrast to the *no explanations* system and the explanations encouraged participants to correctly rely on its output. Conversely, when the system was incorrect, on average, users in the *with explanations* category tend to agree with the system more than users in the *no explanations* category. However, there was no significant difference $\left(\left(\chi^2(1,76) = 0.61, p > 0.43\right)\right)$ in the user agreement when the system was incorrect, suggesting that no significant unwarranted (or inappropriate) trust is developed. Based on this analysis, it can be inferred that users in the *with explanations* category frequently develop appropriate trust with the system compared to users in the *no explanations* category.



(a) **Average Error per Trial (Percentage)**



(b) **Average Time per Trial (Seconds)**



(c) **User Agreement with the System (Percentage)**

Metric	No Explanation	With Explanation
Average Error Per Trial	13.12% ± 7.4%	**9.74% ± 6.03%**
Average time per Trial (seconds)	986.25 ± 561.35	**517.42± 198.1**
User Agreement with System	74.38% ± 7.78%	**78.95%± 4.95%**

(d) Mean and standard deviation for performance and agreement of novice participants.

Fig. 6. The plots show the distribution of user responses based on user task-performance and agreement with the system among our two study conditions (NoExp: *no explanations* and WithExp: *with explanations*). Lower scores for the first two measures indicate better performance, i.e., lower errors and less performance time per each trial while higher scores for agreement indicate higher reliance on the system for answering queries. Table shown in (d) is a summary of findings through mean and standard deviation of each metric. These findings align with the results from Kruskal-Wallis non-parametric test reported in section 6.5. We measured the average user error and time per trial and the fraction of instances on which their answer agreed with the system's answer. Bold results indicate significantly higher score.

Finally, 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* [28], where humans tend to trust an intelligent system by virtue of its 'intelligence' alone. Rather, the results of this study suggest that agreement was appropriately aligned with the queries where the system provided the correct answer. However, it is to be noted that our study was not designed to specifically focus on the potential effects of explanations on automation bias, and therefore this still remains an open area for further research.

7 DISCUSSION AND CONCLUSION

In this paper, we proposed a new explainable framework for activity recognition (AR), which we call explainable activity recognition (XAR). We surmised that such a framework would use Explainable Artificial Intelligence (XAI) techniques to provide enough model transparency to the users such that it will allow them to: (1) build a good mental model of how the system functions; (2) use and interact with the system more effectively, specifically understanding when it will succeed and when it is likely to fail; and (3) improve reliance (and possibly trust) on the system.

We then proposed a general approach for building an XAR/XAI system. Our approach uses the following pipeline:

(1) build an accurate model for activity recognition using deep neural network architectures and learning algorithms
(2) build a tractable probabilistic model over the interpretable random variables in the application domain using the output of the deep learning model as input treating the latter as a noisy sensor; the tractable model further improves the accuracy of the deep learning model
(3) answer queries posed by the user and generate explanations by performing probabilistic inference over the tractable model; tractability ensures that query answers and explanations are accurate and can be generated in real-time

We used this general approach to build an XAR system for activity recognition and applied it on two datasets, the TACoS cooking video dataset [96] and the wet lab dataset [72]. In both datasets, we assumed that the activity is defined as a *(action, object, location)* triple. Our system had two-layers; the first layer used a deep convolutional neural network called GoogLeNet [111] and the second layer used a new tractable model called dynamic conditional cutset networks (DCCNs). The latter is a novel representation which extends and generalizes the recently proposed conditional cutset networks representation [88] to temporal domains.

In our system, GoogLeNet helped detect complex spatial patterns in each frame of each video while the DCCN helped capture the relationships between the various activities as well as temporal dynamics. The DCCN answered queries posed by the user and provided explanations via fast, accurate probabilistic inference. It also helped decipher the output of the black-box GoogLeNet architecture by summarizing and aggregating its decisions (see "Video explanations" in Fig. 5), suggesting alternative hypotheses that are likely to be true (see "Detected Combinations of components" in Fig. 5) and providing confidence on its detected components (see "Component Score" in Fig. 5).

We evaluated our system along two dimensions: prediction accuracy and explanation effectiveness. Via a thorough ablation based empirical evaluation, we found that the "explainable model" which combines a DCCN and a neural network is superior in terms of prediction accuracy to a "non-explainable model" which only uses a neural network. This verifies our hypothesis that DCCN corrects the errors made by the neural network by leveraging temporal information as well as relationships between activities. The usefulness of our explanations was also corroborated by the user studies where explanations helped the user solve simple yes/no question-answering tasks more accurately and with greater ease (measured using time required to complete the given task).

7.1 Future Work

Although our new dynamic conditional cutset network framework generates high-quality samples from the posterior distribution, both MAR and MAP inference over it are intractable (or NP-hard) in general. To this end, one line of future work is to investigate temporal models on which both MAP and MAR inference tasks can be solved in polynomial time. We are currently investigating specific structural constraints for achieving this objective building on our prior work [100] on this topic. A second avenue for future work is to combine DCCNs

with more sophisticated deep learning architectures such as 3D CNNs [27] and transformers [115] and evaluate whether the combined approach improves (or at least does not degrade) performance or not.

In this paper, we only considered simple selection queries with yes/no answers. An interesting direction to expand upon would be to use more complex kinds of queries such as counting queries (e.g., *how many carrots are there in the video?*) and queries where a given task can be accomplished in multiple ways (e.g., using a spoon instead of a spatula to stir soup) which would require the use of common-sense reasoning. In order to achieve this objective, we will have to develop a novel representation for activities and build an ontology to represent hierarchies of activities. This would allow for other interesting queries involving super-activities and sub-activities. A simple query of this type might be something like "Does the person in the video cook a potato?" which might consist of the sub-activities *(cut, potato, ∗)*, *(move, potato, pot)*, *(move, pot, stove)*, *(turn on, stove, ∗)* and *(turn off, stove, ∗)*. Since these are cooking videos, we might even ask the system if the person in the video follows the recipe correctly. Once our system is able to answer these kinds of complex queries, we could then think about more varieties of explanations that might be tailored to specific kinds of queries. Finally, we would re-design our human studies to accommodate these new queries and explanations and evaluate their usefulness to different categories of end users.

8 ACKNOWLEDGEMENTS

This work was supported by the DARPA Explainable Artificial Intelligence (XAI) Program under contract number N66001-17-2-4032, and by the National Science Foundation grants IIS-1652835, IIS-1528037, and IIS-1762268. We would also like to thank four anonymous reviewers whose comments and suggestions helped improve and clarify this manuscript.

REFERENCES

[1] Amina Adadi and Mohammed Berrada. 2018. Peeking inside the black-box: a survey on explainable artificial intelligence (XAI). *IEEE access* 6 (2018), 52138–52160.

[2] Jake K Aggarwal and Michael S Ryoo. 2011. Human activity analysis: A review. *ACM Computing Surveys (CSUR)* 43, 3 (2011), 1–43.

[3] Martin Atzmueller, Naveed Hayat, Matthias Trojahn, and Dennis Kroll. 2018. Explicative human activity recognition using adaptive association rule-based classification. In *2018 IEEE International Conference on Future IoT Technologies (Future IoT)*. IEEE Press, Boston, MA, USA, 1–6.

[4] Francis R Bach and Michael I Jordan. 2002. Thin junction trees. In *Advances in Neural Information Processing Systems*. MIT Press, 569–576.

[5] Sarah Adel Bargal, Andrea Zunino, Donghyun Kim, Jianming Zhang, Vittorio Murino, and Stan Sclaroff. 2018. Excitation Backprop for RNNs. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. IEEE, Salt Lake City, UT, USA, 1440–1449.

[6] Jessa Bekker, Jesse Davis, Arthur Choi, Adnan Darwiche, and Guy Van den Broeck. 2015. Tractable Learning for Complex Probability Queries. In *Advances in Neural Information Processing Systems*, Vol. 28. Curran Associates, Inc., 2242–2250.

[7] Claudio Bettini, Gabriele Civitarese, and Michele Fiori. 2021. Explainable Activity Recognition over Interpretable Models. In *2021 IEEE International Conference on Pervasive Computing and Communications Workshops and other Affiliated Events (PerCom Workshops)*. IEEE, IEEE, 32–37.

[8] Moshe Blank, Lena Gorelick, Eli Shechtman, Michal Irani, and Ronen Basri. 2005. Actions as space-time shapes. In *Tenth IEEE International Conference on Computer Vision (ICCV'05) Volume 1*, Vol. 2. IEEE, 1395–1402.

[9] Aaron F Bobick and James W Davis. 2001. The recognition of human movement using temporal templates. *IEEE Transactions on Pattern Analysis & Machine Intelligence* 23, 3 (2001), 257–267.

[10] Aaron F. Bobick and Andrew D. Wilson. 1997. A state-based approach to the representation and recognition of gesture. *IEEE Transactions on Pattern Analysis and Machine Intelligence (PAMI)* 19, 12 (1997), 1325–1337.

[11] William Brendel, Alan Fern, and Sinisa Todorovic. 2011. Probabilistic event logic for interval-based event recognition. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. 3329–3336.

[12] Hilary Buxton and Shaogang Gong. 1995. Visual surveillance in a dynamic and uncertain world. *Artificial Intelligence* 78, 1-2 (1995), 431–459.

[13] Lee W Campbell and Aaron F Bobick. 1995. Recognition of human body motion using phase space constraints. In *Proceedings of the IEEE International Conference on Computer Vision*. 624–630.

[14] Xuanwei Chen, Rui Liu, Xiaomeng Song, and Yahong Han. 2021. Locating Visual Explanations for Video Question Answering. In *International Conference on Multimedia Modeling*. Springer, 290–302.
[15] Olivier Chomat and James L Crowley. 1999. Probabilistic recognition of activity using local appearance. In *Proceedings. 1999 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (Cat. No PR00149)*. 104–109.
[16] Chao-Kong Chow and Chao-Ning Liu. 1968. Approximating Discrete Probability Distributions with Dependence Trees. *IEEE Transactions on Information Theory* 14 (1968), 462–467.
[17] Adnan Darwiche. 2000. A Differential Approach to Inference in Bayesian Networks. In *Proceedings of the Sixteenth Conference in Uncertainty in Artificial Intelligence*. 123–132.
[18] Nicola Di Mauro, Antonio Vergari, and Floriana Esposito. 2016. Multi-label classification with cutset networks. In *Conference on Probabilistic Graphical Models*. 147–158.
[19] Piotr Dollár, Vincent Rabaud, Garrison Cottrell, and Serge Belongie. 2005. Behavior recognition via sparse spatio-temporal features. In *2005 IEEE International Workshop on Visual Surveillance and Performance Evaluation of Tracking and Surveillance*. 65–72.
[20] Jeffrey Donahue, Lisa Anne Hendricks, Sergio Guadarrama, Marcus Rohrbach, Subhashini Venugopalan, Kate Saenko, and Trevor Darrell. 2015. Long-term recurrent convolutional networks for visual recognition and description. In *Proceedings of the IEEE International Conference on Computer Vision & Pattern Recognition*. 2625–2634.
[21] Arnaud Doucet, Nando De Freitas, Kevin Murphy, and Stuart Russell. 2000. Rao-Blackwellised particle filtering for dynamic Bayesian networks. In *Proceedings of the Sixteenth Conference on Uncertainty in Artificial Intelligence*. 176–183.
[22] Xuguang Duan, Wenbing Huang, Chuang Gan, Jingdong Wang, Wenwu Zhu, and Junzhou Huang. 2018. Weakly supervised dense event captioning in videos. In *Advances in Neural Information Processing Systems*. 3059–3069.
[23] Vincent Dumoulin and Francesco Visin. 2016. A guide to convolution arithmetic for deep learning.
[24] Nir Friedman, Lise Getoor, Daphne Koller, and Avi Pfeffer. 1999. Learning probabilistic relational models. In *Proceedings of the Sixteenth International Joint Conference on Artificial Intelligence (IJCAI)*. 1300–1309.
[25] Jiyang Gao, Chen Sun, Zhenheng Yang, and Ramakant Nevatia. 2017. TALL: Temporal Activity Localization via Language Query. *2017 IEEE International Conference on Computer Vision (ICCV)* (2017), 5277–5285.
[26] Junyu Gao and Changsheng Xu. 2021. Fast Video Moment Retrieval. In *2021 IEEE/CVF International Conference on Computer Vision (ICCV)*. 1503–1512.
[27] Junyu Gao and Changsheng Xu. 2022. Learning Video Moment Retrieval Without a Single Annotated Video. *IEEE Transactions on Circuits and Systems for Video Technology* 32, 3 (March 2022), 1646–1657.
[28] Kate Goddard, Abdul Roudsari, and Jeremy C Wyatt. 2011. Automation bias: a systematic review of frequency, effect mediators, and mitigators. *Journal of the American Medical Informatics Association* 19, 1 (2011), 121–127.
[29] Shaogang Gong and Tao Xiang. 2003. Recognition of group activities using dynamic probabilistic networks. In *IEEE International Conference on Computer Vision (ICCV)*. 742–749.
[30] David Gunning and David Aha. 2019. DARPA's Explainable Artificial Intelligence (XAI) Program. *AI Magazine* 40, 2 (Jun. 2019), 44–58.
[31] Lisa Anne Hendricks, Oliver Wang, Eli Shechtman, Josef Sivic, Trevor Darrell, and Bryan Russell. 2018. Localizing Moments in Video with Temporal Language. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*. Association for Computational Linguistics, Brussels, Belgium, 1380–1390.
[32] Kevin Anthony Hoff and Masooda Bashir. 2015. Trust in automation: Integrating empirical evidence on factors that influence trust. *Human factors* 57, 3 (2015), 407–434.
[33] Robert R Hoffman. 2017. A Taxonomy of Emergent Trusting in the Human–Machine Relationship. *Cognitive systems engineering: The future for a changing world* (2017), 137–163.
[34] Robert R. Hoffman, Shane T. Mueller, Gary Klein, and Jordan Litman. 2018. Metrics for Explainable AI: Challenges and Prospects. *CoRR* (2018).
[35] Timothy Huang, Daphne Koller, Jitendra Malik, G Ogasawara, Bobby S Rao, Stuart J Russell, and Joseph Weber. 1994. Automatic Symbolic Traffic Scene Analysis Using Belief Networks. In *Proceedings of the Twelfth AAAI Conference on Artificial Intelligence*. 966–972.
[36] Yuri A. Ivanov and Aaron F. Bobick. 2000. Recognition of visual activities and interactions by stochastic parsing. *IEEE Transactions on Pattern Analysis and Machine Intelligence (PAMI)* 22, 8 (2000), 852–872.
[37] Bin Jiang, Xin Huang, Chao Yang, and Junsong Yuan. 2019. Cross-Modal Video Moment Retrieval with Spatial and Language-Temporal Attention. In *Proceedings of the 2019 on International Conference on Multimedia Retrieval (ICMR '19)*. Association for Computing Machinery, New York, NY, USA, 217–225.
[38] Matthew James Johnson, David Duvenaud, Alexander B. Wiltschko, Sandeep R. Datta, and Ryan P. Adams. 2016. *Advances in Neural Information Processing Systems (NeurIPS)* (2016), 2946–2954.
[39] Seong-wook Joo and Rama Chellappa. 2006. Recognition of multi-object events using attribute grammars. In *IEEE International Conference on Image Processing (ICIP)*. 2897–2900.
[40] Maximilian Karl, Maximilian Sölch, Justin Bayer, and Patrick van der Smagt. 2016. Deep Variational Bayes Filters: Unsupervised Learning of State Space Models from Raw Data. *International Conference on Learning Representations* (2016).

[41] Andrej Karpathy, George Toderici, Sanketh Shetty, Thomas Leung, Rahul Sukthankar, and Li Fei-Fei. 2014. Large-Scale Video Classification with Convolutional Neural Networks. In *2014 IEEE Conference on Computer Vision and Pattern Recognition*. 1725–1732.
[42] Yan Ke, Rahul Sukthankar, and Martial Hebert. 2007. Spatio-temporal shape and flow correlation for action recognition. In *2007 IEEE conference on computer vision and pattern recognition*. IEEE, 1–8.
[43] Zafar A Khan and Won Sohn. 2011. Abnormal human activity recognition system based on R-transform and kernel discriminant technique for elderly home care. *IEEE Transactions on Consumer Electronics* (2011), 1843–1850.
[44] Diederik P. Kingma and Jimmy Ba. 2015. Adam: A Method for Stochastic Optimization. In *3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Conference Track Proceedings*, Yoshua Bengio and Yann LeCun (Eds.).
[45] Daphne Koller and Nir Friedman. 2009. *Probabilistic Graphical Models - Principles and Techniques*. MIT Press.
[46] Ranjay Krishna, Kenji Hata, Frederic Ren, Li Fei-Fei, and Juan Carlos Niebles. 2017. Dense-captioning events in videos. In *Proceedings of the IEEE International Conference on Computer Vision*. 706–715.
[47] Rahul G. Krishnan, Uri Shalit, and David Sontag. 2015. Deep Kalman Filters.
[48] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. 2012. Imagenet classification with deep convolutional neural networks. In *Advances in Neural Information Processing Systems*. 1097–1105.
[49] Tai Yu Lai, Jong Yih Kuo, Yong-Yi Fanjiang, Shang-Pin Ma, and Yi Han Liao. 2012. Robust little flame detection on real-time video surveillance system. In *Third International Conference on Innovations in Bio-Inspired Computing and Applications*. 139–143.
[50] Ivan Laptev and Tony Lindenberg. 2003. Space-time Interest Points. *Ninth IEEE International Conference on Computer Vision (ICCV'03)*, 432–439.
[51] Ivan Laptev, Marcin Marszalek, Cordelia Schmid, and Benjamin Rozenfeld. 2008. Learning realistic human actions from movies. In *2008 IEEE Conference on Computer Vision and Pattern Recognition*. IEEE, 1–8.
[52] Colin Lea, Austin Reiter, René Vidal, and Gregory D Hager. 2016. Segmental spatiotemporal cnns for fine-grained action segmentation. In *European Conference on Computer Vision*. Springer, 36–52.
[53] Yann LeCun, Léon Bottou, Yoshua Bengio, and Patrick Haffner. 1998. Gradient-based learning applied to document recognition. *Proceedings of IEEE* 86, 11 (1998), 2278–2324.
[54] John D. Lee and Katrina A. See. 2004. Trust in automation: Designing for appropriate reliance. *Human factors* 46, 1 (2004), 50–80.
[55] Jurij Leskovec, Anand Rajaraman, and Jeffrey D. Ullman. 2014. *Mining of Massive Datasets* (second edition ed.). Cambridge University Press, Cambridge.
[56] Zhenqiang Li, Weimin Wang, Zuoyue Li, Yifei Huang, and Yoichi Sato. 2021. Towards visually explaining video understanding networks with perturbation. In *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision*. 1120–1129.
[57] Junwei Liang, Lu Jiang, Liangliang Cao, Yannis Kalantidis, Li-Jia Li, and Alexander G Hauptmann. 2019. Focal visual-text attention for memex question answering. *IEEE transactions on pattern analysis and machine intelligence* 41, 8 (2019), 1893–1908.
[58] Yitao Liang, Jessa Bekker, and Guy Van den Broeck. 2017. Learning the Structure of Probabilistic Sentential Decision Diagrams. In *Proceedings of the Thirty-Third Conference on Uncertainty in Artificial Intelligence*. 100–109.
[59] Zhijie Lin, Zhou Zhao, Zhu Zhang, Qi Wang, and Huasheng Liu. 2020. Weakly-Supervised Video Moment Retrieval via Semantic Completion Network. *AAAI Conference on Artificial Intelligence* (2020), 11539–11546.
[60] Jun S. Liu and Rong Chen. 1998. Sequential Monte Carlo Methods for Dynamic Systems. *J. Amer. Statist. Assoc.* 93, 443 (1998), 1032–1044.
[61] Meng Liu, Xiang Wang, Liqiang Nie, Xiangnan He, Baoquan Chen, and Tat-Seng Chua. 2018. Attentive Moment Retrieval in Videos. In *The 41st International ACM SIGIR Conference on Research & Development in Information Retrieval* (Ann Arbor, MI, USA) *(SIGIR '18)*. Association for Computing Machinery, New York, NY, USA, 15–24.
[62] Daniel Lowd and Pedro Domingos. 2008. Learning Arithmetic Circuits. In *Proceedings of the Twenty-Fourth Conference on Uncertainty in Artificial Intelligence*. 383–392.
[63] Fengjun Lv and Ramakant Nevatia. 2007. Single view human action recognition using key pose matching and viterbi path searching. In *2007 IEEE Conference on Computer Vision and Pattern Recognition*. IEEE, 1–8.
[64] Robert Mateescu and Rina Dechter. 2005. AND/OR Cutset Conditioning. In *Proceedings of the Nineteenth International Joint Conference on Artificial Intelligence*. Morgan Kaufmann Publishers Inc., San Francisco, CA, USA, 230–235.
[65] Nicola Di Mauro, Antonio Vergari, and Teresa Maria Altomare Basile. 2015. Learning Bayesian Random Cutset Forests. In *Foundations of Intelligent Systems - 22nd International Symposium*. 122–132.
[66] David Miller, Mishel Johns, Brian Mok, Nikhil Gowda, David Sirkin, Key Lee, and Wendy Ju. 2016. Behavioral Measurement of Trust in Automation: The Trust Fall. *Proceedings of the Human Factors and Ergonomics Society Annual Meeting* 60 (09 2016), 1849–1853.
[67] Niluthpol Chowdhury Mithun, Sujoy Paul, and Amit K Roy-Chowdhury. 2019. Weakly supervised video moment retrieval from text queries. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. 11592–11601.
[68] Vlad I. Morariu and Larry S. Davis. 2011. Multi-agent event recognition in structured scenarios. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. 3289–3296.

[69] Bonnie M Muir. 1994. Trust in automation: Part I. Theoretical issues in the study of trust and human intervention in automated systems. *Ergonomics* 37, 11 (1994), 1905–1922.
[70] Bonnie M Muir and Neville Moray. 1996. Trust in automation. Part II. Experimental studies of trust and human intervention in a process control simulation. *Ergonomics* 39, 3 (1996), 429–460.
[71] Kevin P. Murphy. 2002. *Dynamic Bayesian networks: representation, inference and learning*. Ph. D. Dissertation. University of California, Berkeley.
[72] Iftekhar Naim, Young Song, Qiguang Liu, Henry Kautz, Jiebo Luo, and Daniel Gildea. 2014. Unsupervised Alignment of Natural Language Instructions with Video Segments. *Proceedings of the AAAI Conference on Artificial Intelligence* 28, 1 (Jun. 2014), 1558–1564.
[73] Pradeep Natarajan and Ramakant Nevatia. 2007. Coupled hidden semi markov models for activity recognition. In *2007 IEEE Workshop on Motion and Video Computing (WMVC'07)*. IEEE, 10–10.
[74] Joe Yue-Hei Ng, Matthew Hausknecht, Sudheendra Vijayanarasimhan, Oriol Vinyals, Rajat Monga, and George Toderici. 2015. Beyond Short Snippets: Deep Networks for Video Classification. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. 4694–4702.
[75] Bingbing Ni, Xiaokang Yang, and Shenghua Gao. 2016. Progressively parsing interactional objects for fine grained action detection. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. 1020–1028.
[76] Alexandru Niculescu-Mizil and Rich Caruana. 2005. Predicting good probabilities with supervised learning. In *Proceedings of the Twenty-Second International Conference on Machine learning*. 625–632.
[77] Juan Carlos Niebles, Hongcheng Wang, and Li Fei-Fei. 2008. Unsupervised learning of human action categories using spatial-temporal words. *International journal of computer vision* 79 (2008), 299–318.
[78] Nuria Oliver, Barbara Rosario, and Alex Pentland. 2000. A Bayesian computer vision system for modeling human interactions. *IEEE Transactions on Pattern Analysis and Machine Intelligence (PAMI)* 22, 8 (2000), 831–843.
[79] Mayu Otani, Yuta Nakashima, Esa Rahtu, and J. Heikkila. 2020. Uncovering Hidden Challenges in Query-Based Video Moment Retrieval. *British Machine Vision Conference* (2020).
[80] Dong Huk Park, Lisa Anne Hendricks, Zeynep Akata, Anna Rohrbach, Bernt Schiele, Trevor Darrell, and Marcus Rohrbach. 2018. Multimodal explanations: Justifying decisions and pointing to the evidence. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. 8779–8788.
[81] James D Park and Adnan Darwiche. 2004. Complexity results and approximation strategies for MAP explanations. *Journal of Artificial Intelligence Research* 21 (2004), 101–133.
[82] Judea Pearl. 1988. *Probabilistic reasoning in intelligent systems: networks of plausible inference*. Morgan Kaufmann.
[83] Mingtao Pei, Yunde Jia, and Song-Chun Zhu. 2011. Parsing video events with goal inference and intent prediction. In *IEEE International Conference on Computer Vision (ICCV)*. 487–494.
[84] Hoifung Poon and Pedro Domingos. 2011. Sum-Product Networks: A New Deep Architecture.. In *Proceedings of the Twenty-Seventh Conference on Uncertainty in Artificial Intelligence*. AUAI Press, 337–346.
[85] Lawrence R Rabiner. 1989. A tutorial on hidden Markov models and selected applications in speech recognition. *Proc. IEEE* 77, 2 (1989), 257–286.
[86] Tahrima Rahman and Vibhav Gogate. 2016. Learning Ensembles of Cutset Networks.. In *AAAI conference on Artificial Intelligence*, Dale Schuurmans and Michael P. Wellman (Eds.). AAAI Press, 3301–3307.
[87] Tahrima Rahman and Vibhav Gogate. 2016. Merging Strategies for Sum-Product Networks: From Trees to Graphs.. In *Proceedings of the Thirty-Second Conference Conference on Uncertainty in Artificial Intelligence*. 617–626.
[88] Tahrima Rahman, Shasha Jin, and Vibhav Gogate. 2019. Cutset Bayesian networks: a new representation for learning Rao-Blackwellised graphical models. In *Proceedings of the Twenty-Eighth International Joint Conference on Artificial Intelligence*. 5751–5757.
[89] Tahrima Rahman, Shasha Jin, and Vibhav Gogate. 2019. Cutset Bayesian Networks: A New Representation for Learning Rao-Blackwellised Graphical Models. In *Proceedings of the Twenty-Eighth International Joint Conference on Artificial Intelligence*, Sarit Kraus (Ed.). 5751–5757.
[90] Tahrima Rahman, Prasanna Kothalkar, and Vibhav Gogate. 2014. Cutset networks: A simple, tractable, and scalable approach for improving the accuracy of Chow-Liu trees. In *Joint European conference on machine learning and knowledge discovery in databases*. 630–645.
[91] Akshay Rangesh, Eshed Ohn-Bar, Kevan Yuen, and Mohan M Trivedi. 2016. Pedestrians and their phones-detecting phone-based activities of pedestrians for autonomous vehicles. In *Proceedings of the Nineteenth IEEE International Conference on Intelligent Transportation Systems*. 1882–1887.
[92] Cen Rao and Mubarak Shah. 2001. View-Invariance in Action Recognition. In *2001 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR 2001)*. IEEE Computer Society, 316–322.
[93] Michaela Regneri, Marcus Rohrbach, Dominikus Wetzel, Stefan Thater, Bernt Schiele, and Manfred Pinkal. 2013. Grounding action descriptions in videos. *Transactions of the Association for Computational Linguistics* 1 (2013), 25–36.

[94] Marco Tulio Ribeiro, Sameer Singh, and Carlos Guestrin. 2016. " Why should i trust you?" Explaining the predictions of any classifier. In *Proceedings of the 22nd ACM SIGKDD international conference on knowledge discovery and data mining*. 1135–1144.
[95] Mikel D Rodriguez, Javed Ahmed, and Mubarak Shah. 2008. Action mach a spatio-temporal maximum average correlation height filter for action recognition. In *2008 IEEE conference on computer vision and pattern recognition*. IEEE, 1–8.
[96] Anna Rohrbach, Marcus Rohrbach, Wei Qiu, Annemarie Friedrich, Manfred Pinkal, and Bernt Schiele. 2014. Coherent multi-sentence video description with variable level of detail. In *German conference on pattern recognition*. 184–195.
[97] Amirmohammad Rooshenas and Daniel Lowd. 2014. Learning sum-product networks with direct and indirect variable interactions. In *Proceedings of the Thirty-First International Conference on Machine Learning*. 710–718.
[98] Dan Roth. 1996. On the hardness of approximate reasoning. *Artificial Intelligence* 82, 1-2 (1996), 273–302.
[99] Chiradeep Roy, Mahsan Nourani, Donald R. Honeycutt, Jeremy E. Block, Tahrima Rahman, Eric D. Ragan, Nicholas Ruozzi, and Vibhav Gogate. 2021. Explainable activity recognition in videos: Lessons learned. *Applied AI Letters* 2, 4 (2021), e59.
[100] Chiradeep Roy, Tahrima Rahman, Hailiang Dong, Nicholas Ruozzi, and Vibhav Gogate. 2021. Dynamic Cutset Networks. In *Proceedings of The 24th International Conference on Artificial Intelligence and Statistics*. 3106–3114.
[101] Chiradeep Roy, Mahesh Shanbhag, Mahsan Nourani, Tahrima Rahman, Samia Kabir, Vibhav Gogate, Nicholas Ruozzi, and Eric D. Ragan. 2019. Explainable Activity Recognition in Videos. In *The Third Worksop on Tractable Probabilistic Models*.
[102] Cynthia Rudin. 2019. Stop explaining black box machine learning models for high stakes decisions and use interpretable models instead. *Nature Machine Intelligence* 1, 5 (2019), 206–215.
[103] Olga Russakovsky, Jia Deng, Hao Su, Jonathan Krause, Sanjeev Satheesh, Sean Ma, Zhiheng Huang, Andrej Karpathy, Aditya Khosla, Michael Bernstein, et al. 2015. Imagenet large scale visual recognition challenge. *International Journal of Computer Vision* 115, 3 (2015), 211–252.
[104] Michael S Ryoo and Jake K Aggarwal. 2006. Recognition of composite human activities through context-free grammar based representation. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. 1709–1718.
[105] Christian Schuldt, Ivan Laptev, and Barbara Caputo. 2004. Recognizing human actions: a local SVM approach. In *Proceedings of the 17th International Conference on Pattern Recognition, 2004. ICPR 2004.*, Vol. 3. IEEE, 32–36.
[106] Eli Shechtman and Michal Irani. 2005. Space-time behavior based correlation. In *Proceeedings of the IEEE International Conference on Computer Vision & Pattern Recognition*. 405–412.
[107] Yaser Sheikh, Mumtaz Sheikh, and Mubarak Shah. 2005. Exploring the space of a human action. In *Tenth IEEE International Conference on Computer Vision (ICCV'05) Volume 1*, Vol. 1. IEEE, 144–149.
[108] Bharat Singh, Tim K Marks, Michael Jones, Oncel Tuzel, and Ming Shao. 2016. A multi-stream bi-directional recurrent neural network for fine-grained action detection. In *Proceedings of the IEEE conference on computer vision and pattern recognition*. 1961–1970.
[109] Young Chol Song, Iftekhar Naim, Abdullah Al Mamun, Kaustubh Kulkarni, Parag Singla, Jiebo Luo, Daniel Gildea, and Henry A Kautz. 2016. Unsupervised Alignment of Actions in Video with Text Descriptions. In *Proceedings of the Twenty-Fifth International Joint Conference on Artificial Intelligence*. 2025–2031.
[110] Thad Starner and Alex Pentland. 1995. Real-time american sign language recognition from video using hidden markov models. In *Proceedings of the International Symposium on Computer Vision*. 265–270.
[111] Christian Szegedy, Wei Liu, Yangqing Jia, Pierre Sermanet, Scott Reed, Dragomir Anguelov, Dumitru Erhan, Vincent Vanhoucke, and Andrew Rabinovich. 2015. Going deeper with convolutions. In *Proceedings of the IEEE International Conference on Computer Vision & Pattern Recognition*. 1–9.
[112] Du Tran, Lubomir Bourdev, Rob Fergus, Lorenzo Torresani, and Manohar Paluri. 2015. Learning Spatiotemporal Features With 3D Convolutional Networks. In *The IEEE International Conference on Computer Vision (ICCV)*. 4489–4497.
[113] Rajesh Kumar Tripathi, Anand Singh Jalal, and Subhash Chand Agrawal. 2018. Suspicious human activity recognition: a review. *Artificial Intelligence Review* 50, 2 (2018), 283–339.
[114] Kush R. Varshney and Homa Alemzadeh. 2017. On the Safety of Machine Learning: Cyber-Physical Systems, Decision Sciences, and Data Products. *Big Data* 5, 3 (2017), 246–255.
[115] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Ł ukasz Kaiser, and Illia Polosukhin. 2017. Attention is All you Need. In *Advances in Neural Information Processing Systems*, I. Guyon, U. Von Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett (Eds.), Vol. 30. Curran Associates, Inc.
[116] Michael Veale, Max Van Kleek, and Reuben Binns. 2018. Fairness and Accountability Design Needs for Algorithmic Support in High-Stakes Public Sector Decision-Making. In *Proceedings of the 2018 CHI Conference on Human Factors in Computing Systems* (Montreal QC, Canada) *(CHI '18)*. Association for Computing Machinery, New York, NY, USA, 1–14.
[117] Rebecca Wexler. June 13, 2017. When a Computer Program Keeps You in Jail. New York Times.
[118] Shu-Fai Wong, Tae-Kyun Kim, and Roberto Cipolla. 2007. Learning motion categories using both semantic and structural information. In *2007 IEEE Conference on Computer Vision and Pattern Recognition*. IEEE, 1–6.
[119] Ziyue Wu, Junyu Gao, Shucheng Huang, and Changsheng Xu. 2021. Diving Into The Relations: Leveraging Semantic and Visual Structures For Video Moment Retrieval. In *2021 IEEE International Conference on Multimedia and Expo (ICME)*. 1–6. ISSN: 1945-788X.

[120] Zuxuan Wu, Xi Wang, Yu-Gang Jiang, Hao Ye, and Xiangyang Xue. 2015. Modeling spatial-temporal clues in a hybrid deep learning framework for video classification. In *Proceedings of the 23rd ACM international conference on Multimedia*. 461–470.

[121] Junji Yamato, Jun Ohya, and Kenichiro Ishii. 1992. Recognizing human action in time-sequential images using hidden markov model. In *Proceedings of the IEEE International Conference on Computer Vision and Pattern Recognition*. 379–385.

[122] Alper Yilmaz and Mubarak Shah. 2005. Actions sketch: A novel action representation. In *2005 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'05)*, Vol. 1. IEEE, 984–989.

[123] Joe Yue-Hei Ng, Matthew Hausknecht, Sudheendra Vijayanarasimhan, Oriol Vinyals, Rajat Monga, and George Toderici. 2015. Beyond short snippets: Deep networks for video classification. In *Proceedings of the IEEE conference on computer vision and pattern recognition*. 4694–4702.

[124] Lihi Zelnik-Manor and Michal Irani. 2001. Event-based analysis of video. In *Proceedings of the 2001 IEEE Computer Society Conference on Computer Vision and Pattern Recognition. CVPR 2001*, Vol. 2. IEEE, II–II.

[125] Zhang Zhang, Tieniu Tan, and Kaiqi Huang. 2011. An Extended Grammar System for Learning and Recognizing Complex Visual Events. *IEEE Transactions on Pattern Analysis and Machine Intelligence (PAMI)* 33, 2 (2011), 240–255.

[126] Tao Zhuo, Zhiyong Cheng, Peng Zhang, Yongkang Wong, and Mohan Kankanhalli. 2019. Explainable video action reasoning via prior knowledge and state transitions. In *Proceedings of the 27th acm international conference on multimedia*. 521–529.