Spring 2022

High Level Learning Using the Temporal Features of Human Demonstrated Sequential Tasks

Madison Brandywine Clark-Turner
University of New Hampshire, Durham

HIGH LEVEL LEARNING USING THE TEMPORAL FEATURES OF HUMAN DEMONSTRATED SEQUENTIAL TASKS

BY

MADISON CLARK-TURNER

BSc in Computer Science and Biology, Franklin and Marshall College, 2014

DISSERTATION

Submitted to the University of New Hampshire

in Partial Fulfillment of

the Requirements for the Degree of

Doctor of Philosophy

in

Computer Science

December, 2021

This dissertation has been examined and approved in partial fulfillment of the requirements for the degree of Doctor of Philosophy in Computer Science by:

Dissertation Director, Momotaz Begum
Assistant Professor of Computer Science,
University of New Hampshire

Laura Dietz
Assistant Professor of Computer Science,
University of New Hampshire

Marek Petrik
Assistant Professor of Computer Science,
University of New Hampshire

Wheeler Ruml
Professor of Computer Science,
University of New Hampshire

Odest Chadwicke Jenkins
Professor of Computer Science and Engineering,
University of Michigan

On August 27th, 2021

Approval signatures are on file with the University of New Hampshire Graduate School.

DEDICATION

To my wonderful wife, Katy, who has stuck with me through all the sleepless nights.

And to Fay, my beautiful daughter, who has been causing them.

ACKNOWLEDGEMENTS

First, I extend my heartfelt gratitude to my mentor, Prof. Momotaz Begum, who has not only provided informed guidance with the direction and writing of my dissertation, but has been an enthusiastic advocate of me and my work from day one. Her dedication to ensuring the highest standards in both the quality of the lab and my work have been inspirational. At times this journey has been a challenge but your consistent help and support, both academic and social, have helped to see it through. Thank you.

It is impossible to talk about Momotaz without also extending thanks to Mostafa Hussein, Paul Gesel, and the rest of the Cognitive Assistive Robotics Lab. Though our group has been small, I have had the luxury to watch it evolve and develop into a thriving, collaborative community. We've celebrated in each others victories and wallowed in each others losses. I truly hope that the friendships made here are lifelong and I look forward to hearing about the future successes that will come out of our lab and your work.

I sincerely thank Profs. Laura Dietz, Marek Petrik, Wheeler Ruml, and Chad Jenkins all of who volunteered their time to be members of my dissertation committee. I am especially grateful for your feedback which has helped to refine not only the content of my dissertation but also my skills as both an orator and a scientist. I also extend this thanks to the greater Computer Science department at UNH whose vested interest in me extended beyond my work to both my mental and physical well being. I have made many friendships with professors, students, and staff alike and I appreciate all your help in my times of need. I must also acknowledge the National Science Foundation which has funded part of this work (IIS-1664554).

Finally, anything I put into writing could only ever be an understatement to the immense role that my family has played in supporting me throughout this PhD. My parents, Jeremy and Naomi, have dedicated time and resources and have consistently found new ways to help me through all the writing and tough times. To my wife Katy, you have been by my side through it all. Every success and rejection, every late night and last minute revision. Your patience and love are without fault. Thank you for helping me get here, and thank you for continuing to be a part of my life.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

LIST OF ABBREVIATIONS

ASD	Autism Spectrum Disorder.
BC	Behavioral Cloning.
CNN	Convolutional Neural Network.
D-ITR-L	Deep Interval Temporal Relationship Learner.
DBN	Dynamic Bayesian Network.
DQN	Deep Q-Network.
FSA	Finite State Automata.
GCN	Graph Convolutional Network.
HMM	Hidden Markov Model.
IAD	Interval Algebraic Descriptor.
ITBN	Interval Temporal Bayesian Network.
ITR	Interval Temporal Relationship.
LfD	Learning from Demonstrations.

LSTM	Long Short-Term Memory.
PTN	Probabilistic Temporal Network.
R-GCN	Relational-Graph Convolutional Network.
RNN	Recurrent Neural Network.
TCN	Temporal Convolutional Network.

GLOSSARY

Behavioral Cloning A policy learning approach where a policy is learned as a mapping of states to actions.

Deep Interval Temporal Relationship Learner My novel contribution. A temporal wrapper that identifies the temporal features present in a video using the pre-trained spatial features in an underlying CNN architecture.

Deep Q-Network A popular deep reinforcement learning architecture. The original network structure is based on a typical CNN. I modified a DQN as part of my exploratory research allowing it to learn a social greeting behavioral intervention.

Durational Variance Variance in the number of frames, seconds, minutes, or other measurement describing a period of time over which a given feature or action is expressed in a video..

Dynamic Bayesian Network A Bayesian Network that connects concepts across time steps.

Finite State Automata A graphical model that represents the world as one of a finite set of states. Transitions can be made to other states along edges as the result of a given stimulus.

Graph Convolutional Network A Convolutional Neural Network structure capable of extracting discriminatory features from graphical structures.

Hidden Markov Model A statistical model for representing a hidden state distribution through observations expressed over time.

Interval Algebraic Descriptor An intermediary structure in D-ITR-L. The IAD is a two dimensional matrix that denotes the relative expression of a set of features over time in a video input.

Interval Temporal Bayesian Network A probabilistic temporal model that leverages interval algebra based representations. The description of temporal relationships in this model are stochastic.

Interval Temporal Relationship A named relationship between two events that describes the degree and order of their overlap. Interval Temporal Relationships compromise the basis for the D-ITR-L temporal wrapper.

Long Short-Term Memory A common pattern learning structural unit found in deep learning architectures.

Probabilistic Temporal Network A probabilistic temporal model that leverages interval algebra based representations in a FSA. The description of temporal relationships in this model are deterministic and bi-directional.

Recurrent Neural Network A neural network design that captures the presence of patterns in temporal or time series data.

Relational-Graph Convolutional Network A Graph Convolutional Network extension that allows for more judicious merging of labels using discrete edge labels.

Temporal Convolutional Network A convolution-based temporal reasoning model that performs 1D convolutions across the duration of a video.

Temporal Features A discriminatory property of a video defined by the expression or co-expression of spatial features in time. In this thesis a temporal feature explicitly captures an Interval Temporal Relationship as it exists between two spatial features..

Video-Scale Features A temporal feature that can span the full duration of a video. This work investigates two categories of video-scale features: *long-term dependencies* and *cyclical activities*..

ABSTRACT

HIGH LEVEL LEARNING USING THE TEMPORAL FEATURES OF HUMAN DEMONSTRATED SEQUENTIAL TASKS

by

Madison Clark-Turner

University of New Hampshire, December, 2021

Modelling human-led demonstrations of high-level sequential tasks is fundamental to a number of practical inference applications including vision-based policy learning and activity recognition. Demonstrations of these tasks are captured as videos with long durations and similar spatial contents. Learning from this data is challenging since inference cannot be conducted solely on spatial feature presence and must instead consider how spatial features play out across time. To be successful these temporal representations must generalize to variations in the duration of activities and be able to capture relationships between events expressed across the scale of an entire video.

Contemporary deep learning architectures that represent time (convolution-based and Recurrent Neural Networks) do not address these concerns. Representations learned by these models describe temporal features in terms of fixed durations such as minutes, seconds, and frames. They are also developed sequentially and must use unreasonably large models to capture temporal features expressed at scale. Probabilistic temporal models have been successful in representing the temporal information of videos in a duration invariant

manner that is robust to scale, however, this has only been accomplished through the use of user-defined spatial features. Such abstractions make unrealistic assumptions about the content being expressed in these videos, the quality of the perception model, and they also limit the potential applications of trained models. To that end, I present Deep Interval Temporal Relationship Learner (D-ITR-L), a temporal wrapper that extends the spatial features extracted from a typically CNN architecture and transforms them into temporal features.

D-ITR-L-derived temporal features are duration invariant and can identify temporal relationships between events at the scale of a full video. Validation of this claim is conducted through various vision-based policy learning and action recognition settings. Additionally, these studies show that challenging visual domains such as human-led demonstration of high-level sequential tasks can be effectively represented when using a D-ITR-L-based model.

CHAPTER 1
INTRODUCTION

Temporal features play a critical role in understanding and interpreting the visual world. However, capturing and learning from temporal features in a data-driven manner from visual data (video) is a complex task. This dissertation presents Deep Interval Temporal Relationship Learner (D-ITR-L), a wrapper that autonomously crafts descriptive temporal features from the learned spatial features of a backbone convolutional neural network (CNN). I demonstrate how leveraging the D-ITR-L derived representation of temporal features leads to improved performance in both policy learning and activity recognition using video data.

1.1 Motivation

Temporal features describe the abstract relationships that occur between visual spatial features as they are expressed over time. They are particularly important when making inferences about observations collected from visually similar environments where the presence of spatial features alone is insufficient to make distinctions between different observations. For example, consider the task of making tea. Concepts such as order (the teabag was added *before* the water), temporal overlap (the milk was poured *while* the tea was being stirred), and cyclical patterns (sugar was added to the tea *twice*) can only be understood by analyzing their expression in time. The critical role that temporal features play in making data-driven inference is best understood in the context of multi-step sequential tasks including most activities of daily living, furniture assembly, and structured social interactions. Here, the ubiquitous presence of common spatial features across many observations

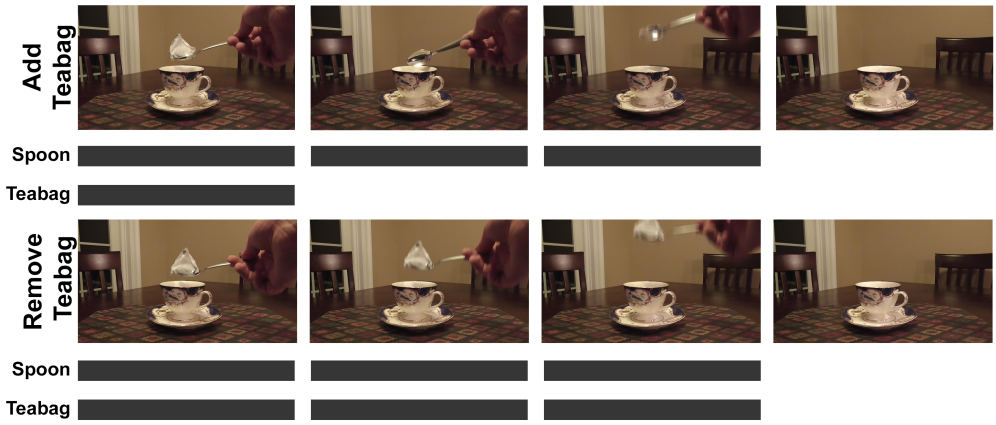


Figure 1.1: Activity Recognition Requires Temporal Understanding

(e.g. in the background) makes it difficult to differentiate specific activities within a task. For instance, a teacup, spoon, and teabag are all likely to be visible during different stages of making tea. However, using only that information, it is difficult to distinguish between adding a teabag to a cup versus removing it. Instead, by using temporal features a more informed representation of the visual data can be crafted that is both discriminatory and usable in more complex inference tasks. For example, in Fig. 1.1, the overlap between the teabag and the spoon can be used to distinguish between the actions of "Add Teabag" and "Remove Teabag". When adding the teabag, the spoon and teabag are initially visible at the same time until the teabag is dropped into the cup whereupon it becomes obscured and is no longer visible. When removing the teabag, the spoon and teabag are both visible from the beginning of the video until they have exited the scene. Identification of and inference from temporal features is the contribution presented in this dissertation.

Historically, identification of temporal features has been accomplished with probabilistic temporal models [4, 5]. These models perform their inference using the temporal features generated from user-defined spatial feature extractors. In contrast, CNNs capture discriminatory features in an autonomous and data-driven fashion. Recent advances in CNNs have extended their feature learning capabilities from images to videos, triggering extensive research interest in activity recognition from raw video data [6]. Although temporal features are a hallmark of human activities, current video-recognition architectures do not leverage

them when making inferences. Despite this, existing video recognition architectures generate impressive results, primarily because of the spatial nature of the majority of standard benchmark video datasets upon which they are validated. Cao *et al.* [7] specifically address the spatial focus of contemporary models and their evaluation criteria. They argue that current benchmark video datasets can be classified primarily by their spatial content and they do not challenge video-recognition architectures when it comes to representing temporal information. I echo this sentiment and assert that the properties of these datasets have encouraged state-of-the-art video-recognition architectures to adopt designs that are ill-suited for long-form and visually similar video settings. Understanding typical human activities (e.g. multi-step, sequential tasks) in natural settings demands an even greater degree of temporal representation than what is currently required of the most temporally-focused benchmark video datasets presented in the computer vision literature. Learning robust temporal features while leveraging the CNN's ability to learn spatial features is an unexplored research domain and is the focus of my work.

1.2 Challenges of Learning Temporal Features in Convolutional Architectures

Two critical challenges impede temporal feature learning within CNN architectures: the absence of duration invariant feature representation and the use of *ad hoc* solutions to capture video-scale features.

1.2.1 Duration Invariance

Video observations, especially those that focus on human participants, are prone to variations in the temporal dimension. When and how long a feature is expressed is rarely consistent. Fig. 1.2 captures an example of this phenomenon. The act of waving is one activity that can vary dramatically in its length (and start time) depending on the person performing the task (frame numbers are depicted in the top left of each image). The act of waving is not defined by its duration; (in some cases a single flick of the wrist might be sufficient). Consequently

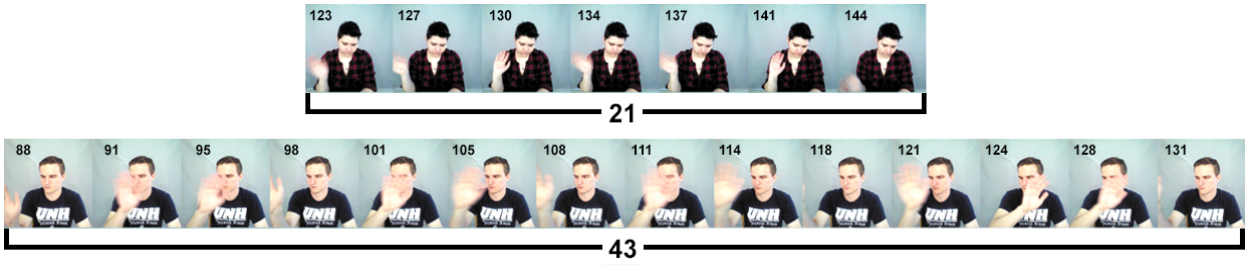


Figure 1.2: Variance in the Duration of a "Wave" Action.

we could anticipate that a single representation would suffice for both examples. Such a representation would be identified as duration invariant, and would be ideal as it generalizes better to the data regardless of the time that a person spends waving.

Regrettably, convolutional architectures represent temporal features in a duration dependent manner. Features are defined in concrete expressions such as the number of frames, seconds, and/or minutes over which they are expressed. To capture temporal variations of the same activity, these models must generate multiple latent representations of the activity, one for each duration. Recurrent Neural Network (RNN) architectures possess greater potential to capture temporal features in a duration invariant manner. RNNs aggregate features in a sequential frame-by-frame manner and maintain a representation that does not use duration by merging periods of similar expression together [8, 9]. In practice, the presence of noisy feature expression in real world video data encourages the development of low-level feature patterns. These patterns are defined by explicit durations and are therefore limited in the same manner as convolutional approaches. Duration dependent concerns are an extension of the well known scale invariant limitations of traditional 2D-CNNs extended to the third dimension [10]. Finally, the default solution when CNN models falter, to accrue additional samples, is naive and infeasible for many researchers whose ability to create new data is limited by physical constraints and human fatigue.

Describing temporal features in a duration invariant fashion requires the use of abstract relationships. This can be accomplished by placing greater focus on the moments when a spatial feature starts and stops being expressed and giving less consideration for the period

of expression between these points. Assuming the set of spatial features that capture the start of a wave as "X" and the set of features that conclude the wave as "Y", you can use the abstract terminology of X *before* Y to represent both of the video observations depicted in Figure 1.2.

1.2.2 Video-Scale Features

Temporal features can be expressed over the course of an entire video. For example, making tea occurs in several stages: boiling water, adding the teabag to the cup, and pouring water into the cup. To identify that tea was made correctly, all three sub-tasks must be present and specific temporal constraints, such as ordering, cannot be violated (e.g. water must be boiled *before* it is poured). Similarly, cyclical activities, such as adding spoonfuls of sugar to a teacup, can be expressed over an extended duration. An explicit number of repetitions can sometimes convey meaning and should be clearly represented when developing temporal features from video. Fig. 1.3 depicts simplified examples of these two categories of video-scale features. These examples show frames of videos in which colored blocks (blue, green, and red) are moved from one opaque container to another. Frames proceed left to right and are colored when blocks are visible. Long-term dependencies such as order are represented by movement of the blue and green blocks. Cyclical activities are represented by the movement of one, two, and three red blocks. Distinguishing between the two examples of long-term dependencies and the three cyclical activities requires a robust way to model the relationships between these spatial features given that these videos last upwards of 200 frames.

Capturing video-scale temporal features is challenging for contemporary video recognition architectures. Deep learning-based approaches are built upon hierarchical design: constructing larger and more complex representations from smaller, low-level representations. The same tenant has been adopted when modelling temporal data, which is problematic for videos of long duration. Capturing distant relationships in the data requires a network that is deep enough to span the distance between the composite aspects of a video-scale temporal feature.

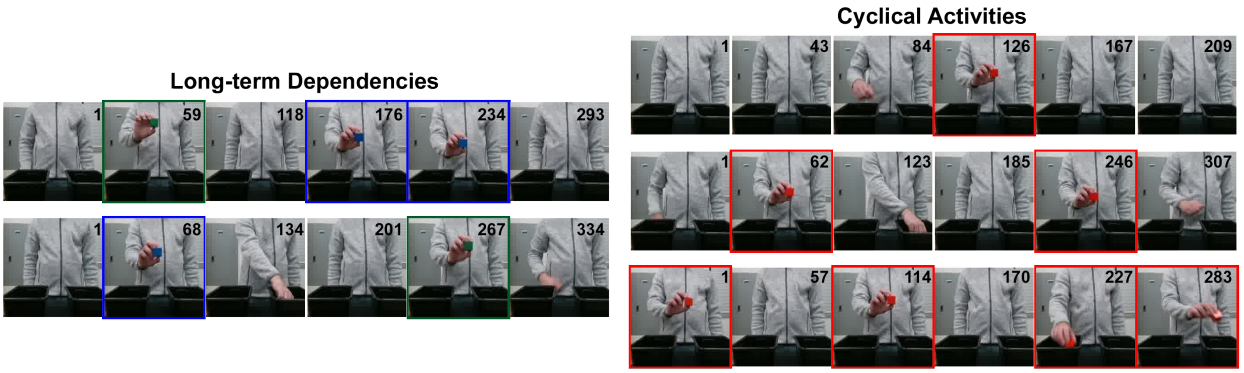


Figure 1.3: Simplified Examples of Video-Scale Features.

Convolution-based networks are notorious for their computational demands. In order to manage the structural requirements of a long-form video, these architectures have had to make accommodations. Depending on the implementation, these models sacrifice either fidelity or robustness to work. Models that sacrifice fidelity employ frame-skipping approaches that linearly sample frames from throughout a video to avoid performing computations at every time step [9, 11]. However, in order for these approaches to select appropriate frame strides, they use computationally expensive (and redundant) ensembles of models. Approaches that sacrifice robustness perform frame-level inference, but are instead more vulnerable to variances in the duration of activities in the video input [12, 13]. The natural inclination to segment the video input into manageable chunks is also ill-conceived as it requires either expert knowledge or heavy annotation of the dataset [14].

Capturing video-scale temporal features is best accomplished using a graphical implementation. A graph can directly relate the features in the video regardless of their temporal location, allowing for easy recognition of distant connections regardless of scale.

1.3 Proposed Solution

CNN architectures are capable of autonomously learning spatial features, but lack the ability to capture and infer from temporal features. Probabilistic approaches make their inference using temporal features, but are designed to operate only on hand-crafted spatial features. I

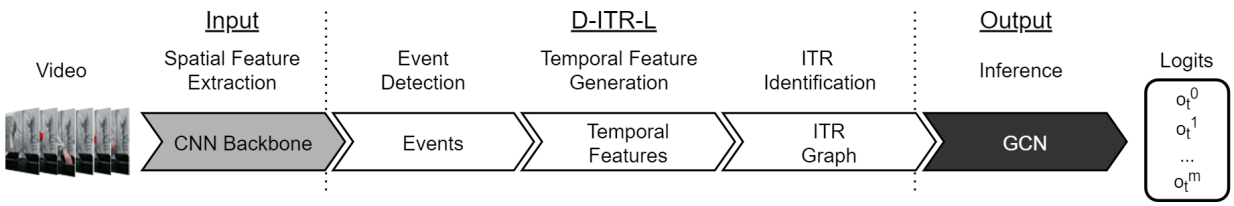


Figure 1.4: The D-ITR-L Pipeline.

merge these two disparate approaches in Deep Interval Temporal Relationship Learner (D-ITR-L), a wrapper that uses the spatial features learned by CNN architectures in concert with the descriptive and powerful representations common among probabilistic temporal modelling approaches.

D-ITR-L operates as a pipeline (Fig. 1.4) transforming spatial features, identified by a CNN backbone, into a graph of abstract relationships that are used as the basis for further inference. This process happens in three stages. First, spatial events (discrete durations where spatial features are expressed) are detected within the output of a traditional CNN backbone. This is accomplished by the generation and subsequent thresholding of an Interval Algebraic Descriptor, a custom representation of spatial features across time. After the spatial events have been detected they are used to generate an exhaustive list of abstract temporal relationships through a pairwise combination of events. Finally, the corpus of temporal features is organized into a graphical structure termed an Interval Temporal Relationship graph. This final representation is duration invariant, a result of using abstract-terms to define temporal relationships, and is capable of representing video-scale features, due to the properties of a graphical structure at inference time. Inference using an ITR graph is accomplished through the use of a Graph Convolutional Network (GCN).

1.4 Contribution

The critical contribution of this work is D-ITR-L a temporal wrapper fashioned as a pipeline that extracts temporal relationships present in a video observation. These temporal relationships are formatted in an ITR graph, a novel structure that conveys the expression of

temporal features across the duration of a full video and in a duration invariant manner. Furthermore, this representation can be used as a substitute for pixel-based deep learning inference tasks. Through the ITR graph's combined use of abstract relationships and graphical format, D-ITR-L can describe features that span the duration of a video. This is accomplished without the use of concrete terms describing the duration (i.e. the number of frames, seconds, or minutes) that a feature is expressed.

D-ITR-L-based inference results in more effective state estimation in applications that use long-duration videos of visually-similar environments, such as human demonstrated sequential tasks. This works is of particular interest to Learning from Demonstration research where activities of daily living are concerned such as service robot applications. This claim is defended through several experiments (Chapters 5, 6, and 7). Conversely, it is expected that D-ITR-L will perform poorly in applications that require representations that capture duration or those that are well defined by a large selection of spatial features. Video datasets that capture these properties are not investigated in this dissertation.

1.5 Chapter Summary

This dissertation is separated into the following chapters:

Chapter 2: Exploratory Work: An End-To-End Model For High-level Task Learning From Video Demonstrations

I describe my exploratory work in which I developed an end-to-end deep learning architecture designed to learn how to perform a high-level sequential task policy from human-led demonstrations. My implementation used a Deep Q-Network and leveraged a RNN in the form of a Long Short-Term Memory cell to model the temporal dynamics of an audio-visual input. This exploratory work not only generated publications in both RO-MAN 2017 and HRI 2018 but also identified two challenges in vision-based task learning from demonstrations: 1) that perceptual aliasing is a critical problem that can jeopardize task learning and

2) learning temporal information from vision data is a precondition for conducting temporal reasoning and there is no effective data-driven tool that can accomplish this. I collaborated with a fellow Master's student on his attempt to solve the first challenge which produced a co-authored papers in RO-MAN 2019 and ICRA 2019. The research I performed to address the second challenge has become my dissertation.

Chapter 3: Literature Review

As context for my work I define Interval Algebra and Graph Convolutional Networks. I also provide a literature review detailing the current research as it pertains to identifying and learning from temporal features collected from raw video data. I describe the various limitations of existing architectures and how contemporary and classical models fail to provide a holistic model that can learn video-scale features, in a duration invariant manner.

Chapter 4: Deep Interval Temporal Relationship Learner

The chapter presents Deep Interval Temporal Relationship Learner (D-ITR-L), the proposed method for learning video-scale features in a duration invariant way.

Chapter 5: Temporal Features for Policy Learning

D-ITR-L presents temporal features in a manner designed to specifically address concerns relating to *duration invariance* and *video-scale features*. I validate this claim in the context of state estimation for a policy learning model. Included in this chapter is a description of the steps taken to establish D-ITR-L as a state estimator and the steps taken to capture and investigate the challenges of temporal features.

Chapter 6: Temporal Features for Human Activity Recognition

Contemporary deep-learning activity-recognition architectures perform well as a result of the spatially-focused benchmark video datasets on which they are evaluated. The challenges of learning temporal features (*duration invariance* and *video-scale features*) are poorly represented in such data. Instead, these obstacles are most evident in video domains where humans perform activities in natural settings. I illustrate how examples from these domains challenge contemporary models and, as a result of the specific design choices employed in D-ITR-L, how these challenges can be overcome by robust temporal representation.

Chapter 7: Temporally-Guided Feature Selection

Temporal representations are necessary to effectively represent video data. The latent temporal representations used by D-ITR-L are dependent on clear recognition of a set of informative spatial features. I conduct a rigorous quantitative analysis to measure the contribution that a temporal information-guided approach can effect in spatial feature selection. Additionally, I investigate how the properties of human understandable features selected in this way can contribute to temporal feature learning in D-ITR-L.

Chapter 8: Conclusion

The dissertation concludes with a summary of my completed work and an assessment of future directions.

CHAPTER 2

EXPLORATORY WORK: AN END-TO-END MODEL FOR HIGH-LEVEL TASK LEARNING FROM VIDEO DEMONSTRATIONS

I started my PhD research by investigating the feasibility of learning high-level, multi-step, human-robot interaction tasks from visual observations where demonstration videos can be long in duration, e.g. several hundred seconds. The specific task that I focused on learning was a structured educational intervention consisting of multiple steps where activities at a given step depend on the activities executed in the distant past. Learning such a task therefore requires understanding temporal information from visual data. I developed a general end-to-end model, consisting of a Deep Q Network that leveraged LSTM, for learning high-level multi-step tasks from user-segmented video demonstrations [15, 16]. This preliminary work unearthed two challenges in vision-based task learning from visual demonstrations: 1) that perceptual aliasing is a critical problem that can jeopardize task learning and 2) learning temporal information from vision data is a precondition for conducting temporal reasoning and there is no effective data-driven tool that can accomplish this. Investigations into the former would branch into research on temporal reasoning and hidden state information. Meanwhile, investigations into the later laid the foundation for my current dissertation. This chapter describes the generic end-to-end model and its evaluation with an educational intervention task.

2.1 Introduction

I designed a framework to teach a robot how to learn a high-level sequential task policy after observing tele-operated demonstrations. This work is prototypical and, to establish its viability, it is evaluated within the confines of a single application: a social greeting behavioral intervention designed as a therapy for individuals with Autism Spectrum Disorder (ASD). The responses of this population, when exposed to the same stimuli, can vary dramatically among individuals and it is naive to assume that hand-picked features would suffice to capture all possible observations. A more reasonable approach is to use a data-derived set of features to identify the specific responses of particular individuals. This motivation, along with the recent successes of deep learning, not only as an image processing architecture but also as a tool for policy learning, encouraged an end-to-end model for learning this high-level task from visual observations. Specifically, my model adheres to the structure of a state-of-the-art deep reinforcement learning architecture: the Deep Q-Network (DQN).

2.2 Related Work

In contrast to other high-level learning from demonstration models, my adoption of deep learning is novel. Prior works made simplifying assumptions regarding their perception including leveraging a set of hand-picked visual features that captured the location of a set of discrete utensils in a table setting application [17], using object detectors to recognize specific objects in a pick-and-place task [18], or absolving perception altogether by associating robot actions with ground truths when sorting objects by shape and color [19]. Subsequently, there was little precedent for structuring a policy learner that could draw inference from long video observations and make policy decisions in a partially observable domain; learning high-level sequential task policies requires both.

The most similar research direction when considering the representation of long video observations was human activity recognition. However, learning of abstract reasoning from

data (i.e. high-level LfD) is subtly different than video labeling through deep learning. Video labeling is specific to a video dataset and allows for the discrimination of observations based on the many (potentially unique) properties of an observation. In contrast, high-level LfD hopes to re-use learned components to explain novel perceptions and is interested in capturing the objects and rules of an observation (effectively leading to the same explanation through a different direction). That being said, the two most popular methods among these works were two-stream models [2] and signal inference based methods [20]. Two-stream models combined RGB data with alternative information that capture movement across frames such as optical flow. Signal inference methods implemented common time sequence tools, most frequently Long Short-Term Memory (LSTM). I incorporate both of these design elements into my model.

Representation of partially observable information in the state estimation of an end-to-end deep reinforcement learning model was still an unexplored research domain when this research was being conducted. In this work I adopted a simple strategy (use of a discrete value denoting an action history) to demonstrate the importance of this information and the need for a more complex representation.

2.3 Background: Deep Q-Network

The DQN is a popular CNN-based model for developing policies in deep reinforcement learning environments [21]. The model, rather than performing a traditional classification operation, is tasked with generating action value estimates (q-values) for a given state representation (image/video). A DQN differs from a traditional CNN in three regards. 1) DQNs generate q-values instead of predictions over given actions by ensuring that the network terminates with a regression layer rather than a softmax operator. 2) To avoid divergent learning, a common issue when training neural networks using sequential observations collected from the same source, Minh *et al.* [21] used a buffer of training examples. Training examples were randomly sampled from this buffer to vary the order of the data that the

model is exposed to. 3) DQN training is performed using two separate networks to teach stable q-values. One network that is updated frequently is used to define the policy. The other, updated infrequently, is used to generate state reward estimates.

My model makes several edits to the original DQN structure. Notably I train the model in a supervised approach using a set of known trajectories. I forgo the sample buffer (#2 above) that traditional DQNs possess and instead sample state-action pairs randomly from my observed demonstrations. Work by Hester *et al.* [22] also broached the topic of deep q-learning from demonstration but their approach is applied only to learning how to play Atari video games, as opposed to real-world applications.

2.4 An End-to-End Model for High-Level Task Learning

The visual input from a high-level skill learning task involving human participants differs from the typical input a CNN architecture would expect in three regards. First, it is multi-modal (containing both verbal and non-verbal attributes) either of which could be significant to the task being learned. Secondly, demonstrations are likely to be performed over an extended duration as human responses and actions can take several seconds to be fully delivered. Finally, responses could be composed of separate parts (i.e. looking at the robot and gesturing). These aspects could be fleeting and to avoid missing them requires evaluating the entire video. With these considerations in mind I adapt the DQN to better accommodate the challenges of the data. The modified architecture is depicted in Fig. 2.1 and discussed in the following sections. Model parameters are included in the provided figure with F defining the kernel size, S capturing the stride, N the number of features, and O the output dimensions where relevant. The code for this implementation is available here: [23]

2.4.1 Feature Extraction

Each demonstration can be separated into a sequence of actions (a_t) and observations (where $o_t = (V_t, p_t)$) that last for $t \in T$ timesteps. In this interaction o_t is a composition of an

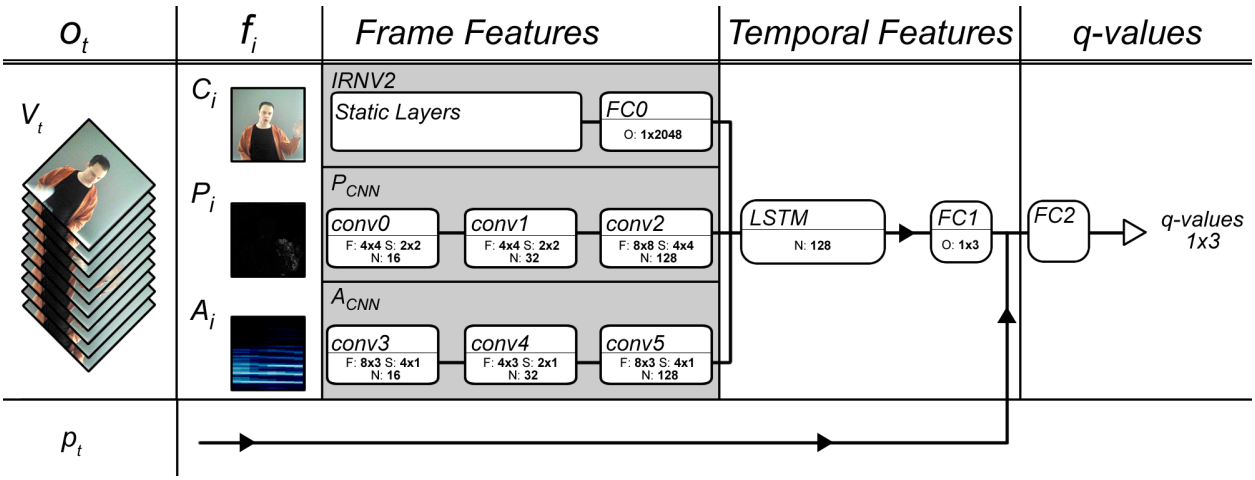


Figure 2.1: Modified DQN for High-Level LfD

audio-video sequence (V_t; composed of $i \in I$ frames) collected through the robot's egocentric camera and the robot's microphone, and hidden state information encoded in p_t. I modelled my approach on existing deep human activity recognition design principles that had performed well on benchmark video datasets. I emulate a 2-stream approach [2] by concatenating the latent representation that the convolutional layers learned from the raw RGB data (C_i) with the latent features that parallel convolutional layers extract from the video's optical flow (P_i, the movement that occurred between adjacent frames) [24]. Theorizing that the co-expression of verbal and non-verbal information is likely to illicit an even more accurate response from the model, I extend this concept to include information present in the audio input as well. To that end I transform the audio signal into a Mel-Spectrogram to provide a 2D spectral representation of the data [25] and proceed to separate it into packets to coincide with the number of frames in the input video (A_i). Mel-Spectorgrams better represent frequencies in the range of human voice than traditional spectral models and are therefore a natural consideration given the strong human presence in demonstrations of sequential tasks. The data (RGB, optical flow, and audio frames) are passed into three convolutional stacks to identify relevant discriminatory features present in the observation. This work employs the concept of transfer learning by leveraging features in a trained deep learning architecture for a novel application [26]. This is accomplished through the use of

(InceptionResNetV2/IRNV2) an image inference architecture that has performed well on a popular image classification benchmark: ImageNet [27]. The RGB frames are passed through IRNV2. No such pre-trained network existed for the optical flow and audio channels at the time that this work was completed so I pass these data-types through CNNs of my own design (P_{CNN} and A_{CNN} respectively). These architectures were composed of 3 convolutional layers that each terminate in a 2048 length feature vector. This output matches that of the IRNV2 architecture giving equal influence to each of the different modal inputs.

2.4.2 Temporal Feature Modelling

The extended duration of these interactions made capturing temporal information in the videos of critical importance. Once the latent spatial features have been extracted from all three data channels they are concatenated and fed sequentially into a Long Short-Term Memory (LSTM) layer. LSTM cells are a popular tool among natural language processing and signal inference architectures and their usefulness in video inference is still being explored in the context of activity recognition [20]. A LSTM cell can capture patterns in the data such as the waving of a hand or auditory patterns. The result of this inference is passed into a fully connected/inference layer ($FC1$) to develop preliminary q-values: estimates of the action values based solely on the most recent observation (o_t) of the task.

In contrast to the simulated reinforcement learning tasks learned by the DQN models (i.e. Atari games [21] and Go [28]), it is rarely the case that the entire world state can be observed directly from observations of high-level tasks. This is a challenge referred to as *perceptual aliasing* and describes how visually similar observations can carry different meaning in context. My system addresses this concern by incorporating an action history into my model's inference. I include an integer (p_t) that increments each time a specific action is called (specifically the prompting action, **PMT**, as defined in Section 2.5.1). This information is concatenated with the preliminary q-values generated from the observation before they are fed through a final linear layer ($FC2$) to generate the final q-value prediction.

The action with the highest q-value is selected as the action to execute in the next step of the policy (a_{t+1}).

2.5 Experiments

The effectiveness of my approach towards learning a high-level task from demonstrations is evaluated in the context of a social greeting behavioral intervention.

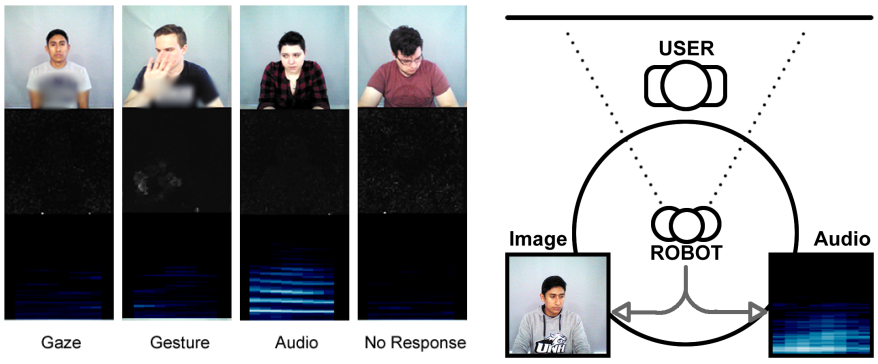
2.5.1 Dataset Description: Social Greeting Behavioral Intervention

My end-to-end model is evaluated on a social greeting focused behavioral intervention in which human participants are encouraged to respond to a greeting provided by a robot. The robot begins the intervention by waving and verbally greeting the participant who can respond in either a *compliant* or *non-compliant* manner as defined by the clinician (a human who remotely operates the robot). Having observed the response the robot can execute one of three actions: a prompting action (**PMT**), a verbal reward followed by the end of the interaction (**REW**), or an unrewarded conclusion to the interaction (**END**). If a compliant response is observed then the robot is expected to execute the **REW** action to reward expected behavior. If the response is *non-compliant* then the robot should first execute **PMT** and encourage the human participant to perform in the expected manner. If a second non-compliant observation is observed the robot should terminate the interaction with **END**.

2.5.2 Dataset Collection

I collected data to train the DQN model through an IRB-approved user study. Volunteers were recruited to take the role of a student while a tele-operated NAO humanoid robot took the role of a teacher to conduct the social greeting behavioral intervention.

Six volunteer participants without ASD from the University of New Hampshire (4 male, 2 female) were recruited for the purpose of collecting demonstration data. Participants performed a total of 18 interactions with the robot. In 12 of the interactions the participant



(a) Various Participant Responses (b) Data Collection Setup

Figure 2.2: Social Greeting Behavioral Intervention Data Collection

complied with the robot's requests, and either ignored or refused to greet the robot in the remaining sessions. Half of the compliant interactions were delivered in order to elicit at least one prompt.

The participants were told to respond to the robot using a specific combination of

- Gaze: maintaining visual contact with the robot
- Gesture: responding to the robot's prompt with a gesture (a wave)
- Audio: responding to the robot's prompt audibly (saying "hello").

Example inputs of the different response types are depicted in Fig. 2.2a. Each row of the figure captures a different modal input: RGB video, optical flow, and audio as a Mel-Spectrogram (C_i, P_i, and A_i respectively). In this intervention responses that consisted of only gaze were considered to be non-compliant as the participant had failed to follow the robot's directions (to say "hello" to them).

The complete dataset is composed of 155 videos depicting the **PMT** action, 118 videos depicting the **REW** action, and 73 videos depicting the **END** action. The RGB data in the **REW** and **END** actions is mirrored to increase the size of the under-represented actions. The videos were recorded at approximately 10fps and varied between 106 and 184 frames in length. Fig. 2.2b shows a birds-eye view of the data collection setup.

2.5.3 Training

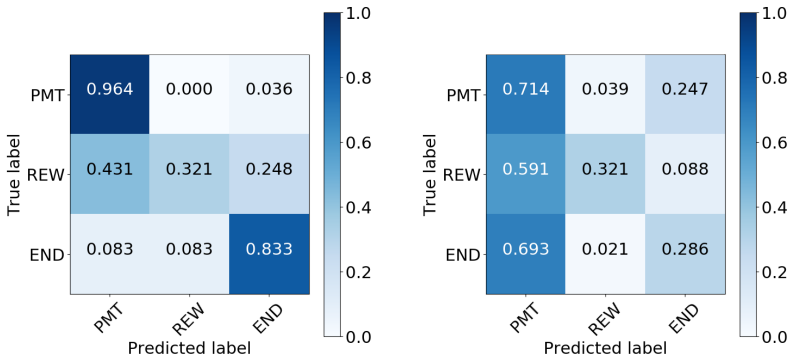
In contrast to traditional DQNs which learn in an unsupervised domain, my network is trained in a supervised manner. The collected tele-operated demonstrations of the behavioral intervention are separated into state-action pairs and the model is trained to associate a given video observation with a particular action. Anticipated future directions of this work consider policy learning of sequential tasks that can be expressed through multiple trajectories. In these applications the role of expected value and cumulative reward may be important when considering the most effective approach to completing a task. However, within the context of the social greeting application reward is synonymous with mapping of states to actions. Given this motivation I hand-engineered a reward function that captures the intent of a clinician performing this therapy. A reward value of 1 is given for the correct execution of terminal actions (**REW** or **END**) and a reward of 0.1 is awarded for the appropriate execution of the **PMT** action. A discount factor of 0.9 was used to penalize the length of the interaction. When evaluating the model in simulation 18 videos from each action were set aside for validation and training was conducted on the remainder. All of the demonstrations were used to train the real-time model and all are assumed to show correct and optimal behavior (i.e. no adversaries).

2.6 Results

The effectiveness of this end-to-end model is assessed in simulation and in real time. The simulated results are used to ascertain the importance of representing partially observable information in the state representation.

2.6.1 Simulated Results

I compare the accuracy of a model that possesses and a model that lacks the temporal information provided by p_t. The model that lacks the temporal information is identical in



(a) Temporal Data Present (b) Temporal Data Absent

Figure 2.3: Simulation Results

structure to the model depicted in Fig. 2.1 with the exception that p_t is not concatenated with the output of $FC1$ and instead the 1×3 output of $FC1$ is passed directly to $FC2$. The model's ability to correctly select an action for a given input is measured as accuracy.

When comparing these two models (Fig. 2.3), identical accuracy is observed when selecting the **REW** action indicating that both models are able to recognize the features necessary to identify a compliant response in the human participants. This is expected as the delivery of the **REW** action is independent of the presence of p_t. However, identification of non-compliant responses differs between the models. When p_t is present the overall accuracy of the model is higher and the **END** action is labelled correctly 83.3% of the time. When this information is absent the model has a harder time differentiating **PMT** from **END** and is only able to correctly identify END actions 28.6% of the time. Without the the hidden state information (the action history) the model is unable to correctly replicate the demonstrated skill.

2.6.2 Real Time Results

Observation Accuracy

When evaluated in real time on human participants (Table 2.1) the modified DQN is able to achieve an overall accuracy of 67.8% (as a measure of correct action predictions) demon-

strating that the model has learned many of the features required to perform the activity. In most situations it was able to correctly select the appropriate action to respond to a given human response. The model performs most effectively when auditory responses are delivered (75% with no other expression and 81.3% with gaze but no gesture) since the participants show greater uniformity in their vocalizations (they all responded by saying "hello" without additional modification or inflection). In contrast poorer accuracy is observed when inferring gestures (25.0% by itself and 6.3% with gaze but no audio). I attribute this loss in accuracy to the more variant manner in which people wave, including between two successive gestural responses. Some waves, despite being visually similar, are delivered over different durations or start at different times relative to the beginning of the video. From these results it can be inferred that the model's representation of a wave is overly specific and generalizes poorly to variations in duration. Given how common such scenarios are in real world observations it is critical that a *duration invariant* alternative to the current standard for temporal inference be employed. This work leverages an LSTM layer to learn temporal representations from the data and is subsequently responsible for correctly recognizing temporal patterns, such as auditory signals and gestural response, as they are expressed in the data. Later works that leverage LSTMs for video inference agree that LSTM models can easily fixate on frame level variations in the data [9, 29, 30] which can define temporal patterns in terms of an explicit duration. My results, reinforce this assertion.

Sequence Inference

My hesitations about the LSTM layer required further analysis and a subsequent investigation indicated that the layer may have learned patterns that were not ideal for representing the task. Fig. 2.4 depicts how the q-values of the different actions change over the course of an observation (green, red, and blue depict the **REW**, **END**, and **PMT** actions respectively). It can be observed that after an auditory response acknowledging the robot (Fig. 2.4 (a) at frame 105) there is a mandatory period of silence that has to follow the audio

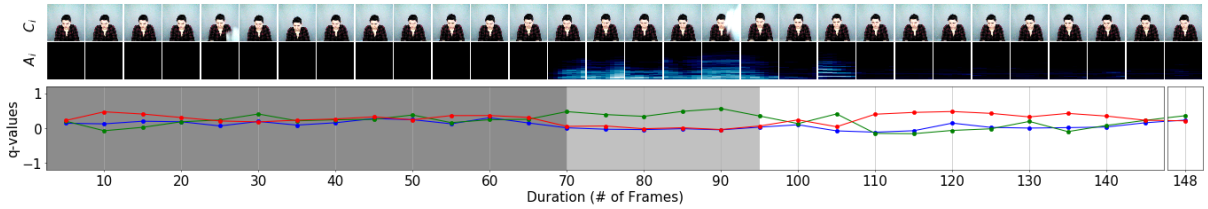
Responses			
Gaze	**Gestural**	**Auditory**	**Accuracy**
No	No	No	95.8%
No	No	Yes	75.0%
No	Yes	No	25.0%
No	Yes	Yes	68.8%
Yes	No	No	87.5%
Yes	No	Yes	81.3%
Yes	Yes	No	6.3%
Yes	Yes	Yes	37.5%
Total			**67.8%**

Table 2.1: Real-time Results of Varied Responses

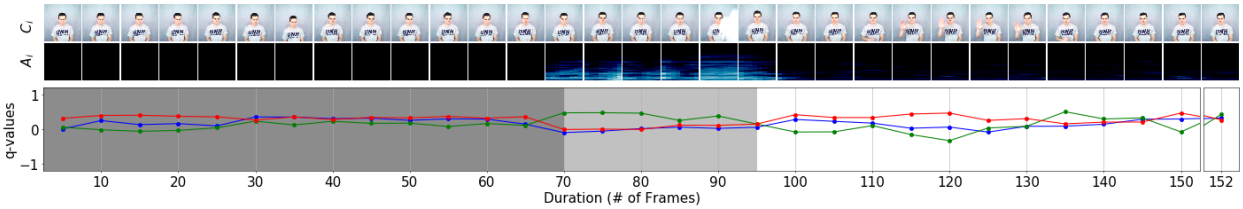
signal before it can be correctly identified as a compliant response. In Fig. 2.4 (b), when a gesture is performed (frames 105-130) it is only when the wave has concluded that the observation is correctly classified. I consider these two responses to be inconsistent with what a teacher of the activity would consider as compliant criteria for participant responses. Furthermore, these results indicate that the model has developed specific, duration dependent representations of these activities. This prompts the question that, if a response had been performed at a more modest pace or been delivered after some hesitation would that activity have been incorrectly identified using the current model? The incongruity between what is expected and observed given the entire video as a visual input suggests that the sequential, frame-aggregation approach employed by the LSTM layer may inhibit its ability to recognize the truly important parts of the video observation. A solution to this involves looking at the video from a global scale in order to recognize where the important aspects of the observation start and end and what long-term dependencies exist among the data.

2.7 Conclusion

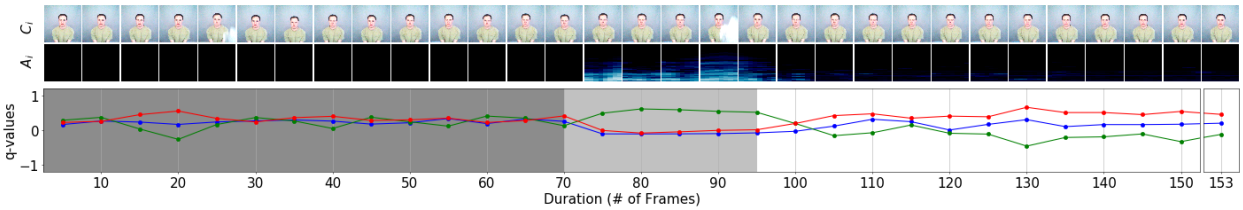
My exploratory work into modelling a high-level skill with an end-to-end deep learning framework sheds light on two specific limitations where inadequate temporal modelling can hinder model performance. The first, is in capturing hidden aspects of the world state not



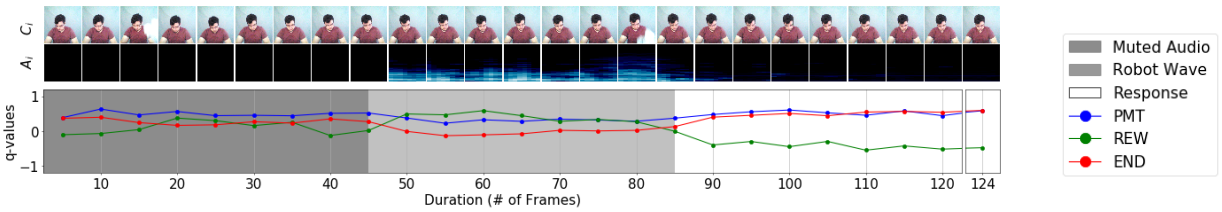
(a) Auditory Response



(b) Gestural Response



(c) Gaze Response



(d) No Response

Figure 2.4: Q-values of Videos Segmented at Different Lengths.

visible in observations. I leveraged action history information in the definition of p_t to prove the importance that this data represents. A more complex representation would represent not only prior action histories but potential loops within a policy. This research direction was explored in collaborative efforts with a Master's student [14, 31] who would later go onto defend this work as his final thesis. The other aspect of high-level tasks in need of robust temporal modelling is the observations themselves. Specifically, the absence of duration invariant structures prevents robust generalization of activities as they are expressed over different time periods. This is evident in the gestural responses depicted in the social greeting behavioral intervention domain. Furthermore, the reluctance of contemporary approaches to investigate full-length video in a single architecture results in a breakdown in the ability to model long-term dependencies, at least where the LSTM layer is concerned.

2.8 Contributions

The described work was presented in the 26th (2017) IEEE International Symposium on Robot and Human Interactive Communication (RO-MAN) [15] and the 2018 ACM/IEEE International Conference on Human-Robot Interaction [16]. Compared to earlier literature on high-level LfD, the approach described here does not make simplifying assumptions about perception nor does it rely on user-defined visual feature extraction. Rather, it directly learns the discriminatory visual features from the data. This work represented one of the first real-world applications of DQN and the first automated delivery of a behavioral intervention. Furthermore, it established two critical challenges when modelling temporal features: 1) perceptual aliasing and a need to develop temporal reasoning solutions that could incorporate hidden state information into the state estimation and 2) the need to represent temporal information as it is expressed in video observations in a duration invariant and scalable manner. The former challenge would be investigated in collaborative works published in the 28th (2019) IEEE International Symposium on Robot and Human Interactive Communication (RO-MAN) [31] and the 2019 IEEE International Conference on Robotics

and Automation (ICRA) [14]. The later challenge is the basis for this dissertation.

CHAPTER 3
LITERATURE REVIEW

3.1 Introduction

Many inference architectures use representations of time to model sequential information such as video. Traditionally, this has been accomplished using probabilistic temporal models [4, 32, 33, 34, 35] , frameworks for temporal reasoning that leverage principled definitions of temporal logic. As these models and their temporal representations have increased in complexity, they have become better suited to representing time in a general sense and modelling temporal representations at scale. However, the advent of deep learning has had a profound change in the manner in which temporal features are developed. Rather than adhere to the tried and true temporal logic used in classical models, new approaches attempt to learn unique temporal descriptors from the data. This approach has been successful when validated on thousands of short clips from benchmark video datasets, but does not scale to longer videos or sparse datasets.

As context for my literature review I describe Allen's Interval Algebra and Graph Convolutional Networks. Allen's Interval Algebra has been leveraged by Time-Interval Models (Section 3.3.3) and Graph Convolution Networks play a critical inference role in my proposed approach to overcoming the challenges of representing temporal features in video data (Chapter 4). Following this background information, I provide an overview of various classical temporal learning models and discuss the factors that led to their fall from popularity. This is followed by a discussion of deep learning-based approaches to modelling temporal data and their weaknesses in the face of inconsistent action durations and video-scale features.

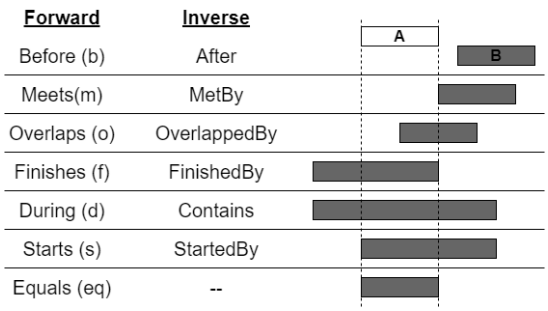


Figure 3.1: The 13 Interval Temporal Relationships

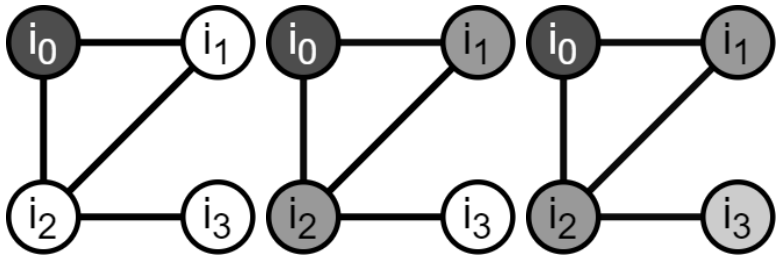
3.2 Background

I begin by discussing the background concepts of Interval Algebra and Graph Convolutional Network, the two tools used by my proposed approach for learning temporal features.

3.2.1 Interval Algebra

Allen's Interval Algebra is a temporal logic that describes 13 interval temporal relationships (ITRs) that can occur between two events, periods of feature expression with explicit start and stop times [36]. Allen defines thirteen different temporal relationships: 6 forward relationships, 6 inverse relationships, and 'equals'. Fig. 3.1 depicts the different relationships visually. Two distinct spatial events 'A' and 'B' are related by the manner in which they begin and conclude.

Describing temporal features in terms of Allen's Interval Algebra has several ideal properties, the most significant to my work is that the relationships are duration invariant. ITRs are defined by how two different event's start and stop times relate to one another. Subsequently, the representation of an ITR does not need to explicitly capture the time when each event occurs and the duration that extends between them. This property has been the basis for prior probabilistic temporal modelling approaches that aim to learn discriminatory aspects of video [4]. However, each instance in which a probabilistic temporal model used interval algebra did so in conjunction with user-defined features. No works have explored the use of ITRs with data-derived features. This is likely a combination of the storage de-



(a) l=0 (b) l=1 (c) l=2

Figure 3.2: Progressing Convolutions in a GCN

mands of this approach which increases quadratically with the inclusion of additional spatial features, and the typically large and uninformative output generated by standard spatial feature extractors. I address these issues by leveraging bottleneck structures [37] to refine the selection of data-driven features and Graph Convolutional Network (GCN) to identify which of the many extracted ITRs are discriminatory given the visual data.

3.2.2 Graph Convolutional Network

Graph Convolutional Networks are a recent advancement in neural networks that allows for deep learning on graphical structures [38]. GCNs operate under the same principles as a CNN: discriminatory features of the graph are learned in a local area and through subsequent convolutions a hierarchy of more complex, and more distant features are developed. The D-ITR-L pipeline extracts the ITRs present across an entire video and presents them in a graphical format. I use a GCN to learn the discriminatory ITRs present in this graphical representation for inference tasks. GCNs trained on ITRs are able to recognize complex temporal relationships that are unique to specific inference labels. The representations learned in this fashion are duration invariant (in that a single representation can capture the presence of an action regardless of the explicit duration over which it is expressed) and can identify long-term dependencies as they are expressed across the entire length of a video.

The design of the GCN is similar in principle to that of traditional convolution-based architectures in that the GCN attempts to match a kernel feature representation (in this

case a sub-graph) to an input (the input graph) and develops an output that denotes where and how well the kernel matched parts of the input. The critical difference being that in CNNs the local area is defined spatially by adjacent pixels or matrix locations. Whereas, in the GCN the local area is defined by edges to adjacent nodes. Consider the graph in Fig. 3.2a which is composed of a set of nodes ($i \in I$), each of which is defined by a vector of properties (h_i^l where $l \in L$ indicates the layer/depth of the GCN) and a set of un-directed edges connecting nodes together. A single convolutional operation applied to node h_i^l of the graph identifies and aggregates the feature vectors of the central node and the nodes that are directly attached to it (j denotes a node in the neighborhood of node i, N_i). The aggregation method is user-defined but in the context of this work is an averaging operation. The aggregated vector is then multiplied by weight values W_l and passed through an activation function (σ) generating a new more complex feature representation (h_i^{l+1}) that will go on to define node i in the next convolution (Fig 3.2b). This operation is formalized in equation 3.1 where c_{ij} is a constant normalizing factor. This work uses the average as an aggregation method and therefore c_{ij} can be defined as the degree (number of edges) of node i.

$$h_i^{l+1} = \sigma\left(\sum_{j \in N_i} \frac{1}{c_{ij}} h_j^l W^l\right) \tag{3.1}$$

Each convolution of a GCN aggregates the information in vertices that are an additional hop away from the source (or starting) vertices. Therefore, each convolution operation serves to develop a more complex latent representation that considers more distant nodes (Fig 3.2c). However, this does restrict the information that a singular node representation can capture to a neighborhood of nodes that is up to a maximum of L edges away. Many extensions to the general GCN structure exist. My work uses the Relational-Graph Convolutional Network (R-GCN) [39] which allows for more control when aggregating edge labels by redefining the neighborhood function to include only those with a given edge label. New representations are defined using equation 3.2.

$$h_i^{l+1} = \sigma\left(W_0^l h_j^l + \sum_{r \in R} \sum_{j \in N_i^r} \frac{1}{c_{ir}} h_j^l W_r^l\right) \tag{3.2}$$

3.3 Classical Methods

Traditional approaches to modelling temporal features have been addressed by a selection of probabilistic graphical models that focus on modelling time. These architectures have evolved in complexity and their representations of time have become increasingly rich in order to model a greater breadth of data. Probabilistic Temporal Models initially developed temporal relationships in the context of singular time-slices, but have since developed to represent relationships over intervals [35, 36, 40, 41]. I consider three general approaches to representing time in probabilistic models: time-slice approaches, grammar parsing models, and time-interval models.

3.3.1 Time-Slice Models

The simplest and most well-known approaches to modelling time are the Hidden Markov Model (HMM) and Dynamic Bayesian Network (DBN). HMMs capture the expression of a hidden state by evaluating observations generated by the system over time. DBNs are Bayesian networks that capture stochastic changes between timesteps. Both of these methods, and their derivatives (Dynamically Multilinked HMM[42], Switching Hidden Semi Markov Model [43], Coupled HMMs [44], Propagation Nets [45], etc.) relate information using singular time-slices and do not consider concepts such as duration. Modelling of temporal features using time-slices is accomplished through point-based relationships [46] and can only capture concepts such as *before*, *during*, and *after*, they are unable to express complex relationships such as the overlap of different concurrent events.

3.3.2 Grammar Parsing Models

Grammar parsing methods derive from natural language processing methods and attempt to learn a temporal representation that matches observed temporal data. These models lack an explicit temporal logic but are capable of mimicking more complex relationships using a combination of point-based relationships. Some examples of methods that fall within this category are Probabilistic Time Petri Nets [47, 48] and Suffix Trees [49]. Both methods use concepts of *before*, *during*, and *after* in the context of when features begin and end. These implementations are effectively precursors to time-interval calculus, a temporal logic employed in time-interval models.

3.3.3 Time-Interval Models

The richest and most complex representation of time in probabilistic temporal models are those that consider relationships between events in the context of intervals. These approaches leverage the temporal calculus of Allen's Interval Algebra (discussed at length in Section 3.2.1). Interval algebra describes temporal features as a measure of their temporal overlap. The Probabilistic Temporal Network (PTN) [40] develops a Finite State Automata (FSA) that captures the overall temporal structure of an activity using interval algebra. The FSA in this work is similar in structure to the ITR graph generated by D-ITR-L, as both represent spatial events as the nodes of a graph and use temporal relationships when defining the graph's edges. However, while the FSA in PTN is a learned representation of a class, my implementation is an encoding that captures an exhaustive description of all observed relationships within a video. The ITR graph is used for inference within D-ITR-L which is conducted through the use of a GCN. The Interval Temporal Bayesian Network (ITBN) [4] extends the concepts of PTN further by using stochastic representations of the interval relationships within the FSA. Stochastic temporal descriptions are a non-deterministic method for representing temporal relationships and provide the ITBN greater flexibility and robustness in representing complex domains. Models developed using time-interval based

representations such as the ITBN and PTN are complex and checking temporal consistency of triangle-based temporal relationships and their need to evaluate all possible network structures can become computationally intractable with an increase in network complexity [4, 35]. Nevertheless, ITBN and PTN continue to act as benchmarks for comparing probabilistic temporal models and are referenced frequently in the context of modern activity recognition works [50, 51, 52, 53, 54].

3.3.4 Validation

Probabilistic temporal models have been applied to many cross-disciplinary problems that challenge their ability to represent temporal features in a duration invariant and scalable fashion. The tasks being addressed are challenging domains composed of visually similar, real world observations such as emerging sports strategies [4, 55], transforming facial expressions [33, 34], and vehicle loading and unloading [4]. However, in each of these applications temporal features have been developed using spatial feature information that has been extracted according to hand-picked methods. For example, transforming facial expressions were identified by combining probabilistic temporal models with engineered features that can identify parts of a face. Another example is present in identification of different plays in an American football game which were learned from frame-level annotations that capture the position of individuals on a football field. A common approach when modelling human activities is to overlay a skeletal model on an individual and perform inference using human joint angles [56, 57] but this makes assumptions about the video content of the observation. Namely, that it contains a human, that the human demonstrates skills that can be matched by the skeletal model being employed, and that the quality of the video is good (i.e. that there is little occlusion and that the skeletal model will perform well). These methods are not data driven and while they provide examples of what can be achieved with the rich representation of temporal features they also make strong assumptions regarding the capabilities of modern perception systems.

3.4 Deep Learning Methods

The remarkable success of deep learning architectures in visual inference domains, such as image classification [58] and policy learning [21], have stirred renewed interest in tasks that model complex temporal data such as signal processing and video recognition. The data driven spatial feature finding abilities of deep learning architectures has been critical to their many achievements. However, the approaches taken to capture temporal information using deep learning models disregard the accomplishments of earlier works. Rather than leverage established temporal representations, such as time-interval or time-slice concepts, deep models attempt to learn temporal concepts from the ground up. Temporal representations in deep learning architectures are generated through two methods: Recurrent Neural Networks and convolution-based approaches. I discuss both and elaborate on the design flaws that prevent these models from being *duration invaraiant* and representing *video-scale features*.

3.4.1 Recurrent Neural Networks

Recurrent Neural Network (RNN), and its derivatives Long Short-Term Memory [20, 59, 60] and Gated Recurrent Units [61, 62], can be used for signal processing and time series inference in a duration invariant manner given specific criteria. These methods aggregate feature expression sequentially and periods of consistent feature expression are conglomerated into a single representation. But, this is unrealistic given a real world input, and RNNs frequently over fit on sparse and noisy data[8, 9]. Subsequently, these models are limited to representing short-term patterns in the data rather than variable duration, long-term dependencies. This was observed in my exploratory work (Chapter 2), in which my use of LSTM-learned patterns generalized poorly to novel data and adopted inconsequential properties of human responses into its latent representation.

3.4.2 Convolutional Architectures

The image processing properties of CNNs have encouraged research into convolution-based representations of time [63, 64]. This has been accomplished in two variations: temporal inference using 1D convolutions filters [13, 63] and spatio-temporal inference using 3D convolutional filters [9, 65]. Both variations adhere to a hierarchical design in which larger and more complex representations are derived from lower level concepts. Temporal representations captured in this manner have a fixed duration, duration invariant inference, and a limited scale defined by the depth of the convolutional network. In order to capture a video-scale feature, a deep enough representation must exist in the deep learning architecture to span the duration between the component feature expressions [63]. For example, when making tea a video-scale feature might capture a relationship between adding sugar to a cup at the beginning of the video (frame 1) and stirring the cup at the end (frame 100). Using 1D convolutional filters that cover a duration of 5 frames, representing this video-scale feature as a latent representation would require a model that is 25 layers deep. For comparison many of the deepest architectures today such as ResNet-50, ResNet-101, and ResNet-157 are used in image recognition [66]. These models despite possessing hundreds of layers of convolutions only collapse their representation a handful of times (between 3 and 10 times). By refusing to expand the size over which they develop their representation they can develop a greater number of spatial features without incurring large computational costs. Even when considering these accommodations these architectures perform tens of billions of floating point operations per second and must be distributed between 4 and 16 GPUs. The challenges of developing broad spatial features (features that span the entire spatial dimensions of a frame) are well-established and have developed a number of *ad hoc* solutions [67].

CHAPTER 4

DEEP INTERVAL TEMPORAL RELATIONSHIP LEARNER

This chapter discusses Deep Interval Temporal Relationship Learner (D-ITR-L): a temporal wrapper that transforms autonomously learned spatial features from a video, as identified by a data-driven architecture (in this work a CNN), and converts them into a graphical representation depicting the temporal relationships expressed across a video input. D-ITR-L is the core contribution of my work.

4.1 Temporal Feature Learning Using D-ITR-L

Algorithm 1 D-ITR-L

Input: M, a four dimensional tensor $< F, H, W, T >$, where F: number of features, H: height, W: width, and T: time
Output: A, an ITR Graph
1: $IAD \leftarrow$ maximum function applied to M over H and W ▷ O($F \times H \times W \times T$)
2: $IAD \leftarrow$ IAD thresholded using Φ_f ▷ O($F \times T$)
3: $E \leftarrow$ a set of events are extracted from M ▷ O($F \times T$)
4: $A \leftarrow IdentifyingTemporalFeatures(E)$ ▷ O(E^2))
5: **return** A

Deep Interval Temporal Relationship Learner (D-ITR-L) is a wrapper that extends the feature learning abilities of traditional CNN architectures. D-ITR-L operates as a pipeline as shown in Fig. 4.1 (and broadly described in Algorithm 1) transforming the spatial features identified by a CNN backbone model into temporal features (in the format of an Interval Temporal Relationship (ITR) graph) which are used as the basis for training a GCN as part of a greater end-to-end model, e.g., for state estimation.

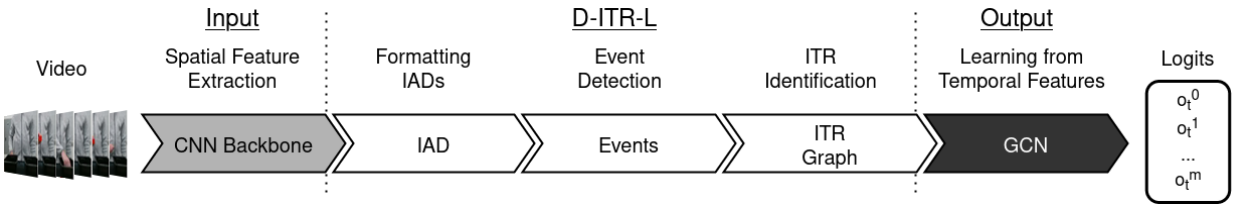


Figure 4.1: The Deep Interval Temporal Relationship Learner Pipeline

D-ITR-L begins by leveraging the output of the CNN backbone to develop a novel format that I term an Interval Algebraic Descriptor (IAD). The IAD denotes events, periods of time in a video when spatial features identified by the CNN are expressed. These events are combined to define temporal features which are described in the context of abstract temporal relationships defined by Allen's Interval Algebra [36] (Section 3.2.1), a duration invariant temporal logic. Temporal features collected across the video are organized into a graphical format allowing for direct inference of distant temporal relationships (ITR Graph). The graph of temporal features generated by D-ITR-L can be used as a superior alternative to the standard, pixel-based input of modern neural network inference architectures. A graph convolutional network (GCN; Section 3.2.2) is used to infer the discriminatory aspects of the temporal features present in this graphical presentation and the resulting logits, a vector or raw prediction values, can be used in various applications including activity recognition and policy learning.

4.1.1 Spatial Feature Extraction

The first step in the D-ITR-L pipeline is identifying a set of informative spatial features from which to build complex temporal relationships. These spatial features are obtained from a pre-trained, user-selected CNN backbone. It is assumed that the CNN backbone has been fine-tuned to recognize spatial features that occur in the dataset upon which the D-ITR-L will be evaluated. The structure and steps to train the backbone networks is outside the scope of my contribution and is elaborated in Sections 5.4 and 6.3.1 when describing hyper-parameter tuning in the context of specific applications.

A common modification made to all spatial feature extractors that leverage D-ITR-L is their use of a bottleneck to constrain the number of spatial features used to generate an ITR Graph. Identifying ITRs generates a pairwise number of edges when given a fixed set of spatial features. I reduce the number of features output by the backbone model by inserting a 1×1 convolutional bottleneck layer [66] prior to the model's inference layer. A more detailed discussion of how the bottleneck is implemented is provided in Appendix A.

4.1.2 Formatting Interval Algebra Descriptors (IAD)

Passing a video into a traditional CNN video architecture generates an activation map (M): a 4-dimensional tensor ($F \times H \times W \times T$) denoting the relative expression of learned features, where F: the number of feature, H: height, W: width, and T: time. Each activation map is transformed into a novel structure that I term an Interval Algebraic Descriptor (IAD) for further processing. The IAD describes when spatial features are expressed in the duration of a video. The IAD reduces the four-dimensional activation map down to a two-dimensional structure ($F \times T$) by collapsing the spatial dimensions (H and W) to a single value. This is accomplished using a maximum operation, applied over the spatial dimensions. The maximum function operates in O(n) time as it must investigate each location in the activation map. Therefore, it operates in O($F \times H \times W \times T$) time, though this operation is typically expedited through parallelism. Compared to other operations the maximum function effectively captures relative feature expression at a time step regardless of how sparse the feature expression is across the spatial dimensions. The reduced representation clearly indicates the relative expression of each feature at each time. Fig. 4.2a shows an example IAD constructed from a video lasting 195 frames. The relative expression of the 32 different spatial features, identified using a VGG-16 backbone CNN, is depicted in greyscale (with darker shades denoting increased expression). Time is expressed along the x-axis and the features as independent rows along the y-axis. Zoomed regions are identified in red for clarity.

4.1.3 Event Detection

Event detection can be performed from the IAD by determining the explicit start and stop times when a spatial feature is actively expressed. This distinction is useful for recognizing the nature of the temporal relationship that exist between two spatial features. A different threshold value (Φ_f) is selected for each feature ($f \in F$) using the average expression of that feature over the entire dataset. I investigated more complex threshold values by varying the average by a factor of the standard deviation, but none were as effective as the one described. Alternative threshold values exist and identifying if a better value exists is a potential future research direction. This requires a single pass over the training dataset to identify Φ_f and a subsequent pass to threshold the data. Values in the IAD that fall below (Φ_f) are set to 0. The thresholding operation occurs in O(n) time, this equates to an operating time of O($F \times T$)

Each event (the regions greater than Φ_f) is defined with a four-tuple $< t_s, t_e, f, f_{mx} >$ denoting the timestamps when the event started (t_s) and ended (t_e), and a description of the content of the event using the feature label (f) and the maximum expression of that feature across the event (f_{mx}). Fig. 4.2b depicts the events present in the IAD shown in Fig. 4.2a. The intensity of the shaded regions matches the value of f_{mx}. Extraction of events takes place over O($F \times T$) time as each location in the IAD must be checked to identify whether the value is active or not. The event extraction process generates E unique events. In the worst case there will be $(F \times \frac{T}{2})$ events in E in the unlikely case that an spatial feature is expressed on every odd time increment and is not expressed on every even time increment.

4.1.4 Interval Temporal Relationship Identification

The events (E) identified in Fig. 4.2b are the basis for recognizing the presence of ITRs in the input video. Pseudocode describing this process is provided in Algorithm 2. Additional clarification is provided by Fig. 4.3a which shows a simplification of a thresholded IAD of a video. The example captures four example events expressed across time. Identification of

Algorithm 2 IdentifyingTemporalFeatures

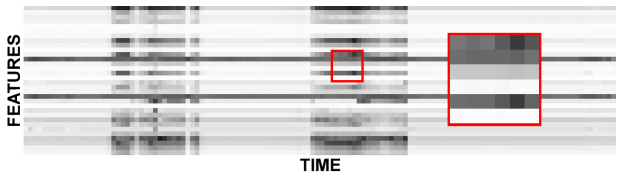
Input: E, a list of spatial events each defined by a tuple $< t_s, t_e >$
Output: List of Temporal Features

$A \leftarrow []$ ▷ List of ITRs
$E \leftarrow E.sort()$ ▷ Sorted in order of t_s followed by t_e
for i in range (0, length(E)) **do**
    $r \leftarrow None$
    for j in range (i, len(E)) **do**
        if r != 'before' **then**
            $r \leftarrow IdentifyITR(E[i], E[j])$
        end if
        $A \leftarrow (E[i], r, E[j])$ ▷ Temporal Feature added to List
    end for
end for
return A

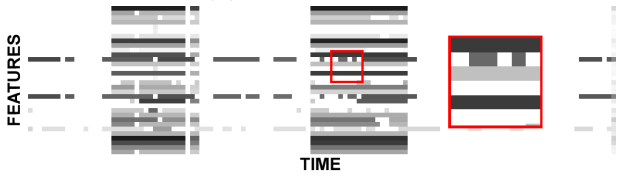
Algorithm 3 IdentifyITR

Input: x, y, two spatial events each defined by a tuple $< t_s, t_e >$
Output: r, an ITR Relationship

if $x.t_e < y.t_s$ **then**
    return 'meets'
end if
if $x.t_s < y.t_s$ & $x.t_e < y.t_e$ & $x.t_e > y.t_s$ **then**
    return 'overlaps'
end if
if $x.t_s < y.t_s$ & $x.t_e > y.t_s$ **then**
    return 'during'
end if
if $x.t_s > y.t_s$ & $x.t_e == y.t_e$ **then**
    return 'finishes'
end if
if $x.t_e < y.t_e$ & $x.t_s == y.t_s$ **then**
    return 'starts'
end if
if $x.t_s == y.t_s$ & $x.t_s == y.t_s$ **then**
    return 'equals'
end if
return 'before'



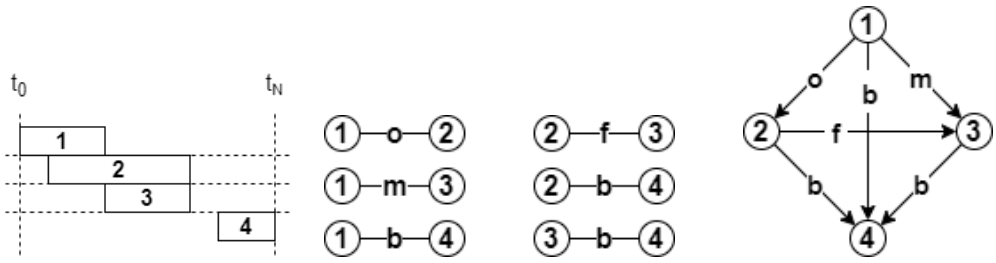
(a) A raw IAD



(b) A thresholded IAD

Figure 4.2: Event detection using IAD.

temporal features requires a pairwise comparison between the spatial events ($E(E-1)$) and is an $O(E^2)$ task, but heuristics learned in an earlier collaborative work [14] can be leveraged to reduce the number of computations that must be performed. The first consideration when reducing complexity is to only investigate *forward* ITRs (listed in Fig. 3.1) and avoid redundant calculations from *inverse* relationships. Directed relationships between events are replaced with undirected connections effectively reducing the number of unique ITR relationships from 13 to 7 (Alg. 3). The second approach to expediting temporal relationship identification is the application of an algorithm to sort the events in ascending order of t_s and t_e (line 2 of Alg. 2), and then iterating through the events in a pairwise manner to find the ITR that relates each pair of events (line 7). This approach takes advantage of logical truths about Interval Algebra, for example, if event 'A' is related to event 'B' by a *before* relationship, then an event 'C' that starts at the same time or later than event 'B' will also be related to event 'A' by that same ITR (i.e. *before*; Alg. 2 line 6). The algorithm limits the number of comparison operations that must be conducted to generate the entire set of temporal features from a video. Fig. 4.3b demonstrates the resulting list of identified ITRs, where we have related event 1 to event 2 by an "overlaps" ITR, event 1 and event 3 by a "meets" ITR, so on and so forth. This list of ITRs is subsequently assembled into an ITR graph. Events become the nodes (labelled with f_i, and weighted by f_{mx}) and the ITRs



(a) IAD Example (b) ITR List (c) ITR Graph

Figure 4.3: Transformation from thresholded IAD (a) to a list of ITRs (b) to an ITR Graph (c). ITR labels match those in Fig. 3.1.

become the edges. Fig. 4.3c depicts an ITR graph. The ITR graph is the collection of all temporal features in an input video.

4.1.5 Learning From Temporal Features

A GCN is used to perform inference on the ITR graph. The relational GCN (R-GCN) [39] learns the discriminatory relationships present in a graph whose edges are defined by a discrete set of labels rather than continuous values. This offers a more natural integration of the data. In my work I used the same R-GCN architecture defined in [39]. Modification of the hyper-parameters in this architecture may generate better inference from the ITR graph but would also represent an additional uncontrolled variable in the subsequent evaluation of this architecture. GCN inference begins at the nodes and with further convolutions extends a hierarchy to neighboring nodes. Through the use of the R-GCN different weights provide a bias to how different ITRs relate events, for example an 'equals' ITR is a less common ITR and is therefore more likely to carry greater temporal significance than the 'before' ITR which is relative common. The use of a GCN in inference allows for sufficiently deep GCNs to create a network of ITRs that describe complex dependencies among the temporal features. In the context of Fig. 4.3c, the combined relationships of event 1 "overlaps" event 2 which "finishes" at the same time as event 3 might be considered a discriminatory temporal feature compared to the other relationships in the graph. With an ITR graph as input, the output of the GCN is akin to the output of a regular CNN, a vector of values that

represent the contents of the video (logits). Using a real world example, logits describe the relative presence of high-level concepts in the video input such as evidence that someone is pouring water or turning on the stove in a tea making sequential task. Logits are raw and unrefined values and it is expected that they will be used in concert with other deep learning architectures to perform a specific function such as activity recognition (through the use of a softmax layer) or policy learning (through the use of a specific policy learning architecture).

4.2 Contributions

The D-ITR-L pipeline is the foundation upon which the rest of the work in this thesis has been established. D-ITR-L is a tool that transforms the pixel representation of a video into a graphical representation of interval temporal relationships. This representation presents information in a duration invariant and a scalable video-scale fashion. To the best of the authors knowledge no other models can capture video-scale features in a duration invariant manner using a data-driven architecture. D-ITR-L is validated in three experiments distributed over the next 3 chapters. Two descriptions of this work have been included in separate submissions to the IEEE International Conference of Robotics and Automation 2022.

CHAPTER 5
TEMPORAL FEATURES FOR POLICY LEARNING

Temporal features captured in an ITR Graph describe how spatial features are expressed in time and can be substituted in any inference task that would typically be modelled using spatially-derived features. I demonstrate this in the context of policy learning from visual demonstrations. Training robots to learn from long video observations with redundant spatial features represents a markedly different visual format than what is encountered among other computer vision tasks, and other CNN-driven policy learning architectures. The emphasis on visually-similar video observations creates two challenges of this learning environment: *durational variance* (how features can be expressed at different scales of time) and *video-scale features* (feature expression at the scale of a full video as opposed to a over short snippets), I employ D-ITR-L to address both. In this chapter, I discuss how the D-ITR-L framework can be extended for use in policy learning and validate D-ITR-L's capabilities when tackling the challenges that dominate this visual domain. The code for this implementation is available at [68].

5.1 Related Works

Contemporary vision-based LfD architectures leverage the discriminatory feature learning properties of CNNs in their design. Many of these architectures are designed for low-level control applications and as a result use single frames or short clips of video (lasting fewer than 10 frames) for inference [69, 70]. By focusing on shorter-form video input these works do not require solutions to the challenges generated by long duration videos. These approaches

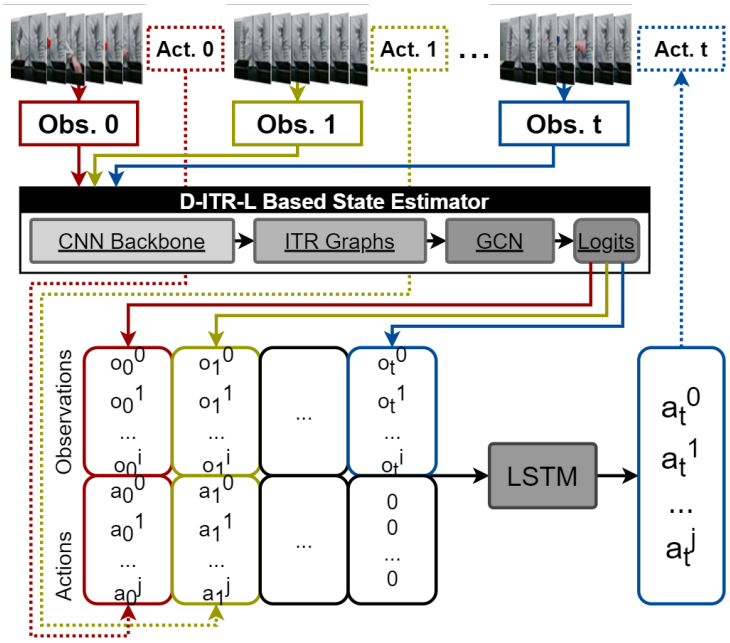


Figure 5.1: The D-ITR-L policy learning pipeline.

are often reactionary in nature, encouraging the learned model to respond to specific spatial stimuli as soon as they are observed. In contrast, vision-based high-level policy learners must perform inference over longer observations including video. These models typically rely on simplifying assumptions [19] or user knowledge [71] when implementing robot perception. The exception being my earlier work [16] which extracted useful features from full-length video input using a CNN-architecture (Chapter 2).

5.2 Extending D-ITR-L for use in Policy Learning

I present an architecture that relies on D-ITR-L for state estimation when learning the policy of a sequential task from visual demonstration. Through this approach performance of D-ITR-L and the critical role that temporal features play in learning vision-based task policies can be quantitatively evaluated when learning temporal features.

D-ITR-L was extended for policy learning in a manner consistent with Behavioral Cloning (BC) [72] in which the policy is learned as a mapping of states to actions using a set of labeled state-action pairs [73]. CNN-based architectures are often used for state estimation in

contemporary BC approaches that learn from vision data [73]. D-ITR-L uses temporal features, instead of raw pixel data, providing it with a greater state estimation ability which, in turn, contributes to better policy learning. A simple extension of D-ITR-L is described that uses a Long Short-Term Memory (LSTM) cell to infer actions over the state-action history, the entire system (D-ITR-L and LSTM) is trained in an end-to-end manner. LSTMs have been used for action selection in prior vision-based BC works [16, 74], where the discrete time steps and short duration observations have been particularly effective. In this application the LSTM is not used to infer information from the observation but instead from logits expressed in discrete time steps. The aforementioned challenges and criticisms of the recurrent architecture related to duration invariance and video-scale features do not apply here as this input does not deal with raw video input.

The D-ITR-L-assisted policy learning architecture operates as a pipeline (Fig. 5.1). A video-based observation of arbitrary duration taken at time (o_t) is fed as input into D-ITR-L to generate an I length vector of logits (o_t^i where $i \in I$). The length of I is user-defined and should be large enough to capture all of the potential observation states needed to define the policy. These logits (Section 4.1.5) are combined with the logits generated by observations in previous time steps and a one hot encoding of prior actions (a_t^j where $j \in J$). The length of J is defined by the number of available actions in the task. Zeros are used to represent the action to be inferred (a_t). The resulting two dimensional matrix is an estimation of the state (S) and is fed sequentially into an LSTM layer. The LSTM generates values for each of the policy's actions and the action with the highest value is selected to be performed in the subsequent time step (a_t).

5.3 Block Stacking

The strength of D-ITR-L at learning temporal features and, in turn, sequential task policy from visual demonstrations is best demonstrated through a task (and environment) where the same actions are executed over variable durations (*duration invariance*) and where task-

relevant spatial features are distributed over the entire length of the video and may even appear multiple times (*video-scale features*). Existing publicly available benchmark video datasets do not match these criteria. Additionally, the strong spatial bias exhibited by existing datasets allows them to be modeled using only their spatial features [7] (this is elaborated upon in Section 6.1.3). To that end, I designed a sequential block stacking task to evaluate D-ITR-L and its role in policy learning.

5.3.1 Problem Definition

In this task, a human moves colored blocks between two opaque containers while following any of the following rules at each step: move no blocks (n); move one red (r), blue(b), or green(g) block; move a blue block followed by a green block (bg) or vice versa (gb); or move two or three red blocks (rr and rrr respectively). These last two are examples of *video-scale features*. The use of opaque containers focuses learning on temporal features, preventing a single frame or short clip of the video from fully defining the observation. The goal of the experiment is for a robot to stack colored blocks in an order that matches the pattern demonstrated by the human, as shown in Fig. 5.3. The robot is allowed to select one action during each phase of the interaction to either stack a single colored block (R, B, or G) or pass (N).

5.3.2 Demonstration Set

Expert demonstrations are collected with a single human demonstrator and a tele-operated Sawyer robot (Fig. 5.2). Using a RealSense camera I recorded ten RGB videos in which a human moved blocks according to the eight aforementioned observations for a total of 80 videos. Three videos from each observation are set aside for testing purposes and the rest are used for training. To investigate the *duration invariance* aspect of D-ITR-L, I compared two variants of the dataset: one where the movement of blocks and collection of video was **consistently** timed using a metronome, and another where movements were executed

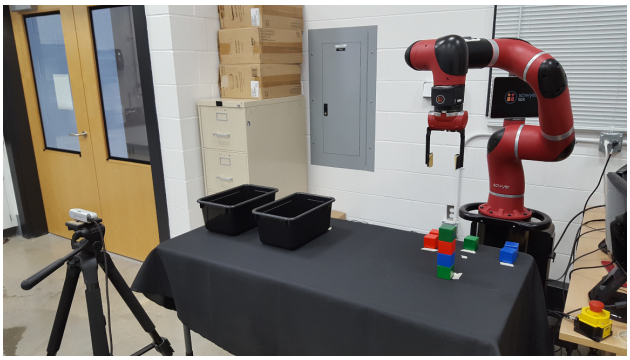


Figure 5.2: The Data Collection Environment

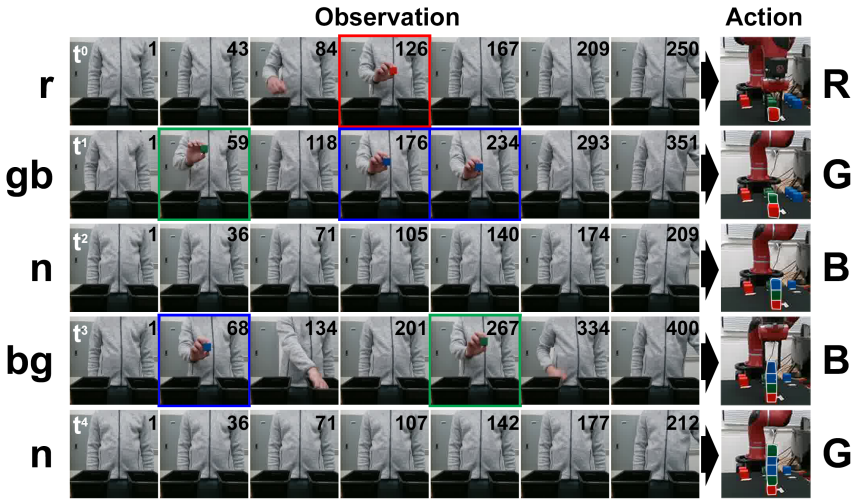


Figure 5.3: An Observation-Action Trace for Stacking 5 Blocks.

inconsistently.

Collecting full demonstrations in the block stacking task is time consuming. It requires several minutes of video to capture observations needed to stack a single tower. Furthermore, contemporary health concerns (COVID-19) required that all demonstrations be collected through a single individual (the author). To limit complications such as participant fatigue, demonstrations were generated in a procedural manner. This was accomplished by sampling one example from each of the eight observations (b,g, r, bg, gb, rr, rrr, or n). These observations were then shuffled together and mapped to appropriate robot action choices (R, G, B, or N). Each demonstration lasts 13 actions allowing each of the observations to be depicted once. Observations depicting no action (n) were used to pad the observation trace

to match the length of the action trace. An example demonstration lasting five action steps is depicted in Fig. 5.3. Due to tower instability real world evaluation on a robot was conducted using demonstrations trimmed to allow for the placement of only five blocks. A total of 100 demonstrations were generated of which 90 were used for training and the remainder were reserved for evaluation. Trajectory learning is beyond the scope of this work and it is assumed that the robot knows where the blocks are located and how to grasp and stack the blocks, the focus of this work is to generate a strong representation of the state using the latent temporal information present in the videos of this sequential task.

5.4 Backbone Model Preparation

Four CNN backbone structures are contrasted in this work. I selected two popular image inference (VGG-16 [75] and WideResNet [76]) architectures and two video inference architectures (I3D [65] and Temporal Shift Module (TSM) [11]). D-ITR-L and other temporal inference models described in Section 5.5 use the fixed spatial features identified by these architectures in their inference. It is entirely possible that a different selection of backbone models could generate a better set of spatial features and subsequently achieve better performance than the reported results. However, the merit of this work is not on improving the spatial representation but rather on developing strong temporal representations from available spatial features regardless of their quality.

5.4.1 Pre-processing

Due to the sparse nature of our videos the input is pre-processed using Gaussian blur and background subtraction [77]. Gaussian blur reduces the per-pixel variances common in observations from real-world video allowing for smooth application of additional pre-processing functions. Background subtraction masks static, background features such as the environment in an observation and instead highlights the information present in dynamic moments, those being moments when features move. The visual input was resized to match the archi-

Backbone	Inconsistent	Consistent
I3D	8	16
Temporal Shift Module	16	16
Wide ResNet	16	16
VGG-16	32	32

Table 5.1: Backbone-CNN Bottleneck Size in the Block Stacking Dataset

tecture it was being fed into.

5.4.2 Feature Bottleneck

Identifying ITRs generates a pairwise number of edges when given a fixed set of spatial features (Section 4.1.1). I reduce the number of features output by the backbone model by inserting a 1×1 convolutional bottleneck layer [66] prior to the model's inference layer. The number of features to reduce down to is user-defined. I performed a grid-search to find the best value for each model from 8, 16, 32, and 64 spatial features. Given the resource consumptive nature of D-ITR-L I identified that 64 spatial features was the most I was able to consistently place on a computer using 2 Titan X GPUs without instituting *ad hoc* solutions to limit the number of times that a given feature could be expressed as a unique event. Three instances of each backbone models were developed at each bottleneck size. The highest performing bottleneck was leveraged when comparing the different temporal inference models. The CNN-backbone models were all trained on the consistently timed and inconsistently timed datasets separately to better fit the learned spatial features to the data within each datasets. Subsequently, identical backbone models could achieve better accuracy with different bottleneck sizes. The bottleneck sizes used with these datasets are listed in Table 5.1.

5.4.3 Training

Each backbone network came with a pre-trained model which was fine-tuned to recognize discriminatory visual features in the block stacking dataset. The sparse expression of infor-

mative spatial features in the videos impedes learning. This limitation can be overcome by temporarily applying a max pooling layer to the back-end of my backbone, reducing along the temporal dimension when fine-tuning the backbone models. This process removes the expression of temporal features and focuses solely on the expression of spatial (in the case of image inference models) and spatio-temporal features (in the case of video inference models). The backbone network is trained to predict observation labels over 50 epochs, using a batch size of 8 videos, with an Adam optimizer utilizing a learning rate of 0.0001 and a cross entropy loss. After fine-tuning the network I discard the max pooling layer to allow for expression of features in time and allow the temporal inference architectures to function. I also fix the learned CNN features to prevent them from being further modified. Subsequent inputs to the backbone model generate *activation maps* (described in Section 4.1.2) and depict when the learned features occur in the input. The full policy learning model is trained to predict the appropriate action from state S using the same parameters used to fine-tune the spatial features.

5.5 Results

I compare D-ITR-L, which learns from temporal features, against three other data-driven deep learning approaches: *linear* inference (no temporal consideration), *LSTM* (a RNN), and *Temporal Convolutional Network (TCN)* [13] (a convolution-based approach). Many variations of RNNs and convolution-based models exist but they all suffer from the same structural limitations mentioned in Section 3.4. LSTM and TCN represent the most classic versions of their architectures and serve to demonstrate the fundamental weaknesses in their respective design principles. I investigate how these four implementations improve upon spatial features learned by two image-based (WideResNet and VGG-16) and two video-based (Temporal Shift Module and I3D) backbone architectures. As previously mentioned, vision-based LfD has not developed any specific novel tool for representing time or inferring from long-form video, subsequently, I do not include any specific LfD-based architecture

Backbone	Linear	LSTM	TCN	D-ITR-L
I3D	53.3%	53.3%	46.7%	**60.0%**
Temporal Shift Module	46.6%	70.0%	**90.0%**	**90.0%**
Wide ResNet	46.6%	76.6%	**90.0%**	**90.0%**
VGG-16	66.6%	**90.0%**	86.6%	**90.0%**

Table 5.2: Total Accuracy of Baseline and D-ITR-L-based Policy Learning Applications on Block Stacking given **Consistently** Timed Video Observations.

in this work. Each architecture generates state estimation logits from video observations which are used in the same end-to-end architecture described in Section 5.2. I investigate how D-ITR-L compares against the baseline models at capturing temporal features given the aforementioned challenges.

5.5.1 Duration Invariance

Variance in duration is present across all observations and I measure the accuracy (as a percentage of correct action predictions) across the entire dataset as opposed to a specific observation. When trained on the consistently timed dataset (Table 5.2), the linear, LSTM, and TCN models outperformed architectures trained on inconsistently timed data. This is expected given the many duration dependent architectures evaluated. Models using LSTM and TCN learned patterns that can shift (translate) along the time axis. These models outperformed the static representations of linear models. TCN was particularly effective when the duration of events was consistent. Among the CNN backbones, the video-based architectures (I3D and Temporal Shift Module) performed worse than the image-based architectures (WidesResNet and VGG-16). I attribute this to the increased challenge present in generalizing spatio-temporal features to video compared to just spatial features.

D-ITR-L dominated all other methods when task duration varied (inconsistently timed data; Table 5.3). I attribute this to the duration invariant feature representation that other models lack. Curiously, D-ITR-L tended to perform better using the more variable data than the easier to model consistently timed data. Feature expression (though inconsistent)

Backbone	Linear	LSTM	TCN	D-ITR-L
I3D	38.0%	38.0%	30.0%	**40.0%**
Temporal Shift Module	40.0%	82.0%	73.3%	**94.0%**
Wide ResNet	40.0%	65.8%	90.0%	**96.7%**
VGG-16	56.0%	72.0%	73.3%	**98.0%**

Table 5.3: Total Accuracy of Baseline and D-ITR-L-based Policy Learning Applications on Block Stacking given **Inconsistently** Timed Video Observations.

Obs.	Linear	LSTM	TCN	D-ITR-L
gb	100.0%	100.0%	66.7%	100.0%
bg	0.0%	100.0%	66.7%	100.0%
r	0.0%	0.0%	100.0%	100.0%
rr	0.0%	0.0%	33.3%	66.7%
rrr	100.0%	100.0%	33.3%	100.0%

Table 5.4: Action Accuracy of Baseline and D-ITR-L-based Policy Learning Applications on Block Stacking. The results are generated from the VGG-16 Backbone model.

may have been more easily delineated in the variable data.

Block movement events in the consistently timed dataset were sometimes performed in faster succession than in the inconsistently timed dataset. This caused similar feature expression to bridge adjacent block movement events. Identifying when features begin and end from this concentrated data is more challenging and can prevent accurate identification of temporal features. Regardless, where consistently timed data is concerned, D-ITR-L matched or improved upon the results of other methods.

5.5.2 Video-Scale Feature

Video-scale feature representation is assessed by how well a model distinguishes between the *bg* and *gb* observations (long-term dependencies) and the *r*, *rr*, and *rrr* observations (cyclical motifs). Accuracy is a measure of the model's ability to correctly select the next three actions following an observation. Three examples were used for each observation. The analysis in this section is presented in the context of the inconsistently timed results and the VGG-16 CNN backbone model (Table 5.4). Inconsistently timed results were collected with fewer constraints and are, subsequently, a better representation of real world data. VGG-16

performed the best of the backbone models investigated on this dataset according to Table 5.3.

The linear model failed to learn the video-scale features and moved the green block followed by the blue block for both *bg* and *gb*. Similarly, it moved three red blocks regardless of how many were actually depicted in the observation. LSTM was able to learn the visually dissimilar long-term dependencies, but was unable to learn the visually similar cyclical motifs and again moved three blocks in all instances. The feature expression in cyclical motifs is the same and thus distinguishing patterns from their frame-to-frame transition is more challenging. TCN was able to capture both video-scale features, but did so poorly. TCN uses duration dependent 1D convolutional layers and could represent the short r observation well, but when the distance between features increased the latent representation was unable to generalize.

D-ITR-L was able to distinguish between both long-term dependencies and all three cyclical motifs. D-ITR-L's results are not perfect and one instance of the *rr* observation was mis-interpreted as *rrr*. This inaccuracy can be traced to the CNN-backbone's inability to consistently recognize the red block feature. Noisy feature expression in the activation map cascaded through the D-ITR-L pipeline resulting in inaccurate event detection and subsequently incorrect feature recognition. D-ITR-L is a wrapper that acts upon the information provided to it by the CNN backbone. The quality of its temporal features is fundamentally tied to the quality of the spatial features present in the backbone model.

5.6 Contributions

The evaluations on the Block Stacking dataset empirically show that the D-ITR-L-driven approach to temporal feature modelling can overcome the two challenges being addressed in this dissertation: *duration invariance* and *video-scale features*. The block stacking dataset was developed to demonstrate this failing in contemporary works. While it is possible that alterations to the hyper-parameters or general architectures of the backbone CNN models

could improve their accuracy I demonstrate that in all cases the D-ITR-L temporal wrapper achieved improved accuracy over the baseline despite using the same set of spatially-learned features. This work has been submitted to the IEEE International Conference of Robotics and Automation 2022.

CHAPTER 6
TEMPORAL FEATURE FOR HUMAN ACTIVITY RECOGNITION

Probabilistic temporal model-based approaches to activity recognition achieved state-of-the-art performance on visually challenging domains through the use of hand-crafted features. I show that the same is possible when using the data-derived spatial features of a backbone convolutional neural network in conjunction with D-ITR-L. The code for this implementation is available at [68].

6.1 Related Work

Deep learning is the *de facto* approach to data-driven human activity recognition and has spawned a multitude of different approaches and derivatives that align with consistent design principles. I describe these general architectures and their approach to learning temporal features. I also discuss the state of contemporary benchmark video datasets and their limitations when validating an activity recognition model's ability to represent temporal information.

6.1.1 Pre-Deep Learning Activity Recognition

Approaches to activity recognition that do not leverage deep learning but which do use data driven spatial features do not generate representations of time. Methods such as visual bag of words [78, 79, 80, 81, 82], dense trajectories [83, 84], and histograms of oriented gradients [85, 86] are spatial feature extractors that have been leveraged for activity recognition. However, inference in these methods is often accomplished through the use of simple classification models that ignore time such as Support Vector Machines [6, 82], Nearest Neighbors [56], and

Integrated Methods	Interleaved Methods	Separate Methods
MML [93]	TPN [94]	Asyn-TF [64]
SlowFast [95]	TSM [11]	Multiscale TRN [12]
ECO [9]	STM [96]	ConvGRU [62]
I3D [65]	TrajectoryNet [97]	TCN [13]
2-stream [2]	R(2+1)D [98]	CNN-LSTM [20]

Table 6.1: Popular CNN Models for Video Inference.

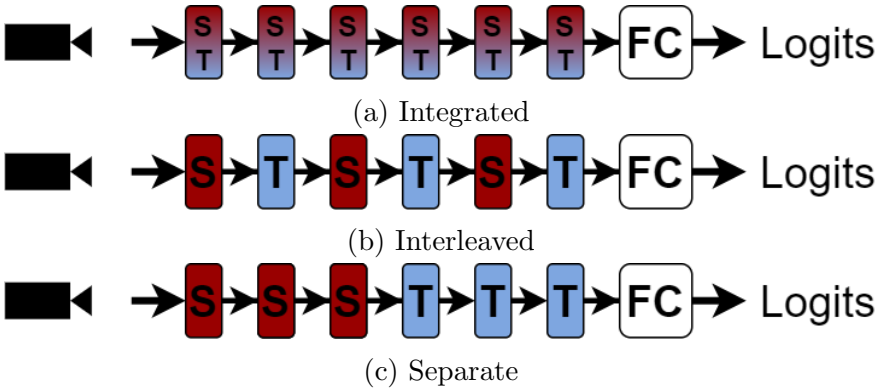
Random Forest [84]. More complex models have leveraged the use of space-time volumes [87, 88], space-time trajectories [89], and shape motion hybrid models [90, 91]. These approaches do not explicitly represent time but rather make inferences from the spatial expression of the features across a video. Many of the methods described in these works have since been combined into deep architectures to leverage the greater spatial feature detection properties of deep learning with earlier inference methods [79, 92].

6.1.2 Activity Recognition Post Deep Learning

Table 6.1 lists three broad categories of deep learning-based video inference architecture: Integrated, Interleaved, and Separate. These general structures address the possible approaches to combining spatial and temporal information that may be present in video. Included in the table are several popular CNN-driven deep learning architectures that represent these categories. Fig. 6.1 visualizes the general structure of each approach, which take video as input and generate logit values with the aid of a fully connected (FC) linear layer. Inside the network, inference is conducted using either spatial (red) or temporal (blue) feature representation layers. Integrated models combine spatial and temporal inference in a single approach.

Integrated Methods

Integrated models represent spatial and temporal features using a single structure. This is accomplished primarily through the use of 3D convolutional filters (ECO [9], SlowFast



(a) Integrated

(b) Interleaved

(c) Separate

Figure 6.1: General Structures of Temporal Representation in Deep Learning Models

[95], I3D [65]) or by the modifying the information in the input such that 2D convolutions perform 3D inference (i.e. convolutions of optical flow; 2-stream [2], MML [93]). These representations are rich, but they are also cumbersome. In order to generalize effectively they often require very large training datasets to overcome variances not just in time, but also simultaneously in space.

Interleaved Methods

Interleaved models alternate spatial and temporal feature learning. These models vary in their implementation. Structures such as R(2+1)D [98] take a straightforward approach and literally interweave 2D spatial convolution layers with 1D temporal convolution layers. Others, such as the Temporal Shift Module (TSM) [11], rely only on 2D convolutions, but in-between spatial convolutions they shift features along the temporal dimension to smooth the representations across time. These models are lightweight and typically generalize better to variances in time than integrated models [98].

Separate Methods

Separate methods learn temporal features after spatial features have been extracted. They are closest in structure to classical methods and rely on strong representation of spatial features to perform deeper temporal reasoning. Separate models frequently fall into one of

(a) UCF-101 [1] (b) HMDB-51[2] (c) Kinetics [3]

Figure 6.2: Examples from Benchmark Video Datasets

two sub-categories described in detail in Section 3.4: convolution-based approaches (TCN [13]) and recurrent neural network based models (LSTM [20] and GRU[62]). D-ITR-L follows the design principle of a separate method.

6.1.3 Datasets

Contemporary benchmark activity recognition datasets span the gamut in terms of their content and properties and, subsequently, can be used to assess a myriad of different challenges related to video inference. Many of the standard benchmark activity recognition datasets capture a broad selection of class labels and examples within those classes. These datasets are manicured to capture activities in a handful of frames, from a variety of view points, and a different array of sources. As a result, they challenge a model's ability to identify and generalize the spatial concepts that define these activities. For example, Fig. 6.2 captures several examples of classes within three popular benchmark video datasets: UCF-101 [1], HMDB-51[2], and Kinetics [3]. These examples are all distinguishable using a single frame as opposed to information regarding their temporal content. A specific example is the 'billiards' class from within the UCF-101 dataset (Fig. 6.2a) which can be easily distinguished

Dataset	Avg. Frames per Example
Jester	35.63 ± 2.31
Something-Something	45.59 ± 12.56
IKEA Furniture Assembly	62.29 ± 75.92
Crepe Sub-Actions	248.12 ± 194.07
Crepe Full-Recipes	3949.27 ± 703.21

Table 6.2: Average number of frames in Video Datasets

from other class labels based on the presence of an explicit spatial cue: a billiards table.

In recognition of the "spatial focus" of benchmark activity recognition datasets Cao *et al.* [7] encourage the use of temporally challenging datasets such as Jester [99] and Something-Something [100]. These datasets focus expressly on actions. Jester focuses on hand movements and gestures while Something-Something captures actions that manipulate objects in the environment. Despite Cao *et al.*'s assertions, these datasets still pale in comparison to the temporal content found in human demonstrations of high-level tasks. Jester and Something-Something are both short in duration (see Table. 6.2), limiting the potential for durational variance when compared to longer videos. The videos also focus on simple short-term transformations (either translation or deformations, Fig. 6.3 and Fig. 6.4) as opposed to long-term actions. Since the strength of D-ITR-L is its ability to capture video-scale features in a duration-invariant manner, it is counterintuitive to evaluate D-ITR-L on the vast majority of standard video benchmark datasets. Furthermore, the use of a bottleneck in the D-ITR-L architecture limits the expression of spatial features in these models and will likely result in poorer accuracy given the strong spatial focus of these benchmark video datasets.

Instead, I follow suit with earlier temporal feature learning work and focus on videos that have similar spatial content and which are long in duration (lasting more than 50 frames). Prior probabilistic temporal modelling approaches intent on capturing temporal features followed similar reasoning in their dataset collection [4, 33, 34, 55], but none of these datasets are available for public use. Consequently, I have selected two publicly available datasets that capture human-led demonstrations of tasks: IKEA furniture construction and

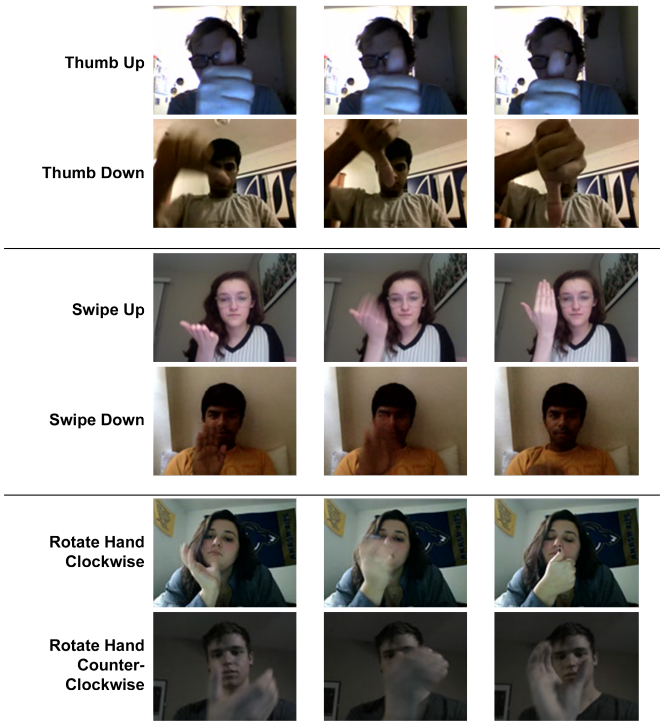


Figure 6.3: Example class labels from the Jester Video Dataset

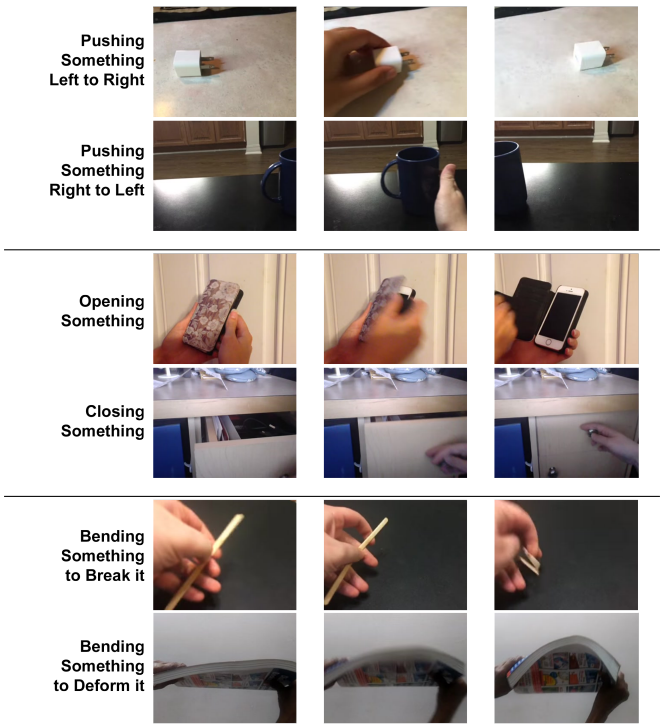


Figure 6.4: Example class labels from the Something-Something Video Dataset

Figure 6.5: Actions in the IKEA Furniture Assembly Dataset

Crepe recipe following. These datasets have so far been explored in the context of visual inference applications, not for activity recognition. I have modified them for use in this activity recognition.

6.2 Datasets for D-ITR-L Evaluation

The Crepe and IKEA datasets capture human-led demonstrations of two high-level multistep tasks: furniture construction and recipe following. This section discusses the content of these datasets and how I modified them to evaluate D-ITR-L's ability to capture temporal features and employ them for activity recognition.

IKEA: Furniture Assembly

The IKEA furniture assembly domain captures participants as they construct and deconstruct a table. The dataset is composed of 101 videos and captures 14 actors as they perform the necessary steps to complete the furniture assembly task (Fig. 6.5): picking up and screwing the four legs onto a table, flipping the table upright and then upside down, and then unscrewing and replacing the four legs of the table. The video has been timestamped to indicate when each of the actions is being performed and captures participants constructing

the table on the floor and on a raised work station. In the original data annotation, specific labels are attributed to the order in which the legs are added. This captures purely spatial features identifiable by the number of legs that remain secured in the table. I merge the individual labels into the less specific action classes of *attach leg* and *remove leg* to reduce the datasets reliance on spatial features and better focus on temporally-oriented actions. Rather than selecting a range of frames from the full video on which to do inference (the approach employed by the dataset's authors), I separate the videos into their individual actions which are labelled and used to train and validate my models. The video was recorded at 30fps and down sampled to 10fps.

Actions from the IKEA dataset are generally twice as long as videos in other benchmark activity recognition datasets (Table 6.2) and vary dramatically in size from between 8 frames up to 502 frames in length. The wide range in the duration of activities within this dataset is ideal for assessing the robustness of learned models when generalizing different durations of activities. Many of the observed actions capture video-scale features. Videos within the class labels of 'attach leg'/'detach leg' demonstrate long-term dependencies in the order that low-level actions are expressed. Attaching the leg to the table can be summarized in the following stages: 1) angle the table leg above the table and 2) insert the table leg into a hole in the base of the table. The detachment of the leg inverses these operations. Videos within the class labels of 'spin in'/'spin out' capture cyclical patterns as they generally require a human to re-position their hand repeatedly in order to secure or loosen the table leg.

Originally this dataset was used for human action forecasting by Han *et al.* [101] and then later for human pose forecasting using skeletal models by Toyer *et al.* [102]. Despite the author's claim that the dataset can be used for long-term video dependency understanding, little is done to evaluate the dataset from this perspective. Han *et al.*'s work is closest to my own and investigates the ability to predict actions using short clips of raw video lasting up to 50 frames. Their model uses an ensemble of LSTM models in coordination with a combined RGB/optical flow feature extractor and achieves an accuracy of 51.7% when predicting

Recipe Name	Task Order
Lemon Sugar	Stir, Pour, Spread, Flip, Pour, Sprinkle, Fold
Banana Chocolate	Stir, Pour, Spread, Cut, Flip, Transfer, Grate, Fold
Cheese Ham	Stir, Pour, Spread, Cut, Grate, Flip, Transfer, Fold
Cheese Ham Parsley	Stir, Pour, Spread, Cut, Grate, Flip, Transfer, Sprinkle, Fold
Goat Cheese Spinach	Stir, Pour, Spread, Cut, Flip, Transfer, Fold
Goat Cheese Spinach Nutmeg	Stir, Pour, Spread, Cut, Flip, Transfer, Sprinkle, Fold

Table 6.3: Crepe Recipes

subsequent actions. Action forecasting is a different task from action recognition in that inference is made on partial action executions rather than full activities. Subsequently, action forecasting does not need to represent the full action dynamics in its latent representation and does not need to consider variable duration as it tries to identify periods of transition between different atomic actions in an observation [103]. My work focuses on action recognition and should not be compared directly with the results given in Han *et al.* [101] since the format of the videos and the inference task are different.

Crepe Dataset

The Crepe dataset contains videos that capture volunteer chefs as they prepare 53 crepe meals according to 6 possible recipes. The recipes (listed in Table 6.3) are each composed of 9 potential sub-actions (Fig. 6.6). Each video captures three volunteers preparing crepes adjacent to one another and has frame level annotations to identify the sub-action, the recipe being performed, and a bounding box centered on the chef. The developers of the dataset, Lee *et al.*, used the crepe-making application as validation for an application on attention relocation and multi-person action recognition [84]. Their implementation used a hierarchy of machine learning methods: dense trajectories for spatial feature extraction and bounding box generation, random forest for action recognition, and a novel person tracking method. The authors evaluated their model at two scales: sub-action recognition and recipe recognition. Lee *et al.* report their per-label accuracy when evaluating the sub-actions within

Figure 6.6: Actions in the Crepe Dataset

Cut	Grate	Pour	Spread	Sprinkle	Stir	Transfer
70.6%	96.2%	64.8%	87.8%	5.6%	81.8%	63.6%

Table 6.4: Lee *et al.*'s Results on the Crepe Sub-action Dataset

the Crepe dataset, I have included their results in Table 6.4. Unfortunately, their reporting is incomplete (missing the 'flip' and 'fold' actions). Recipe recognition performance was measured as a value of Area Under the Curve and is reported to be 0.697. The code-book of dense trajectories used by Lee *et al.*'s implementation has not been made publicly available.

I evaluate my work in the same two-part approach as Lee *et al.*: I first investigate recognition of *sub-actions* and then I investigate recognition of *full-recipes*. One of the original applications of this dataset was multi-person activity recognition. Therefore, the dataset contains videos that capture 3 chefs preparing different recipes in parallel. To perform traditional activity recognition, I modify the original videos to focus on the activities of a single chef as they perform either a sub-action or as they complete a recipe. To accomplish this I use the frame-level bounding box information to isolate the region where each chef is

performing and then crop the video in duration to focus on the specific skill, again using the frame-level information to determine when a given task is being performed. When training to recognize these datasets, I exposed spatial feature extractors only to the sub-action dataset. Spatial features learned from the sub-action dataset were then used in the development and evaluation of temporal features on the same dataset and the full-recipe dataset.

The videos in the sub-action dataset were collected at 30fps and down-sampled to 10fps, but are still far longer in duration than any of the earlier mentioned benchmark video datasets (Table 6.2) averaging 248 frames in length. Many of these tasks exhibit durational variance with individuals displaying assorted levels of confidence and deliberation when performing different tasks. Several of the sub-actions also capture long-term dependencies and cyclical patterns. Actions such as 'cut' and 'grate' are defined by features representing both a quick repetitive motion and the slow deliberate movement of ingredients. These skills are most easily distinguished by the order in which these skills are portrayed. In 'cut', the ingredients are cut then placed into the pan, whereas in 'grate' the ingredients are collected and grated before being left in place for later use. Cyclical Actions are also represented by the 'sprinkle' and 'transfer' tasks. The 'sprinkle' task involves a singular "pick up and deposit" visual cue whereas the 'transfer' task typically occurs in two-repetitions of this pattern.

The full-recipe version of this dataset scales up these temporal challenges as they are composed of several sub-actions. The ordering of these actions is itself a type of long-term dependency and the long scale of these videos (the shortest video is 2630 frames and the longest is 5838 frames) captures not only the durational variance of the sub-actions themselves, but delays between when those sub-actions are expressed.

6.3 Experiments

6.3.1 Training

In this work I use the same four spatial feature extractors/CNN-backbones as described in Section 5.4: Temporal Shift Module, Wide ResNet, I3D, and VGG-16. Using their origi-

nal ImageNet learned features, these models are fine-tuned to recognize features on these new datasets: IKEA and Crepe Sub-Actions. Frames in both datasets are resized to a format compliant with the original architecture. Background subtraction and smoothing pre-processing steps are applied only on the Crepe sub-actions dataset. Pre-processing is not applied on the IKEA dataset as many of the actions displayed in the video are subtle with little bodily movement ('screw in' and 'screw out' for example). The lack of movement in these demonstrations would cause a background subtraction approach to occlude the critical information in these activities. While there is some difference in the spatial background characteristics of these videos, they are both inconsequential to recognizing different activities and are uniformly expressed across the classes in the dataset.

This work employs a bottleneck to constrain the complexity of the ITR graphs. A grid-search is used to identify the most effective bottleneck size from options of 16, 32, and 64. In all cases a bottleneck of 64 produced the best results for both datasets. These tasks are more complex than the block demonstration depicted in Chapter 5 and must be represented by a greater selection of spatial features in order to effectively generate a reasonable performance. It is possible that an even wider bottleneck could have generated better results, but each additional spatial event generates a pairwise increase in the number of temporal features captured within the ITR graph, a computational limitation given the D-ITR-L's current architecture and the specifications of the machine used to run these experiments (using 2 Titan X Pascal GPUs). Regardless, this bottleneck is applied only to the inference conducted by D-ITR-L and does not impact the performance of other temporal inference models.

All of the models in this work (both backbone spatial extractors and temporal feature learners) were trained for 50 epochs using an Adam optimizer with a learning rate of 1e-3 and a cross entropy loss function. In both datasets 70% of the examples for each class were used for training and the remaining 30% were used to evaluate the models. A batch size of 4 was used with the IKEA and Crepe Sub-Action datasets, but due to space limitations a batch size of 1 was used when training on the Full-Recipe version of the Crepe Dataset.

6.3.2 Results

The spatial feature extractors supply a fixed set of visual features expressed in videos of the evaluation datasets. I contrast D-ITR-L's temporal inference using these sets of features against other data-driven temporal inference approaches. This work investigates a linear, a recurrent (LSTM), and a convolution-based (TCN) model. These inference architectures were previously discussed in Section 5.5.

IKEA

When evaluated within the confines of the IKEA dataset, D-ITR-L outperforms all other data-driven temporal inference mechanisms (Table 6.5) regardless of the features extracted by the CNN-backbone. A brief summary of these results finds that the linear model performs the most poorly across all CNN backbones. This is expected given that the linear model lacks the ability to represent temporal information in an informed way. The convolution-based (TCN) and recurrent (LSTM) models perform similarly. These architectures are duration dependent and ill-suited to modelling long-term dependencies, two properties that the IKEA dataset captures. In all cases D-ITR-L shows an improvement over the baseline models, with margins of between 2.38% in I3D and 21.57% in Temporal Shift Module. The highest accuracy achieved on the IKEA dataset is 65.27% when D-ITR-L used the spatial features extracted from the VGG-16 backbone model. The performance of the TCN model and the D-ITR-L model on this set of spatial features are investigated more deeply in Fig. 6.7.

TCN (Fig. 6.7a) has the second highest results when analyzing the IKEA dataset using spatial features from the VGG-16 backbone and represents a baseline when comparing the performance of D-ITR-L (Fig. 6.7b). Both models are able to effectively represent the 'pick leg' class label. This action is particularly short and is not easily confused with other actions. The classes that showed the greatest improvement in quality are those previously established to contain video-scale features: 'attach leg'/ 'detach leg' and 'spin in'/'spin out'. These examples are often misconstrued in the TCN model, but are more easily delineated

Backbone	Linear	LSTM	TCN	D-ITR-L
I3D	53.20%	56.49%	56.49%	**58.87%**
Temporal Shift Module	24.86%	27.06%	25.59%	**48.63%**
Wide ResNet	17.92%	21.94%	21.39%	**42.60%**
VGG-16	43.69%	50.63%	51.00%	**65.27%**

Table 6.5: Accuracy on the IKEA Furniture Assembly Dataset

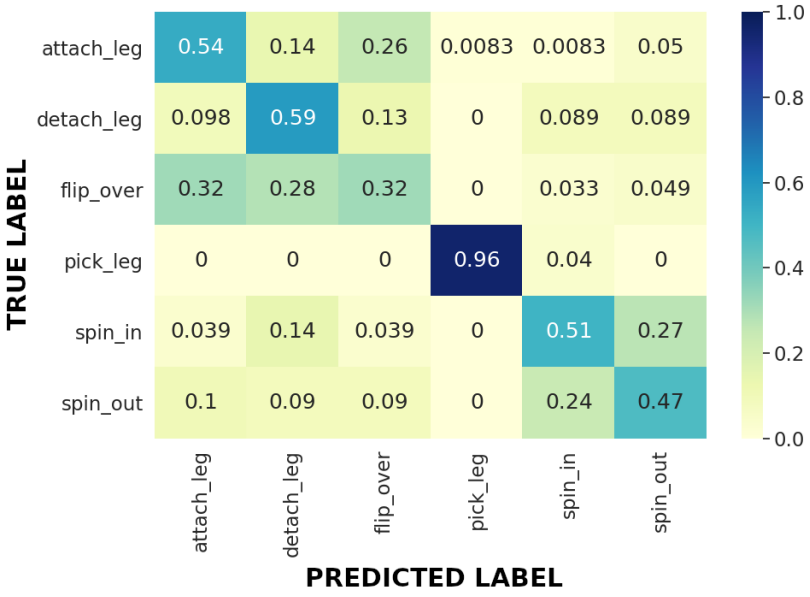
in the D-ITR-L approach. Recognition of the 'detach leg' action increased from 59% to 82% and while the accuracy of the 'attach leg' action did not improve, the number of times the action is mislabelled as 'detach leg' decreased from 14% to 3.3%. Interestingly, both models incorrectly label the 'flip over' actions using either the 'attach leg' or 'detach leg' class labels. This is likely due to difficulty in modelling the spatial features that represent the movement of the table. Individuals rotated the table in a variety of ways according to their preference and the class-label is broad capturing both flipping the table upright and upside down. The actions of the human required to flip the table parallel those of someone attaching and removing a table leg (i.e. squatting and standing) which could contribute to this inaccuracy. The 'spin in' and 'spin out' labels capture cyclical actions and improve from an accuracy of 51% and 47% respectively to 69% in both examples. Mislabelling of the 'spin in' and 'spin out' actions is reduced from 27% and 24% to 2.6% and 3.8% respectively.

Crepe

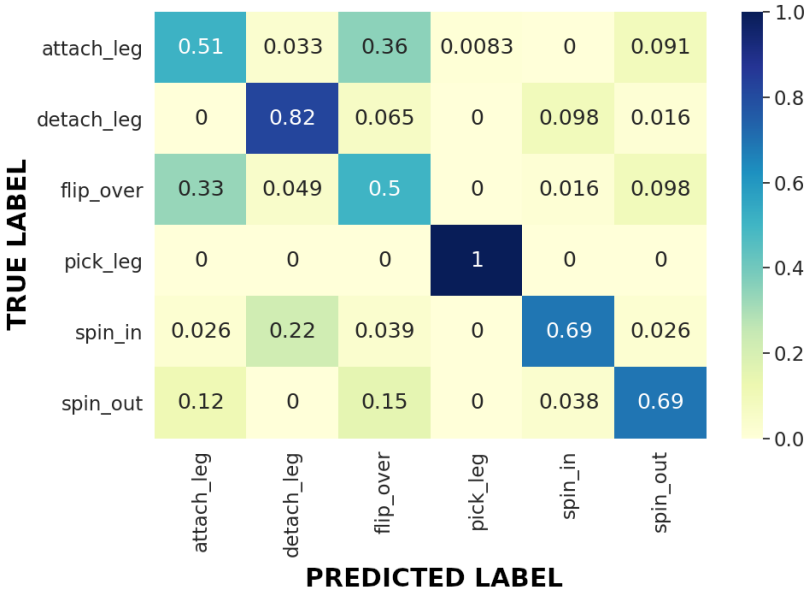
The Crepe dataset is evaluated at two scales: shorter sub-actions and the longer full recipes.

Crepe Sub-Action

I evaluate the ability of different models to represent the sub-actions expressed in the Crepe dataset. My investigation found that a D-ITR-L-based approach to representing temporal features from this dataset is superior to other baseline models (Table 6.6). The results of this experiment generally align with those of the IKEA dataset. Specifically that the linear model in most cases performs the worst, that inference using TCN and LSTM is comparable, and



(a) TCN Model



(b) D-ITR-L Model

Figure 6.7: Confusion Matrix of VGG-16 Backbone Results on the IKEA Dataset

Backbone	Linear	LSTM	TCN	D-ITR-L
I3D	61.67%	67.07%	70.64%	**73.05%**
Temporal Shift Module	57.49%	67.07%	53.29%	**72.46%**
Wide ResNet	29.98%	52.10%	61.67%	**70.06%**
VGG-16	58.68%	56.28%	59.28%	**66.46%**

Table 6.6: Accuracy on the Crepe Sub-Action Dataset

Backbone	Linear	LSTM	TCN	D-ITR-L
I3D	10.53%	10.53%	21.05%	**36.84%**
Temporal Shift Module	15.79%	15.79%	15.79%	**26.32%**
Wide ResNet	15.79%	15.79%	15.79%	**21.05%**
VGG-16	15.79%	21.05%	21.05%	**31.58%**

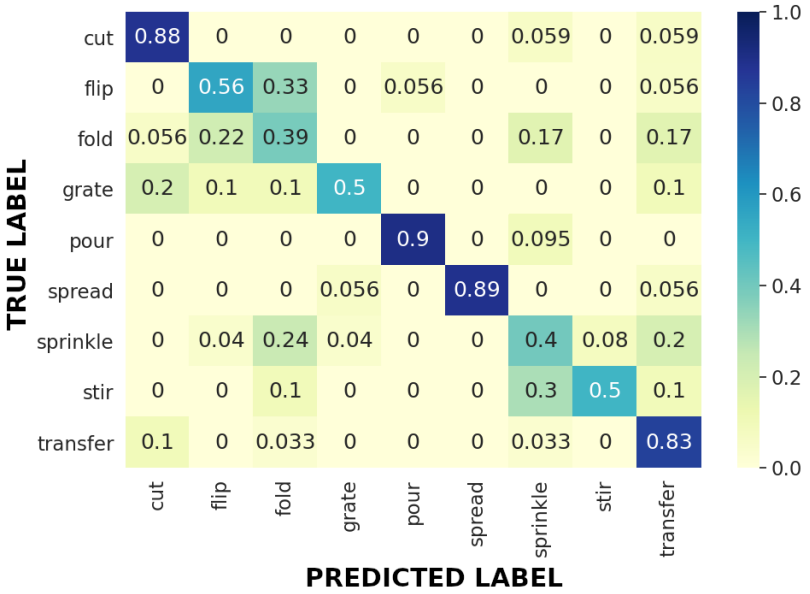
Table 6.7: Accuracy on the Crepe Full-Recipe Dataset

that D-ITR-L shows improvement over the nearest baseline video architecture regardless of the type of backbone model used (between 2.4% and 8.39% using the I3D and Wide ResNet backbone's respectively).

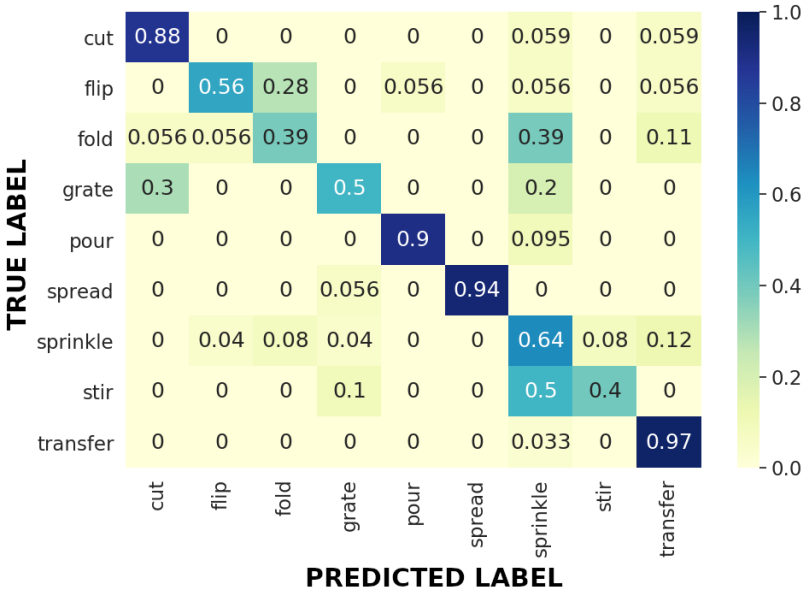
A confusion matrix capturing the per-action accuracy of the best performing model (D-ITR-L using an I3D backbone) and the nearest baseline (TCN) is presented in Fig. 6.8. The D-ITR-L shows a reduction in the mislabelling of several classes. With regards to the classes that capture video-scale features, there is an improvement of 24% in the 'sprinkle' and 14% in the 'transfer' classes which capture a cyclical activity. There is similarly a reduction of 8% when mislabelling the 'sprinkle' action as 'transfer'. The sub-actions of 'cut' and 'grate' capture long-term dependencies, but both models of this dataset show a high misclassification rate recognizing 'grate' actions as 'cut' (20% in TCN and 30% in D-ITR-L). An inspection of these videos suggests that mislabelled examples of 'grate' demonstrate low-level action ordering that better resembles that used in many of the 'cut' examples: fast rhythmic movement, followed by an action (to return the grating utensil to its original location).

Crepe Full Recipe

Finally, a study was done on the full crepe recipe dataset Again D-ITR-L outperformed earlier models, however, the accuracy of these models is more muted in comparison to the



(a) TCN Model



(b) D-ITR-L Model

Figure 6.8: Confusion Matrix of I3D Backbone Results on the Crepe Action Dataset

results on the IKEA and Crepe Sub-Action datasets. Many of the baseline models in this dataset fail to generalize to the data and achieve a random accuracy (15.79%) or over fit to a single class label that is under-represented in the validation dataset (10.53%). In contrast, D-ITR-L creates a more robust representation of the data and is able to recognize some of the demonstrated recipes, specifically those with the 'sprinkle' sub-action (see Table 6.3)). When using the I3D baseline, the 'sprinkle' sub-action demonstrate the greatest improvement in recognition when transitioning from a TCN-based inference model to a D-ITR-L-driven approach. Correctly identifying this action label reduces the potential recipes that the observation could belong to by half.

The large scale of the full-recipe dataset is formidable and demonstrates the current computational limits of the D-ITR-L architecture. ITR graphs from the full-recipe dataset number in the thousands of nodes and hundred of thousands of temporal relationships/edges (Table 6.8). With this increased complexity there is more of an inference challenge when identifying which of the many temporal relationships are discriminatory. These concerns can be addressed in one of two ways. The first demands an increase in computational power. By removing or relaxing the bottleneck, a greater variety of temporal features and a richer representation can be used to describe the contents of a video. This incurs a massive computational demand on these systems as the graphical representation experiences a pairwise increase in the number of edges with the inclusion of more spatial features. With further improvements to computational architectures, especially as they relate to GPU design, more data can be stored in memory allowing for more elaborate architectures. My work specifically used 2 Titan X GPUs and was restricted to using a bottleneck of size 64 to ensure all data could be correctly loaded and manipulated at run time. The second approach requires structural changes to the method by which GCN operates. A stochastic representation of ITRs, similar to the one used in [4], would allow for more flexibility in the representation of temporal features. This would require changes to the GCN architecture to facilitate inference on the new graphical structure. Specifically, the inference operation

Dataset	Backbone	Avg. Nodes	Avg. Edges
IKEA	VGG-16	56.21 ± 46.20	$1,509.06 \pm 1,671.23$
Crepe Sub-Action	VGG-16	353.24 ± 228.45	$10,588.73 \pm 8,475.32$
Crepe Full-Recipe	I3D	$2,975.09 \pm 725.13$	$159,286.47 \pm 61,722.03$

Table 6.8: ITR Graph Size in Best Performing D-ITR-L Models By Dataset

(Section 3.2.2) would need to not only focus on specific temporal relationships but would also need to consider partial expression of specific relationship. This change is complex and represents an entirely new research direction.

6.4 Conclusion

Effective temporal feature representation addresses concerns of *duration invariance* and *video-scale features*. However, the current benchmark video datasets do not challenge these properties and instead validate a model's ability to generalize to a selection of spatial features. Instead, video datasets that focus on human demonstration of sequential tasks can be used to challenge a model's ability to represent temporal features. Using the IKEA and Crepe datasets, I have validated D-ITR-L's ability to extrapolate temporal features from complex, real world observations of human activities. I have also summarized new research directions that would enhance the descriptive power of D-ITR-L on more challenging datasets.

6.5 Contributions

The experiments presented in this chapter demonstrate D-ITR-L's ability to extract video-scale features in a duration-invariant manner from two benchmakrk video-datasest. D-ITR-L outperforms all backbone models for all datasets. These results have been submitted to the IEEE International Conference of Robotics and Automation (ICRA) 2022.

CHAPTER 7

TEMPORALLY-GUIDED FEATURE SELECTION

Compared to a purely spatial alternative, temporal features are empirically better when conducting inference tasks from video data. Temporal features are defined not only by rich representation of temporal relationships, but also by the informed selection of spatial features upon which they are based. I perform a rigorous quantitative analysis to measure the contribution of a temporal information-guided approach to spatial feature selection. Features selected in this manner are used to train a policy learning architecture where they achieve better accuracy using fewer features and a smaller training dataset when compared to an approach that focuses on spatial feature selection. Following this study, I qualitatively analyze the properties of spatial features that are desirable when constructing a temporal model, providing insight into the representations encoded by a D-ITR-L-driven model.

7.1 Introduction

Spatial feature extractors, such as the CNN backbones used in Chapters 5 and 6 generate a monolithic semantic representation of the contents of a video. Many of these features identified in this manner are uninformative, redundant, or capture features that are unique to specific observations throughout the duration of videos [104, 105, 106]. The presence of these low quality features complicates learning in video recognition models by reducing their ability to generalize effectively and by inflating the number of computations that must be performed. These concerns are exacerbated in videos of visually similar environments such as demonstrations of high-level sequential tasks, where the discriminatory power of spatial

features is weakened.

Models that lack a representation of time do not consider the implications of order and duration. These models are likely to overlook specific, short-duration features in favor of features that capture more general contents and are expressed over long-durations as they are more heavily represented in the data. A temporally cognizant approach is capable of recognizing the importance of when and how long features are expressed. Video demonstrations are often described by a sequence of critical sub-goals rather than a single super-concept. For example, when pouring water from a pitcher you must engage in the following sequence of briefly expressed steps: pick-up a pitcher, tilt the pitcher, pour the water, and then return the pitcher. This chapter explores the importance of the spatial features that compose latent temporal representations, and then using those learned temporal structures to reciprocally improve and prune the set of spatial features used for inference tasks.

This work can be separated into two parts: 1) Section 7.2, a quantitative study investigating the significance of spatial features recognized by a temporal model (D-ITR-L) and 2) Section 7.3, a qualitative analysis of those features to identify consistent properties that are ideal in developing temporal features.

7.2 Temporally-Informed Spatial Feature Selection

I discuss the feature ranking approach used to determine the most significant features in a spatial (linear) and temporal (D-ITR-L) driven model. These features are then used to train a separate policy learning model.

7.2.1 Feature Selection and Appraisal

Spatially and temporally-ranked features are used to train a policy learning approach designed by a collaborator [107]. The policy learning model is not a contribution of this dissertation but is designed by a collaborating PhD student. The feature selection and appraisal process is denoted in Fig. 7.1. Videos are used to generate a set of fixed spatial

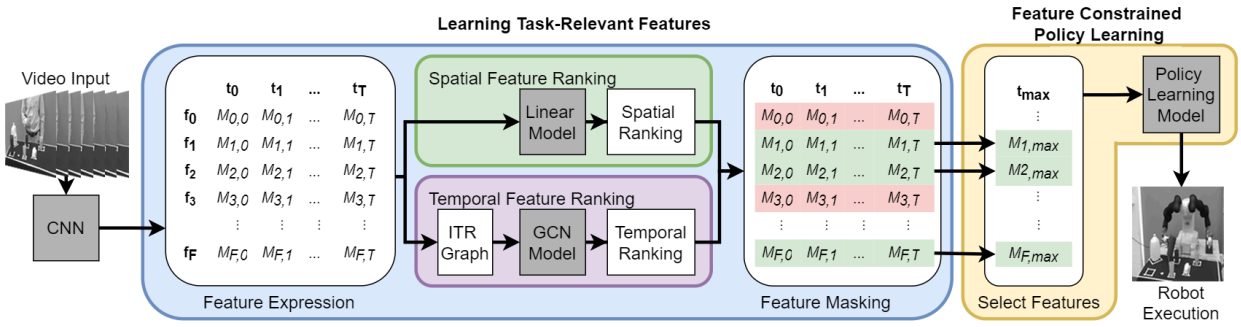


Figure 7.1: The Pipeline for Policy Learning in a Tea Making Task

features according to a backbone CNN. These features are used to train both a linear model and GCN using D-ITR-L-derived spatial features. These models are used to rank the importance of the input features. The most highly ranked features by each model are masked, and then used to train the aforementioned policy learning model.

Feature Ranking

I use a popular ranking approach, erasure ranking [106], to identify the most important features in two video-recognition architectures by class label ($c \in C$). Erasure ranking iterates through each of the input features ($f \in F$) and evaluates the trained model's performance when a specific feature is removed. The greater the impact on the model's logit values (V) the greater the significance that the given feature contributes towards the trained model. The ranking (R) of a feature is evaluated as a normalized sum over all examples (e) of a given class in the demonstration set ($\mathcal{D}$) can be calculated using

$$R(f, c) = \sum_{e \in \mathcal{D}} \frac{V(e, c, f) - V(e, c, \neg f)}{V(e, c, f)} \tag{7.1}$$

To rank the temporally-grounded visual features extracted using the temporal model (D-ITR-L) erasure search is applied over all of the feature labels that compose an ITR graph, removing each node (and all connected edges) that have the given feature label f. When ranking features that are purely spatial in nature, as is the case of a linear model, the model is trained using an IAD, which do not capture the explicit temporal relationships between

Algorithm 4 Using Feature Ranking to Guide Policy Learning

Input: $\mathcal{D}$, the set of video demonstrations
Input: Q_s, a CNN backbone model
Input: Q_t, a D-ITR-L temporal wrapper
Input: f_n, a limit on the number of features to use when training the model
Input: π, a policy learning model
Output: π_s, a policy learning model trained using highly-ranked spatial features
Output: π_t, a policy learning model trained using highly-ranked temporal features

1: $F_0, Q'_s \leftarrow Q_s(\mathcal{D})$ ▷ Train the CNN Backbone with $\mathcal{D}$
2: $Q'_t \leftarrow Q_t(\mathcal{D}, F_0)$ ▷ Train D-ITR-L with $\mathcal{D}$ and the spatial features F_0
3: $R_s \leftarrow ErasureRanking(Q'_s, \mathcal{D})$ ▷ Rank the Spatial model
4: $R_t \leftarrow ErasureRanking(Q'_t, \mathcal{D})$ ▷ Rank the Temporal model
5: $F_s \leftarrow sort(F_0, R_s)$ ▷ Sort the features according to high spatial model ranking
6: $F_t \leftarrow sort(F_0, R_t)$ ▷ Sort the features according to high temporal model ranking
7: $\pi_s \leftarrow \pi(F_s[: f_n])$ ▷ Mask the low-ranked features and train π
8: $\pi_t \leftarrow \pi(F_t[: f_n])$ ▷ Mask the low-ranked features and train π
9: **return** π_s, π_t

features, and investigate how the accuracy performs when each feature is removed. The goal of this method is to compare the efficacy of temporally-grounded and purely spatial features in policy learning.

Policy Model

The policy learning architecture used in this work (π) is a Behavioral Cloning approach designed and developed by a collaborator [107]. This policy learning architecture is their contribution and its implementation has been intentionally omitted from this thesis. A detailed description of the policy learner is available at [107] and the associated code for the experiments discussed in this chapter can be located at [108]. The policy learning model uses a set of user-defined features when representing the state and leverages the information theory principles of Feature Expectation Maximization and the Maximum Entropy Principle when representing and updating probability distributions over the model's actions. Feature Expectation Maximization is a popular method of ensuring that a policy learning model matches the observed distribution of states and actions [109, 110] and the Maximum Entropy Principle asserts that when provided with a choice of multiple distributions when representing

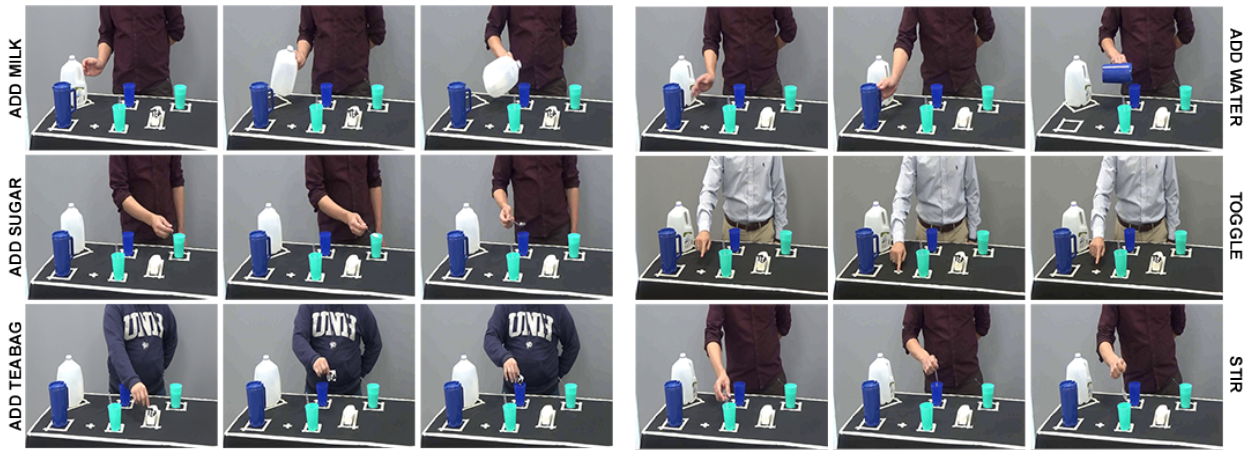


Figure 7.2: Actions from the Tea Making Dataset

a policy that the one with the greatest entropy should be chosen [111, 112]. The stochastic policy learned here is the solution to a convex optimization problem and therefore, unlike neural network policies, offers a convergence guarantee.

I train π using features that were either highly-ranked according to a spatial model (F_s) or a temporal model (F_t). Algorithm 4 describes the process used to conduct feature selection and policy training. Given a dataset of video demonstrations $\mathcal{D}$ I train a spatial model (a CNN Q_s; line 1). The spatial features identified by this model F_0 are then used to train a temporal model (D-ITR-L Q_t; line 2). I apply erasure ranking to the trained spatial (Q'_s) and temporal (Q'_t) models to ascertain the importance each model attributed to the spatial features R_s and R_t (lines 3 and 4). This importance is used to order the spatial features and the f_n most highly-ranked features are used to train π (lines 5-8). This process results in two versions of the policy learner: one in which the the policy is learned from features highly-ranked by a spatial model (π_s) and another where features are highly ranked by a temporal model (π_t). These models are evaluated in Section 7.2.4.

7.2.2 Dataset

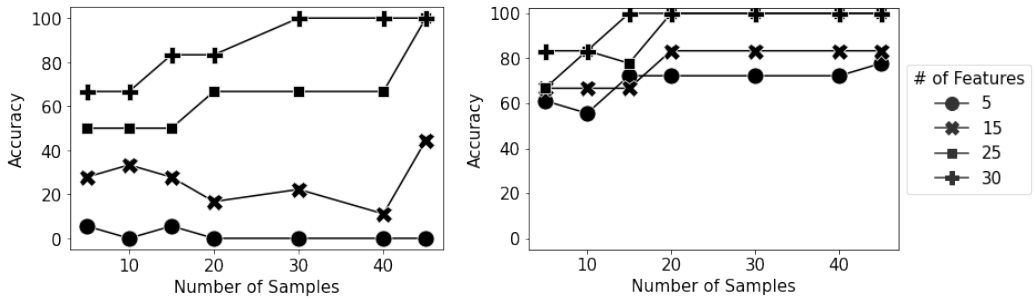
This work is evaluated on human-led demonstrations of a high-level sequential task: making tea (Fig. 7.2). Tea Making has been used as an example when describing temporal features

throughout this dissertation as it well-represents many of the challenges of inferring features from similar tasks in this domain. Four volunteers participated in an IRB-approved user-study to capture video demonstrations of the tea making process. Participants in the activity were asked to demonstrate the procedure they would follow for making a cup of tea using the following atomic actions: *toggle on/off the oven*, *add water*, *add sugar*, *add milk*, *add teabag*, and *stir*. The participants had access to a number of tea making tools (e.g. cups, teabag, and sugar) which were placed in fixed locations on a table. A total of 12 demonstrations were collected from each individual for a total of 48 videos which were subsequently segmented and labeled by hand. Videos were collected at 30fps and down-sampled to 10fps.

7.2.3 Training

Based on the high performance observed in earlier chapters (Chapter 5 and Chapter 6), I use VGG-16 [75] to capture the presence of spatial features as they are expressed in the atomic actions of the Tea Making dataset. VGG-16 (originally trained on ImageNet) is fine-tuned to recognize the action labels of videos in the tea-making dataset. Frames are reduced to 224×224 and subject to background subtraction before being fed into the CNN. The model is trained using an Adam optimizer with a learning rate of $1e-3$ over 50 epochs and with a batch size of 4. This model employs a bottleneck to reduce the number of spatial features from 2048 to 32 for use in inference of D-ITR-L. For consistency visualizing the contents of spatial features I used this bottleneck to evaluate the spatial model.

An R-GCN was again used to learn from the temporal features extracted by D-ITR-L. A simple linear layer was used to make class inference when training using an explicitly spatial model. In both cases, the networks were trained using the same optimizer (Adam), learning rate ($1e-3$), number of epochs (50), batch size (4), and loss function (Cross Entropy).



(a) Spatial features (b) Temporal features

Figure 7.3: Accuracy of the learned policy where the states are defined by the most highly-ranked features according to the two different ranking methodologies.

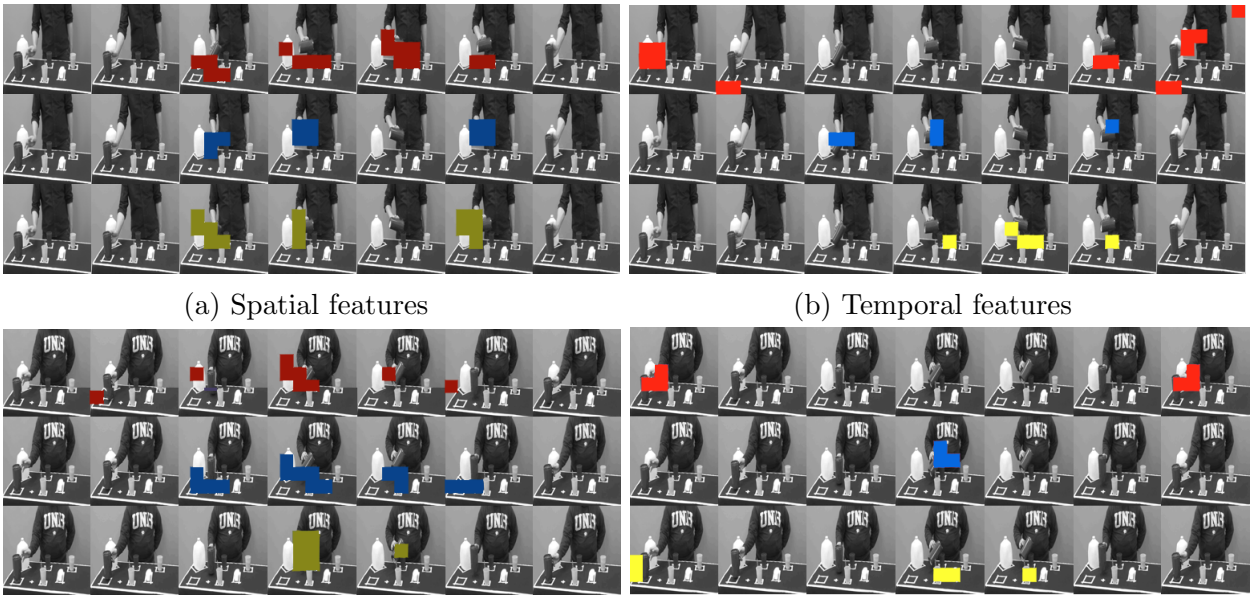
7.2.4 Results

I investigate the benefits that temporally-driven feature selection afford to state estimation in a policy learner. I compare increasing subsets of features that are highly-valued by either a temporal model and a spatial model. This investigation observed how policy performance decreases with a reduction in the set of available features used to describe the state of the videos in the dataset, and the number of samples per action label used to train the model.

Figs. 7.3a and 7.3b show findings from these experiments. For the same number of samples and task-features, the policy accuracy is higher when task-features are temporally-grounded as compared to when they are purely spatial in nature. The full accuracy can be achieved with $n = 30$ temporally grounded task-features collected only from 15 demonstrations whereas 30 demonstrations are needed to achieve the same accuracy with the same number of spatial task-features. Even with only 5 demonstrations, 5 temporally grounded task-features can achieve an accuracy of 60%. But more than 25 spatially-grounded features are required to achieve the same accuracy. These results demonstrate the potential role temporally grounded features play in learning good policies from complex video domains using only a handful of video demonstrations.

7.3 Feature Visualization

I perform a visualization-based analysis to try and discern the properties of the temporally-grounded features that lead to their greater significance. I contrast some of the highly ranked spatially-grounded and temporally-grounded features identified by the ranking system. I investigate two examples from four of the actions in the tea making task: 'add water' (Fig. 7.4), 'add sugar' (Fig. 7.5), 'stir' (Fig. 7.6), and 'add milk' (Fig. 7.7). Frames and features have been hand-selected for clarity. In all figures *a and b* and *c and d* capture the same frames from the same video. The background video has been depicted in greyscale in order to highlight the location of feature expression which are depicted in a variety of colors. Color denotes a particular feature as identified by each architecture. Brighter colors are features ranked highly by the temporal model and dimmer colors are features ranked highly by the spatial model. The same colors are used in examples of a singular action (i.e. the color and feature labels used in (a) and (c), and (b) and (d) are the same).



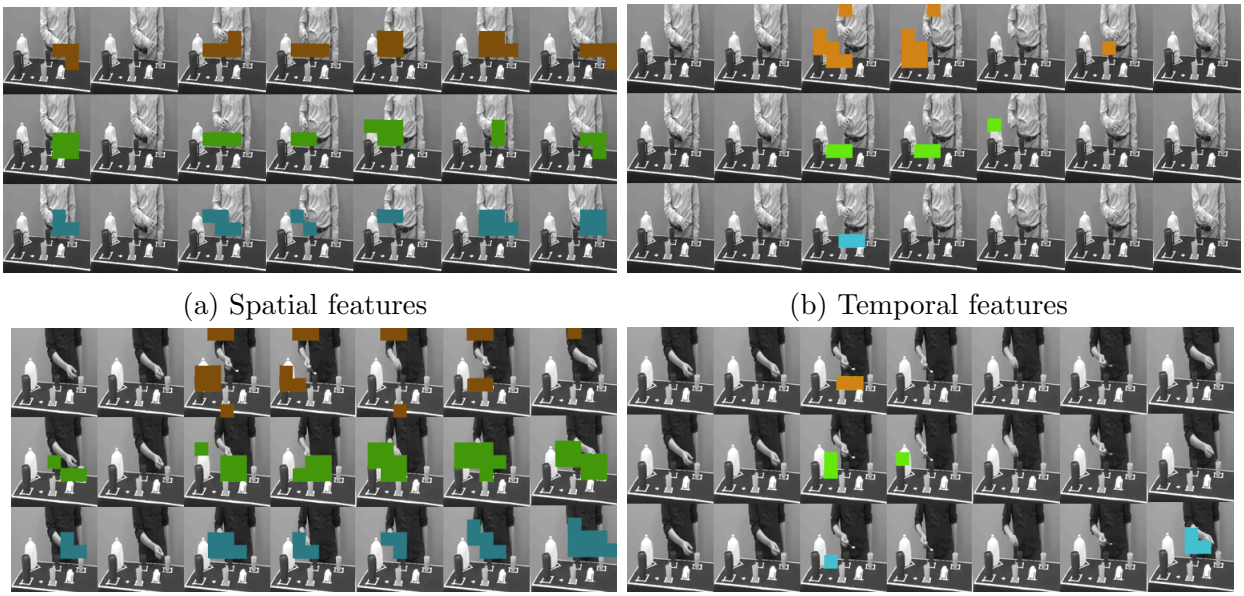
(a) Spatial features (b) Temporal features

(c) Spatial features (d) Temporal features

Figure 7.4: Visual analysis of the features in the context of the 'Add Water' action.

In most examples, I observed that spatially-grounded features capture similar spatial

features that are expressed in a majority of frames. When spatial features are not observed in a frame it was common for those features to be missing in the other highly-ranked spatially-based features. Temporally-grounded features captured more specific features related to the task being demonstrated. In the 'add water' task we see that the features capture picking up and returning the pitcher (red) and features for the pouring of the water (blue and yellow). The features in the 'add sugar' action capture those frames where the individual deposited the spoon into the mug (blue) and the reorientation (green) and return of the spoon (orange). The temporally-grounded features in the 'stir' action were intermittently expressed as the stir action was performed (lime and purple) capturing the cyclical pattern of the participants hand.



(a) Spatial features

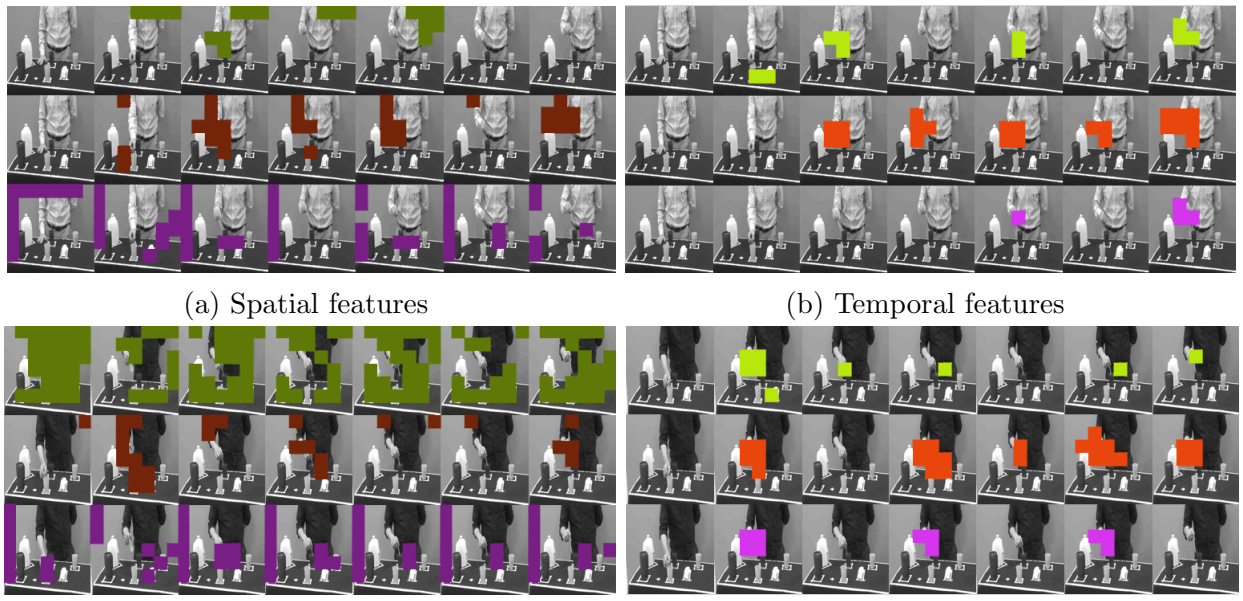
(b) Temporal features

(c) Spatial features

(d) Temporal features

Figure 7.5: Visual analysis of the features in the context of the 'Add Sugar' action.

However, it is not always the case that features in the spatial and temporal rankings are different. In the case of the 'add milk' action a spatial feature is ranked highly by both methods (blue) as it captures a significant aspect of the task: rotating the milk carton. But the supporting features in each method differ. The spatially-grounded features (dark orange, dark yellow) capture ambient features about the participants general location. In contrast,



(a) Spatial features (b) Temporal features

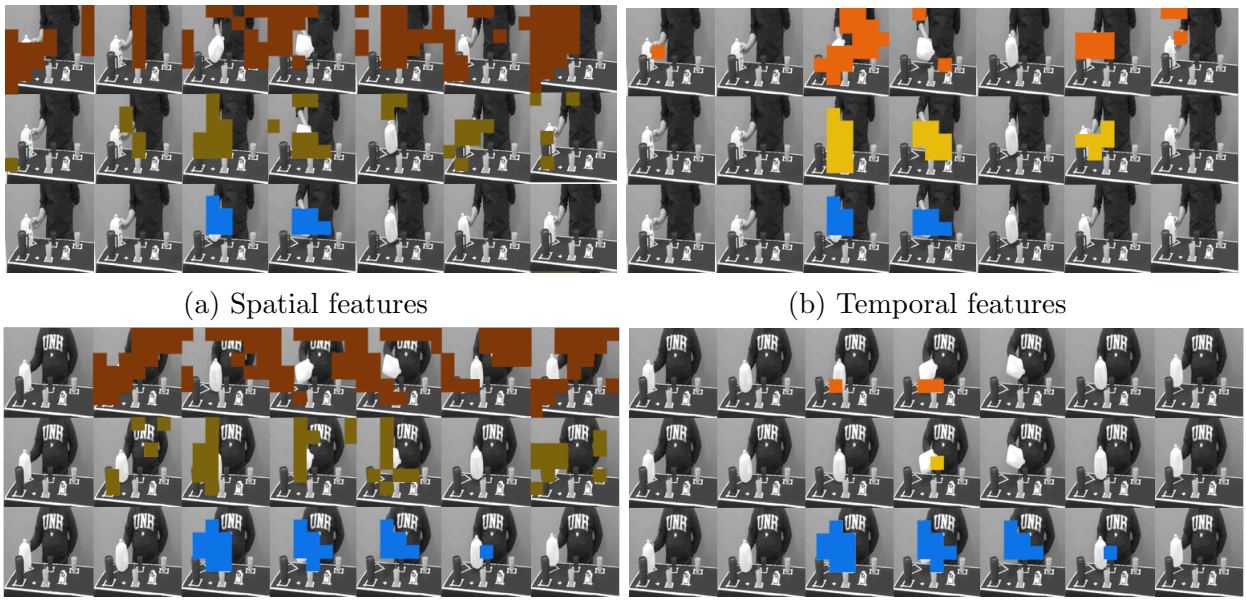
(c) Spatial features (d) Temporal features

Figure 7.6: Visual analysis of the features in the context of the 'Stir' action.

the temporally-grounded features focus specifically on the milk task including picking up and returning the milk carton (orange, yellow).

7.4 Conclusion

Temporal features are defined both in the unique representation given to them by the temporal inference approach used to model them and by the spatial features from which they are developed. A quantitative analysis demonstrated that at several different scales the features selected by a temporally-cognizant model were superior to those selected by a standard linear model. In the experiments described, these features better represent the state of the tea making application than spatial features allowing for greater accuracy when using fewer training examples. A qualitative analysis revealed these features to be scattered throughout the video capturing critical moments of the tea making activity. In contrast, features ranked by the linear model were redundant and ambient, capturing concepts expressed across the full length of the video.



(a) Spatial features (b) Temporal features

(c) Spatial features (d) Temporal features

Figure 7.7: Visual analysis of the features in the context of the 'Add Milk' action.

7.5 Contributions

This work was a collaborative project with another member of the Cognitive Assistive Robotics Lab and has been submitted to the IEEE International Conference of Robotics and Automation (ICRA) 2022. The results of this investigation highlight the role that temporal features can play in feature selection and vice versa. Furthermore, I investigate the properties of spatial features that make them good fits for use with the D-ITR-L representation.

CHAPTER 8
CONCLUSION

My dissertation has proposed a novel temporal representation that is built upon the state-of-the-art data-driven deep learning approaches to video recognition. This work directly addressed challenges involved with the processing of video data that posses long durations and similar visual content. Such videos are prolific in human-led demonstrations of high-level skills. The temporal representations of probabilistic temporal models have demonstrated proficiency in modelling these activities, however, they do so using hand-picked spatial features. User selection of spatial features assumes expert knowledge of the domain and limits the future application of learned models. Conversely, the most effective data driven approaches towards learning spatial features, Convolutional Neural Networks, rely on weaker representations of time when conducting video inference.

CNN-based video recognition architectures have largely been evaluated on spatially focused video benchmarks that do not adequately challenge their ability to represent temporal features. Given the more difficult video data investigated in this work, they must overcome two challenges related to this domain: variance in the duration of activities and representation of temporal features that span the entire length of a video. Standard CNN-driven temporal modelling approaches, such as Recurrent Neural Networks and convolution-over-time, represent features in a duration dependent and hierarchical fashion. Duration dependent architecture are not designed to address variances in duration in any meaningful manner. And the use of hierarchical design scales poorly requiring unreasonably deep networks to capture them in their entirety. These challenges can be addressed by employing the repre-

sentation of time established in probabilistic temporal models. These works used abstract terms to describe spatial events and temporal relationships in terms not related to a fixed duration. They also leveraged graphical structures to relate distant concepts from across an entire video.

My dissertation has described D-ITR-L, a temporal wrapper that presents CNN-learned spatial features using the temporal representation of probabilistic temporal models. I used this inference to conduct effective learning in challenging video domains. Validation of the claims has been accomplished in three studies. They demonstrated: 1) that D-ITR-L generated features are both robust to variations in duration and can model temporal features expressed across the length of a video, 2) that challenging video domains possessing long duration and similar visual contents are described most effectively using the aforementioned representation, and 3) that temporal feature information can be used retroactively to improve upon the spatial representation of a model. I supported these points using a combination of policy learning and action recognition problem settings.

8.1 Future Directions

Following the success of D-ITR-L, it is important to address future directions that may be taken with this work.

8.1.1 Enhanced Temporal Representation

One limitation of D-ITR-L is the use of spatial feature bottlenecks to address the quadratic growth of the ITR graph. Addressed in Section 6.3.2, advancements in computational capabilities or the extensions to the design of Graph Convolutional Networks could allow for richer representation of videos. These would allow for the removal or reduction in the use of bottlenecks and the inclusion of stochasticity into the temporal representation respectively. Either approach would improve upon the ability of these models to represent the contents of a video.

8.1.2 End-to-end Learning

Finally, I investigated the properties of those spatial features identified as useful to the representation generated by D-ITR-L. Features that were specific to sub-actions within a video generated the informed representation of the task being demonstrated. A natural extension of my work would be to include D-ITR-L into an end-to-end architecture. This would refine the spatial feature extractor symbiotically improving the strength of the spatial and temporal representations used by the model.

8.2 Contributions

Deep Interval Temporal Relationship Learner (D-ITR-L) provides a novel representation of temporal features for video recognition tasks. These features leverage the data driven spatial feature extraction abilities of Convolutional Neural Networks and are useful when modelling videos with long durations and similar spatial content. The D-ITR-L-derived representation is both robust to variations in the duration of activities and can scale to capture critical information expressed across the entire-length of a video. D-ITR-L has been neither designed nor evaluated on video observations that are defined by either an explicit duration or a strong spatial properties. Subsequently, it should be anticipated that this approach may preform poorly under such circumstances. This work has been the focus of several submitted (ICRA 2022 and CoRL 2021) and published works (RO-MAN 2017 [15] and HRI 2018 [16]) and has galvanized research into temporal reasoning within our lab that has influenced a number of collaborative works (submitted to CoRL 2021 and published in RO-MAN 2019 [31] and ICRA 2019 [14]).

LIST OF REFERENCES

[1] Khurram Soomro, Amir Roshan Zamir, and Mubarak Shah. Ucf101: A dataset of 101 human actions classes from videos in the wild. *arXiv preprint arXiv:1212.0402*, 2012.

[2] Karen Simonyan and Andrew Zisserman. Two-stream convolutional networks for action recognition in videos. In *Advances in neural information processing systems*, pages 568–576, 2014.

[3] Will Kay, Joao Carreira, Karen Simonyan, Brian Zhang, Chloe Hillier, Sudheendra Vijayanarasimhan, Fabio Viola, Tim Green, Trevor Back, Paul Natsev, et al. The kinetics human action video dataset. *arXiv preprint arXiv:1705.06950*, 2017.

[4] Yongmian Zhang, Yifan Zhang, Eran Swears, Natalia Larios, Ziheng Wang, and Qiang Ji. Modeling temporal interactions with interval temporal bayesian networks for complex activity recognition. *IEEE transactions on pattern analysis and machine intelligence*, 35(10):2468–2483, 2013.

[5] Vladimir Ryabov and André Trudel. Probabilistic temporal interval networks. In *Proceedings. 11th International Symposium on Temporal Representation and Reasoning, 2004. TIME 2004.*, pages 64–67. IEEE, 2004.

[6] Djamila Romaissa Beddiar, Brahim Nini, Mohammad Sabokrou, and Abdenour Hadid. Vision-based human activity recognition: a survey. *Multimedia Tools and Applications*, 79(41):30509–30555, 2020.

[7] Kaidi Cao, Jingwei Ji, Zhangjie Cao, Chien-Yi Chang, and Juan Carlos Niebles. Few-shot video classification via temporal alignment. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 10618–10627, 2020.

[8] Jingran Zhang, Fumin Shen, Xing Xu, and Heng Tao Shen. Temporal reasoning graph for activity recognition. *IEEE Transactions on Image Processing*, 29:5491–5506, 2020.

[9] Mohammadreza Zolfaghari, Kamaljeet Singh, and Thomas Brox. Eco: Efficient convolutional network for online video understanding. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 695–712, 2018.

[10] Bharat Singh and Larry S Davis. An analysis of scale invariance in object detection snip. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 3578–3587, 2018.

[11] Ji Lin, Chuang Gan, and Song Han. Tsm: Temporal shift module for efficient video understanding. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 7083–7093, 2019.

[12] Bolei Zhou, Alex Andonian, Aude Oliva, and Antonio Torralba. Temporal relational reasoning in videos. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 803–818, 2018.

[13] Colin Lea, Michael D Flynn, Rene Vidal, Austin Reiter, and Gregory D Hager. Temporal convolutional networks for action segmentation and detection. In *proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 156–165, 2017.

[14] Estuardo Carpio, Madison Clark-Turner, Paul Gesel, and Momotaz Begum. Leveraging temporal reasoning for policy selection in learning from demonstration. In *2019 International Conference on Robotics and Automation (ICRA)*, pages 7798–7804. IEEE, 2019.

[15] Madison Clark-Turner and Momotaz Begum. Deep recurrent q-learning of behavioral intervention delivery by a robot from demonstration data. In *2017 26th IEEE International Symposium on Robot and Human Interactive Communication (RO-MAN)*, pages 1024–1029. IEEE, 2017.

[16] Madison Clark-Turner and Momotaz Begum. Deep reinforcement learning of abstract reasoning from demonstrations. In *Proceedings of the 2018 ACM/IEEE International Conference on Human-Robot Interaction*, pages 160–168, 2018.

[17] Kalesha Bullard, Baris Akgun, Sonia Chernova, and Andrea L Thomaz. Grounding action parameters from demonstration. In *Robot and Human Interactive Communication (RO-MAN), 2016 25th IEEE International Symposium on*, pages 253–260. IEEE, 2016.

[18] Staffan Ekvall and Danica Kragic. Robot learning from demonstration: a task-level planning approach. *International Journal of Advanced Robotic Systems*, 5(3):33, 2008.

[19] Richard Cubek, Wolfgang Ertel, and Günther Palm. High-level learning from demonstration with conceptual spaces and subspace clustering. In *Robotics and Automation (ICRA), 2015 IEEE International Conference on*, pages 2592–2597. IEEE, 2015.

[20] Jin Wang, Liang-Chih Yu, K Robert Lai, and Xuejie Zhang. Dimensional sentiment analysis using a regional CNN-LSTM model. In *Proceedings of the 54th annual meeting of the association for computational linguistics (volume 2: Short papers)*, pages 225–230, 2016.

[21] Volodymyr Mnih, Koray Kavukcuoglu, David Silver, Andrei A Rusu, Joel Veness, Marc G Bellemare, Alex Graves, Martin Riedmiller, Andreas K Fidjeland, Georg Ostrovski, et al. Human-level control through deep reinforcement learning. *Nature*, 518(7540):529–533, 2015.

[22] Todd Hester, Matej Vecerik, Olivier Pietquin, Marc Lanctot, Tom Schaul, Bilal Piot, Dan Horgan, John Quan, Andrew Sendonaris, Ian Osband, et al. Deep q-learning from demonstrations. In *Thirty-Second AAAI Conference on Artificial Intelligence*, 2018.

[23] Madison Clark-Turner. Deep reinforcement abstract lfd, 2017. https://github.com/AssistiveRoboticsUNH/deep_reinforcement_abstract_lfd.

[24] Christoph Feichtenhofer, Axel Pinz, and Andrew Zisserman. Convolutional two-stream network fusion for video action recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 1933–1941, 2016.

[25] Li-Chia Yang, Szu-Yu Chou, Jen-Yu Liu, Yi-Hsuan Yang, and Yi-An Chen. Revisiting the problem of audio-based hit song prediction using convolutional neural networks. *arXiv preprint arXiv:1704.01280*, 2017.

[26] Sinno Jialin Pan and Qiang Yang. A survey on transfer learning. *IEEE Transactions on knowledge and data engineering*, 22(10):1345–1359, 2010.

[27] Christian Szegedy, Sergey Ioffe, Vincent Vanhoucke, and Alexander A Alemi. Inception-v4, inception-resnet and the impact of residual connections on learning. In *AAAI*, pages 4278–4284, 2017.

[28] Fei-Yue Wang, Jun Jason Zhang, Xinhu Zheng, Xiao Wang, Yong Yuan, Xiaoxiao Dai, Jie Zhang, and Liuqing Yang. Where does alphago go: From church-turing thesis to alphago thesis and beyond. *IEEE/CAA Journal of Automatica Sinica*, 3(2):113–120, 2016.

[29] Xindi Shang, Donglin Di, Junbin Xiao, Yu Cao, Xun Yang, and Tat-Seng Chua. Annotating objects and relations in user-generated videos. In *Proceedings of the 2019 on International Conference on Multimedia Retrieval*, pages 279–287, 2019.

[30] Mengmeng Xu, Chen Zhao, David S Rojas, Ali Thabet, and Bernard Ghanem. G-tad: Sub-graph localization for temporal action detection. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 10156–10165, 2020.

[31] Estuardo Carpio, Madison Clark-Turner, and Momotaz Begum. Learning sequential human-robot interaction tasks from demonstrations: The role of temporal reasoning. In *2019 28th IEEE International Conference on Robot and Human Interactive Communication (RO-MAN)*, pages 1–8. IEEE, 2019.

[32] Eran Swears, Anthony Hoogs, Qiang Ji, and Kim Boyer. Complex activity recognition using granger constrained dbn (gcdbn) in sports and surveillance video. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 788–795, 2014.

[33] Zhi Zeng and Qiang Ji. Knowledge based activity recognition with dynamic bayesian network. In *European conference on computer vision*, pages 532–546. Springer, 2010.

[34] Yongqiang Li, S Mohammad Mavadati, Mohammad H Mahoor, Yongping Zhao, and Qiang Ji. Measuring the intensity of spontaneous facial action units with dynamic bayesian network. *Pattern Recognition*, 48(11):3417–3427, 2015.

[35] Li Liu, Li Cheng, Ye Liu, Yongpo Jia, and David S Rosenblum. Recognizing complex activities by a probabilistic interval-based model. In *Thirtieth AAAI conference on artificial intelligence*, 2016.

[36] James F Allen and George Ferguson. Actions and events in interval temporal logic. *Journal of logic and computation*, 4(5):531–579, 1994.

[37] François Chollet. Xception: Deep learning with depthwise separable convolutions. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 1251–1258, 2017.

[38] Thomas N Kipf and Max Welling. Semi-supervised classification with graph convolutional networks. *arXiv preprint arXiv:1609.02907*, 2016.

[39] Michael Schlichtkrull, Thomas N Kipf, Peter Bloem, Rianne Van Den Berg, Ivan Titov, and Max Welling. Modeling relational data with graph convolutional networks. In *European Semantic Web Conference*, pages 593–607. Springer, 2018.

[40] Eugene Santos Jr and Joel D Young. Probabilistic temporal networks: A unified framework for reasoning with time and uncertainty. *International Journal of Approximate Reasoning*, 20(3):263–291, 1999.

[41] Alexander Artikis, Evangelos Makris, and Georgios Paliouras. A probabilistic interval-based event calculus for activity recognition. *Annals of Mathematics and Artificial Intelligence*, 89(1):29–52, 2021.

[42] Shaogang Gong and Tao Xiang. Recognition of group activities using dynamic probabilistic networks. In *Proceedings ninth IEEE international conference on computer vision*, pages 742–749. IEEE, 2003.

[43] Thi V Duong, Hung Hai Bui, Dinh Q Phung, and Svetha Venkatesh. Activity recognition and abnormality detection with the switching hidden semi-markov model. In *2005 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'05)*, volume 1, pages 838–845. IEEE, 2005.

[44] Matthew Brand, Nuria Oliver, and Alex Pentland. Coupled hidden markov models for complex action recognition. In *Proceedings of IEEE computer society conference on computer vision and pattern recognition*, pages 994–999. IEEE, 1997.

[45] Yifan Shi, Yan Huang, David Minnen, Aaron Bobick, and Irfan Essa. Propagation networks for recognition of partially ordered sequential action. In *Proceedings of the 2004 IEEE Computer Society Conference on Computer Vision and Pattern Recognition, 2004. CVPR 2004.*, volume 2, pages II–II. IEEE, 2004.

[46] Michael H Bohlen, Renato Busatto, and Christian S Jensen. Point-versus interval-based temporal data models. In *Proceedings 14th international conference on data engineering*, pages 192–200. IEEE, 1998.

[47] Massimiliano Albanese, Rama Chellappa, Vincenzo Moscato, Antonio Picariello, VS Subrahmanian, Pavan Turaga, and Octavian Udrea. A constrained probabilistic petri net framework for human activity detection in video. *IEEE Transactions on Multimedia*, 10(8):1429–1443, 2008.

[48] Yrvann Emzivat, Benoit Delahaye, Didier Lime, and Olivier H Roux. Probabilistic time petri nets. In *International Conference on Applications and Theory of Petri Nets and Concurrency*, pages 261–280. Springer, 2016.

[49] Raffay Hamid, Siddhartha Maddi, Aaron Bobick, and Irfan Essa. Structure from statistics-unsupervised activity analysis using suffix trees. In *2007 IEEE 11th International Conference on Computer Vision*, pages 1–8. IEEE, 2007.

[50] Imen Jegham, Anouar Ben Khalifa, Ihsen Alouani, and Mohamed Ali Mahjoub. Vision-based human action recognition: An overview and real world challenges. *Forensic Science International: Digital Investigation*, 32:200901, 2020.

[51] Daniele Liciotti, Michele Bernardini, Luca Romeo, and Emanuele Frontoni. A sequential deep learning application for recognising human activities in smart homes. *Neurocomputing*, 396:501–513, 2020.

[52] Li Liu, Shu Wang, Bin Hu, Qingyu Qiong, Junhao Wen, and David S Rosenblum. Learning structures of interval-based bayesian networks in probabilistic generative model for human complex activity recognition. *Pattern Recognition*, 81:545–561, 2018.

[53] Li Liu, Shu Wang, Guoxin Su, Zi-Gang Huang, and Ming Liu. Towards complex activity recognition using a bayesian network-based probabilistic generative framework. *Pattern Recognition*, 68:295–309, 2017.

[54] Kyle Lund, Sam Dietrich, Scott Chow, and James Boerkoel. Robust execution of probabilistic temporal plans. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 31, 2017.

[55] Behjat Siddiquie, Yaser Yacoob, and Larry Davis. Recognizing plays in american football videos. *University of Maryland, Tech. Rep*, 111, 2009.

[56] Junwu Weng, Chaoqun Weng, and Junsong Yuan. Spatio-temporal naive-bayes nearest-neighbor (st-nbnn) for skeleton-based action recognition. In *Proceedings of the IEEE Conference on computer vision and pattern recognition*, pages 4171–4180, 2017.

[57] Sijie Song, Cuiling Lan, Junliang Xing, Wenjun Zeng, and Jiaying Liu. An end-to-end spatio-temporal attention model for human action recognition from skeleton data. In *Proceedings of the AAAI conference on artificial intelligence*, volume 31, 2017.

[58] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. Imagenet classification with deep convolutional neural networks. *Advances in neural information processing systems*, 25:1097–1105, 2012.

[59] Klaus Greff, Rupesh K Srivastava, Jan Koutník, Bas R Steunebrink, and Jürgen Schmidhuber. LSTM: A search space odyssey. *IEEE transactions on neural networks and learning systems*, 28(10):2222–2232, 2016.

[60] Shi Xingjian, Zhourong Chen, Hao Wang, Dit-Yan Yeung, Wai-Kin Wong, and Wangchun Woo. Convolutional LSTM network: A machine learning approach for precipitation nowcasting. In *Advances in neural information processing systems*, pages 802–810, 2015.

[61] Kyunghyun Cho, Bart Van Merriënboer, Caglar Gulcehre, Dzmitry Bahdanau, Fethi Bougares, Holger Schwenk, and Yoshua Bengio. Learning phrase representations using rnn encoder-decoder for statistical machine translation. *arXiv preprint arXiv:1406.1078*, 2014.

[62] Debidatta Dwibedi, Pierre Sermanet, and Jonathan Tompson. Temporal reasoning in videos using convolutional gated recurrent units. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops*, pages 1111–1116, 2018.

[63] Serkan Kiranyaz, Onur Avci, Osama Abdeljaber, Turker Ince, Moncef Gabbouj, and Daniel J Inman. 1d convolutional neural networks and applications: A survey. *Mechanical systems and signal processing*, 151:107398, 2021.

[64] Gunnar A Sigurdsson, Santosh Divvala, Ali Farhadi, and Abhinav Gupta. Asynchronous temporal fields for action recognition. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 585–594, 2017.

[65] Joao Carreira and Andrew Zisserman. Quo vadis, action recognition? a new model and the kinetics dataset. In *proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 6299–6308, 2017.

[66] Christian Szegedy, Wei Liu, Yangqing Jia, Pierre Sermanet, Scott Reed, Dragomir Anguelov, Dumitru Erhan, Vincent Vanhoucke, and Andrew Rabinovich. Going deeper with convolutions. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 1–9, 2015.

[67] Lichen Zhou, Chuang Zhang, and Ming Wu. D-linknet: Linknet with pretrained encoder and dilated convolution for high resolution satellite imagery road extraction. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops*, pages 182–186, 2018.

[68] Madison Clark-Turner. Temporal feature lfd, 2021. https://github.com/AssistiveRoboticsUNH/temporal_feature_lfd.

[69] Ashvin Nair, Dian Chen, Pulkit Agrawal, Phillip Isola, Pieter Abbeel, Jitendra Malik, and Sergey Levine. Combining self-supervised learning and imitation for vision-based rope manipulation. In *2017 IEEE international conference on robotics and automation (ICRA)*, pages 2146–2153. IEEE, 2017.

[70] Ilija Radosavovic, Xiaolong Wang, Lerrel Pinto, and Jitendra Malik. State-only imitation learning for dexterous manipulation. *arXiv preprint arXiv:2004.04650*, 2020.

[71] Radoslav Skoviera, Karla Stepanova, Michael Tesar, Gabriela Sejnova, Jiri Sedlar, Michal Vavrecka, Robert Babuska, and Josef Sivic. Teaching robots to imitate a human with no on-teacher sensors. what are the key challenges? *arXiv preprint arXiv:1901.08335*, 2019.

[72] Jonathan Ho, Jayesh Gupta, and Stefano Ermon. Model-free imitation learning with policy optimization. In *International Conference on Machine Learning*, pages 2760–2769. PMLR, 2016.

[73] Faraz Torabi, Garrett Warnell, and Peter Stone. Behavioral cloning from observation. *arXiv preprint arXiv:1805.01954*, 2018.

[74] Zhicheng Gu, Zhihao Li, Xuan Di, and Rongye Shi. An LSTM-based autonomous driving model using a waymo open dataset. *Applied Sciences*, 10(6):2046, 2020.

[75] Karen Simonyan and Andrew Zisserman. Very deep convolutional networks for large-scale image recognition. *arXiv preprint arXiv:1409.1556*, 2014.

[76] Sergey Zagoruyko and Nikos Komodakis. Wide residual networks. *arXiv preprint arXiv:1605.07146*, 2016.

[77] Antoine Vacavant, Thierry Chateau, Alexis Wilhelm, and Laurent Lequievre. A benchmark dataset for outdoor foreground/background extraction. In *Asian Conference on Computer Vision*, pages 291–300. Springer, 2012.

[78] David G Lowe. Distinctive image features from scale-invariant keypoints. *International journal of computer vision*, 60(2):91–110, 2004.

[79] Alexander Richard and Juergen Gall. A bag-of-words equivalent recurrent neural network for action recognition. *Computer Vision and Image Understanding*, 156:79–91, 2017.

[80] Fahad Shahbaz Khan, Joost Van De Weijer, Andrew D Bagdanov, and Michael Felsberg. Scale coding bag-of-words for action recognition. In *2014 22nd International Conference on Pattern Recognition*, pages 1514–1519. IEEE, 2014.

[81] Saima Nazir, Muhammad Haroon Yousaf, and Sergio A Velastin. Evaluating a bag-of-visual features approach using spatio-temporal features for action recognition. *Computers & Electrical Engineering*, 72:660–669, 2018.

[82] Alexandros Iosifidis, Anastastios Tefas, and Ioannis Pitas. Discriminant bag of words based representation for human action recognition. *Pattern Recognition Letters*, 49:185–192, 2014.

[83] Josep Maria Carmona and Joan Climent. Human action recognition by means of subtensor projections and dense trajectories. *Pattern Recognition*, 81:443–455, 2018.

[84] Kyuhwa Lee, Dimitri Ognibene, Hyung Jin Chang, Tae-Kyun Kim, and Yiannis Demiris. Stare: Spatio-temporal attention relocation for multiple structured activities detection. *IEEE Transactions on Image Processing*, 24(12):5916–5927, 2015.

[85] Navneet Dalal and Bill Triggs. Histograms of oriented gradients for human detection. In *2005 IEEE computer society conference on computer vision and pattern recognition (CVPR'05)*, volume 1, pages 886–893. IEEE, 2005.

[86] Hossein Rahmani, Arif Mahmood, Du Q Huynh, and Ajmal Mian. Hopc: Histogram of oriented principal components of 3d pointclouds for action recognition. In *European conference on computer vision*, pages 742–757. Springer, 2014.

[87] Muhammet Fatih Aslan, Akif Durdu, and Kadir Sabanci. Human action recognition with bag of visual words using different machine learning methods and hyperparameter optimization. *Neural Computing and Applications*, 32(12):8585–8597, 2020.

[88] Piotr Dollár, Vincent Rabaud, Garrison Cottrell, and Serge Belongie. Behavior recognition via sparse spatio-temporal features. In *2005 IEEE International Workshop on Visual Surveillance and Performance Evaluation of Tracking and Surveillance*, pages 65–72. IEEE, 2005.

[89] Heng Wang and Cordelia Schmid. Action recognition with improved trajectories. In *Proceedings of the IEEE international conference on computer vision*, pages 3551–3558, 2013.

[90] Tej Singh and Dinesh Kumar Vishwakarma. A hybrid framework for action recognition in low-quality video sequences. *arXiv preprint arXiv:1903.04090*, 2019.

[91] Hueihan Jhuang, Thomas Serre, Lior Wolf, and Tomaso Poggio. A biologically inspired system for action recognition. In *2007 IEEE 11th international conference on computer vision*, pages 1–8. Ieee, 2007.

[92] Sijie Yan, Yuanjun Xiong, and Dahua Lin. Spatial temporal graph convolutional networks for skeleton-based action recognition. In *Thirty-second AAAI conference on artificial intelligence*, 2018.

[93] Stepan Komkov, Maksim Dzabraev, and Aleksandr Petiushko. Mutual modality learning for video action classification. *arXiv preprint arXiv:2011.02543*, 2020.

[94] Ceyuan Yang, Yinghao Xu, Jianping Shi, Bo Dai, and Bolei Zhou. Temporal pyramid network for action recognition. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 591–600, 2020.

[95] Christoph Feichtenhofer, Haoqi Fan, Jitendra Malik, and Kaiming He. Slowfast networks for video recognition. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 6202–6211, 2019.

[96] Boyuan Jiang, MengMeng Wang, Weihao Gan, Wei Wu, and Junjie Yan. Stm: Spatiotemporal and motion encoding for action recognition. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 2000–2009, 2019.

[97] Xiang Jiang, Erico N de Souza, Ahmad Pesaranghader, Baifan Hu, Daniel L Silver, and Stan Matwin. Trajectorynet: An embedded gps trajectory representation for point-based classification using recurrent neural networks. *arXiv preprint arXiv:1705.02636*, 2017.

[98] Du Tran, Heng Wang, Lorenzo Torresani, Jamie Ray, Yann LeCun, and Manohar Paluri. A closer look at spatiotemporal convolutions for action recognition. In *Proceedings of the IEEE conference on Computer Vision and Pattern Recognition*, pages 6450–6459, 2018.

[99] Joanna Materzynska, Guillaume Berger, Ingo Bax, and Roland Memisevic. The jester dataset: A large-scale video dataset of human gestures. In *Proceedings of the IEEE International Conference on Computer Vision Workshops*, pages 0–0, 2019.

[100] Raghav Goyal, Samira Ebrahimi Kahou, Vincent Michalski, Joanna Materzynska, Susanne Westphal, Heuna Kim, Valentin Haenel, Ingo Fruend, Peter Yianilos, Moritz Mueller-Freitag, et al. The" something something" video database for learning and evaluating visual common sense. In *ICCV*, volume 1, page 3, 2017.

[101] Tengda Han, Jue Wang, Anoop Cherian, and Stephen Gould. Human action forecasting by learning task grammars. *arXiv:1709.06391*, 2017.

[102] Sam Toyer, Anoop Cherian, Tengda Han, and Stephen Gould. Human pose forecasting via deep Markov models. In *DICTA*, 2017.

[103] Yu Kong and Yun Fu. Human action recognition and prediction: A survey. *arXiv preprint arXiv:1806.11230*, 2018.

[104] Yue Fan, Yongqin Xian, Max Maria Losch, and Bernt Schiele. Spatial information is overrated for image classification. 2019.

[105] Shaohui Lin, Rongrong Ji, Chenqian Yan, Baochang Zhang, Liujuan Cao, Qixiang Ye, Feiyue Huang, and David Doermann. Towards optimal structured CNN pruning via generative adversarial learning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 2790–2799, 2019.

[106] Jiwei Li, Will Monroe, and Dan Jurafsky. Understanding neural networks through representation erasure. *arXiv preprint arXiv:1612.08220*, 2016.

[107] Mostafa Hussein, Brendan Crowe, Marek Petrik, and Momotaz Begum. Robust maximum entropy behavior cloning. *arXiv preprint arXiv:2101.01251*, 2021.

[108] Madison Clark-Turner and Mostafa Hussein. Hierarchical learner, 2021. https://github.com/AssistiveRoboticsUNH/hierarchical_learner.

[109] Sonia Chernova and Andrea L Thomaz. Robot learning from human teachers. *Synthesis Lectures on Artificial Intelligence and Machine Learning*, 8(3):1–121, 2014.

[110] Takayuki Osa, Joni Pajarinen, Gerhard Neumann, J Andrew Bagnell, Pieter Abbeel, and Jan Peters. An algorithmic perspective on imitation learning. *arXiv preprint arXiv:1811.06711*, 2018.

[111] Shun-ichi Amari. *Information geometry and its applications*, volume 194. Springer, 2016.

[112] Edwin T Jaynes. Information theory and statistical mechanics. *Physical review*, 106(4):620, 1957.

APPENDIX A

Spatial Feature Bottleneck

A common modification made to all spatial feature extractors that leverage D-ITR-L is their use of a bottleneck to constrain the number of spatial features used to generate an ITR Graph. Identifying ITRs generates an pairwise number of edges when given a fixed set of spatial features. This can be problematic for a deep learning video inference approach which must simultaneously store the initial pixel-based video input, CNN architecture, ITR graph, GCN architecture, and necessary gradients for a batch-grouped set of data in GPU memory. Unfortunately, the output of a CNN contains a great many spatial features many of which are redundant or uninformative [104, 105, 106]. I reduce the number of features output by the backbone model by inserting a 1×1 convolutional bottleneck layer [66] prior to the model's inference layer. The use of this bottleneck is popular and wide-spread in the design of Residual Network based deep learning models [65]. The number of distinct feature labels to reduce down to is user-defined in this approach but this does not limit the number of times a spatial feature can be expressed through D-ITR-L. Feature expression can rise and fall throughout a video which can lead to multiple events in which a feature with the same label can be expressed. This is necessary to capture cyclical expression of a given feature in a video. The maximum number of times a feature of a specific label can be expressed is equal to half the number of frames in a video (if the feature is expressed on every odd frame and is not expressed on every even frame), though this is highly unlikely given that real world data is rarely so polarized. Given this limitation I established (through trial and error) that a maximum of 64 unique spatial features should be used when conducting inference on a

computer using 2 Titan X GPUs. This value assumes input of videos of up to 5000 frames in length and is dependent upon both the resource demands of the Backbone CNN and hyper-parameters such as batch size.

APPENDIX B

IRB Approval

University *of* New Hampshire

Research Integrity Services, Service Building
51 College Road, Durham, NH 03824-3585
Fax: 603-862-3564

16-Nov-2016

Begum, Momotaz
Computer Science
61 Edgemere Blvd
Shrewsbury, MA 01545

IRB #: 6578
Study: Robot Mediated Behavioral Intervention
Approval Date: 11-Nov-2016

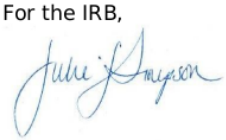
The Institutional Review Board for the Protection of Human Subjects in Research (IRB) has reviewed and approved the protocol for your study as Expedited as described in Title 45, Code of Federal Regulations (CFR), Part 46, Subsection 110.

Approval is granted to conduct your study as described in your protocol for one year from the approval date above. At the end of the approval period, you will be asked to submit a report with regard to the involvement of human subjects in this study. If your study is still active, you may request an extension of IRB approval.

Researchers who conduct studies involving human subjects have responsibilities as outlined in the document, *Responsibilities of Directors of Research Studies Involving Human Subjects*. This document is available at http://unh.edu/research/irb-application-resources. Please read this document carefully before commencing your work involving human subjects.

If you have questions or concerns about your study or this approval, please feel free to contact me at 603-862-2003 or Julie.simpson@unh.edu. Please refer to the IRB # above in all correspondence related to this study. The IRB wishes you success with your research.

For the IRB,



Julie F. Simpson
Director