The 2016 AAAI Fall Symposium Series:
Shared Autonomy in Research and Practice
Technical Report FS-16-05

Learning and Predicting Sequential Tasks Using Recurrent Neural Networks and Multiple Model Filtering

Harish Chaandar Ravichandar, Avnish Kumar, Ashwin P. Dani, Krishna R. Pattipati
Department of Electrical and Computer Engineering,
University of Connecticut, Storrs, CT 06269.
Email: {harish.ravichandar, avnish.kumar, ashwin.dani, krishna.pattipati}@uconn.edu

Abstract

An integral part of human-robot collaboration is the ability of the robot to understand and predict human motion. Predicting what the human collaborator will do next is very useful in planning the robot's response. In this paper, an algorithm for early detection and prediction of human activities is presented. For a given sequential task composed of many steps, a long short-term memory (LSTM) recurrent neural network (RNN) model is trained to learn the underlying sequence of steps. The trained network is then used to make predictions about the subsequent steps the human is about to carry out. The prediction of next steps requires information about the current step that is being carried out. The steps are inferred by observing the motion trajectories of the human arm and predicting where the human is reaching. The trajectories of the arm motion are modeled by using a dynamical system with contracting behavior towards the object. A neural network (NN) is used to learn the dynamics under the contraction analysis constraints. An interacting multiple model (IMM) framework is used for the early prediction of the goal locations of reaching motions. Since humans tend to look in the direction of the object they are reaching for, the prior probabilities of the models are calculated based on the human eye gaze. Experimental results based on an audio amplifier circuit assembly task are used to validate the proposed algorithm.

Introduction

Human activity prediction and intention inference are useful in achieving safe and efficient Human-Robot Collaboration (HRC) and shared autonomy (Klein et al., 2004). Shared autonomy is typically achieved either by giving control to the human or the robot at some specific situations. The task sharing between agents requires anticipation of other agent's actions for optimally choosing one's own actions. Studies in psychology suggest that when two humans interact, they infer each other's intended actions for safe and efficient interaction (Baldwin and Baird, 2001). Taking inspiration from how humans interact with each other, the safety, operational efficiency, and task reliability in HRC could be greatly improved by providing a robot with the capability to infer human actions and intentions. Tasks, such as an assembly task often contain sub-tasks, each containing steps that have to be completed in a specific order while the order in which the sub-tasks are completed could be immaterial. For instance, in a circuit assembly task, sub-tasks such as (1) connecting the power supply to the board (sub-task 1) and (2) aligning the components on the board (sub-task 2) might involve steps that have to be completed in a specific order. However, one might complete Sub-task 1 before completing Sub-task 2 and vice-verse. This fact makes the problem of predicting the human agent's next step very challenging. In this paper, a recurrent neural network (RNN) is used to learn the underlying sequence of steps involved in a sequential task. The knowledge of the underlying sequence is very helpful in either (1) synthesizing a sequence, or (2) making predictions about the subsequent steps given a set of initial steps. Unlike regressive models, such as NNs and Gaussian mixture models (GMMs), RNNs have the ability to capture the temporal nature of a sequence. Other modeling alternatives such as hidden Markov models (HMMs) suffer from the disadvantage of the assumption that the probability of each observation depends only on the current state, which makes contextual effects difficult to model (Graves et al., 2009). While a standard RNN can model a wide variety of contextual effects, it suffers from a well-known problem of vanishing gradients that makes it difficult to capture long range temporal dependencies (cf. (Pascanu, Mikolov, and Bengio, 2013)). To circumvent this issue, the proposed framework uses Long Short-Term Memory (LSTM) units that are suitable for learning sequences that have variable-range temporal dependencies (Hochreiter and Schmidhuber, 1997). The key contributor to an LSTM unit's ability to learn long range temporal dependencies is the cell state that carries information about previous time steps. The LSTM units contain different gates that are used to control the information entering and leaving the cell state. More information regarding the LSTM units can be found in Section 2.

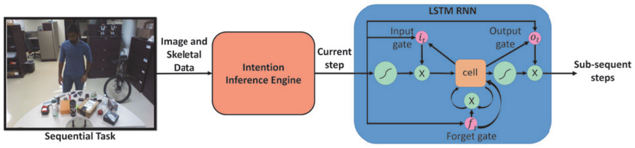

Figure 1: Proposed sequence prediction framework.

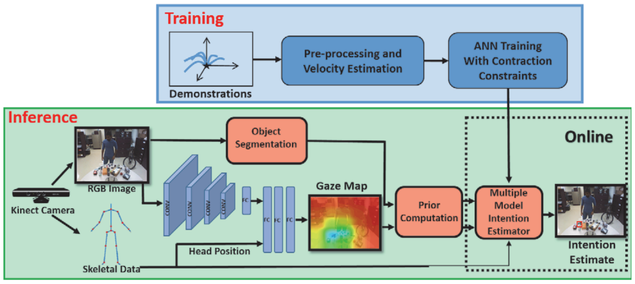

Figure 2: Block diagram of multiple model filtering for intention inference with gaze-based priors.

In order to make the prediction about the next step, the trained LSTM-RNN network requires information about the current step. To this end, a neural network (NN) is used to learn the dynamics of human arm's reaching motion and the trained model is used to infer human intentions. The human intention is modeled as the goal location of a reaching motion. The problem of learning the arm dynamics is formulated as a parameter learning problem with the constraints given by the contraction analysis of nonlinear systems (Lohmiller and Slotine, 1998). Once the NN is trained, intention inference is carried out by using the multiple-model-intention estimator (MMIE) presented in (Ravichandar and Dani, 2015b). The MMIE algorithm uses an interacting multiple model (IMM) filtering approach in which the posterior probabilities of all possible goal locations are computed through model-matched filtering (cf. (Bar-Shalom, Li, and Kirubarajan, 2001)). However, the MMIE algorithm is not suitable for cluttered environments with a large number of objects or possible goal locations. This is due to the fact that the MMIE algorithm will have to consider a large number of models and run many filters in parallel to carry out the inference. A carefully designed prior distribution that is based on heuristics would render the MMIE algorithm suitable to such scenarios by reducing the number of possible goal locations and the time of inference. In this work, an algorithm termed gaze-based MMIE (G-MMIE) is used to calculate the prior distribution based on human eye gaze (Ravichandar, Kumar, and Dani, 2016).

In (Flanagan and Johansson, 2003), it is demonstrated that adults predict action goals by fixating their eye gaze on the end location of an action before it is reached, both when they execute an action themselves and when they observe someone else executing an action. In (Gredebäck and Falck-Ytter, 2015), a survey of various studies that followed (Flanagan and Johansson, 2003) is presented. These studies in (Flanagan and Johansson, 2003; Gredebäck and Falck-Ytter, 2015) suggest that the use of human eye gaze would be helpful in predicting the intention or goal location of human reaching motions. In this paper, the human eye-gaze is used as a heuristic to compute the prior probabilities of the possible goal locations in order to perform early prediction of the goal location of human reaching actions. The convolution neural network (CNN)-based gaze estimation algorithm, presented in (Recasens* et al., 2015), is used to obtain a gaze map. The gaze map is a spatial map that contains the probability of each pixel being the gaze point. The prior distribution is subsequently used in a Bayesian setting to obtain a maximum-a-posteriori (MAP) estimate of the goal location. The block diagrams of the sequence prediction and intention inference approaches are shown in Figs. 1 and 2.

Related Work

Sequence Learning and Prediction In (Koppula, Gupta, and Saxena, 2013), the intended activities of human subjects are inferred by modeling the spatio-temporal relations and object affordances. While the work in (Koppula, Gupta, and Saxena, 2013) proposes a rich model to include object affordances, sub-activities, and their interactions, it aims to label the on-going activity or sub-activity as opposed to predicting the long term future actions of the human. In (Kelley et al., 2008), the intentions of human agents are inferred by using a HMM of the robot's experience and its interaction with the environment. The work in (Kelley et al., 2008) aims to infer intentions for lower granularity goals, such as sub-tasks, before they have been achieved. A vision-based probabilistic algorithm that infers ongoing activities in video sequences is presented in (Ryoo, 2011) where the activities are represented as integral histograms of spatio-temporal features. In (Lan, Chen, and Savarese, 2014), a representation called *hierarchical movements* is presented to describe human activities in a hierarchical manner and predict future actions from still images or short video clips. While the works in (Koppula, Gupta, and Saxena, 2013), (Kelley et al., 2008), (Ryoo, 2011), and (Lan, Chen, and Savarese, 2014) solve the human activity recognition/anticipation problem as a detection problem, the proposed approach models human activity using LSTM-RNNs to make predictions about the sequence of future actions. In (Jain et al., 2016), deep RNNs with LSTM units are used to predict the driving maneuvers using data collected from several sensors. The algorithm in (Jain et al., 2016) uses features extracted from multiple sensors and iteratively predicts the human actions. Given the set of observations, the algorithm labels the *higher level* activity that the human is about to perform. In contrast, the algorithm developed in this paper uses a fusion of IMM filtering framework (for goal location prediction) and LSTM-RNNs to make predictions about the sequence of future actions. In (Hawkins et al., 2013), a Bayesian network is proposed to predict the probability distribution over future actions in the presence of noisy or unreliable sensors. While the algorithm considers possible future actions and decides on a cost-driven robot response, the prediction of future actions cannot be made until the current task is fully observed. In contrast, the proposed framework leverages the intention inference algorithm to infer the current action *before* it is complete. Deep networks are used in (Vondrick, Pirsiavash, and Torralba, 2016) to anticipate the actions and objects in the future by learning their hidden representations from massive amounts of unlabeled videos. In contrast to the algorithm proposed in (Vondrick, Pirsiavash, and Torralba, 2016), the algorithm developed in this paper is able to learn temporal context which can be crucial in many sequential task-based predictions. Furthermore, the algorithm is not restricted to video data.

Intention Inference Human intentions are inferred by estimating/measuring information about body posture (Strabala et al., 2013), gesture and eye gaze (Hart et al., 2014), object affordances (Koppula, Gupta, and Saxena, 2013), and human skeletal movement (Mainprice, Hayne, and Berenson, 2015). Human intention inference has been studied by using hidden Markov models (HMMs) (Ding et al., 2011), growing HMMs (GHMMs) (Elfring, Van De Molengraft, and Steinbuch, 2014), Conditional Random Fields (CRFs) (Koppula, Gupta, and Saxena, 2013), and Gaussian mixture models (GMMs) (Luo and Berenson, 2015). The G-MMIE algorithm models the human arm motion as a continuous nonlinear dynamical system (DS) of human skeletal joints approximated using an NN. The G-MMIE algorithm infers the goal location of reaching motions by leveraging both the human arm motion data and the eye gaze information which is used as a heuristic to compute the prior distribution of the goal location.

Sequential Task Learning and Prediction Architecture

Learning Task Sequences using LSTM-RNNs

Given a sequential task, the paper addresses the problem of learning the sequence of steps involved in a task. Since the steps involved in sequential tasks, by nature, have temporal inter-dependence, recurrent neural networks (RNN) are natural candidates to learn the sequence. However, since simple RNNs suffer from the problem of vanishing gradients, the proposed algorithm uses the LSTM architecture proposed in (Gers, Schraudolph, and Schmidhuber, 2002). The LSTM architecture uses memory cells to store relevant information needed to learn long range temporal dependencies in the data. The hidden layer of the LSTM architecture is implemented by the following composite function

$$i_t = \sigma(W_{xi}x_t + W_{hi}h_{t-1} + W_{ci}c_{t-1} + b_i) \quad (1)$$

$$f_t = \sigma(W_{xf}x_t + W_{hf}h_{t-1} + W_{cf}c_{t-1} + b_f) \quad (2)$$

$$c_t = f_t c_{t-1} + i_t \tanh(W_{xc}x_t + W_{hc}h_{t-1} + b_c) \quad (3)$$

$$o_t = \sigma(W_{xo}x_t + W_{ho}h_{t-1} + W_{co}c_{t-1} + b_o) \quad (4)$$

$$h_t = o_t \tanh(c_t) \quad (5)$$

where σ is the logistic sigmoid function, and i, f, o and c are, respectively, the input gate, forget gate, output gate, cell, and cell input activation vectors, all of which are of the same size as the hidden vector h. The subscripts of the weight matrices are chosen according to the associated layers and gates. For instance, W_{hi} is the hidden-input gate matrix and W_{xo} is the input-output gate matrix. The weight matrices from the cell to gate vectors (e.g.W_{ci}) are diagonal, so element m in each gate vector only receives input from element m of the cell vector. The LSTM-RNN is trained by using the error back propagation through time (BPTT) algorithm. The training algorithm learns the weights of the network by minimizing the loss function given by $loss = -\frac{1}{N_d}\sum_{i=1}^{N_d} ln(p_i)$ where N_d is the number of target sequences and p_i is the perplexity for the ith target sequence. (See (Williams and Zipser, 1995) for details).

Computation of Prior Distribution using Gaze Map

The CNN model from (Recasens* et al., 2015) is used to estimate the gaze map of a given RGB image of size $D_w \times D_h$. The input to the CNN is an image taken from the Kinect camera and the relative position of the subject's head in that image. The output is the gaze map $\mathcal{G}$ of size $D_w \times D_h$ containing the probabilities of each pixel being the gaze point. See (Recasens* et al., 2015) for a detailed description of the CNN framework.

The average prior probability $\bar{p}_j(0)$ of the jth object in the scene is calculated as $\bar{p}_j(0) = \sum_{i\in\mathcal{G}P_j}(\mathcal{G}(i)/NP_j)$, where $\mathcal{G}(i)$ is the probability of the ith pixel being the gaze point, NP_j is the number of pixels associated with the jth object, and $\mathcal{G}P_j$ is the set of all pixel locations associated with the jth object. Of all the objects in the scene, the objects that correspond to the top N_g average prior probabilities are chosen as the likely goal locations. The prior probability of each of the N_g objects being the goal location is given by

$$\mu_j(0) = \frac{\sum_{i\in\mathcal{G}P_j}(\mathcal{G}(i)/NP_j)}{\sum_{j=1}^{N_g}\sum_{i\in\mathcal{G}P_j}(\mathcal{G}(i)/NP_j)} \quad (6)$$

where $\mu_j(0)$ is the prior probability of g_j being the goal location and g_j refers to the location of the jth object.

Intention Inference using G-MMIE Algorithm

Given a sequential task, prediction of what step a human is about to take requires knowledge of the current step being carried out. The G-MMIE algorithm infers this knowledge by predicting the goal location of human arm reaching motions. In order to predict the goal location of reaching motions, the underlying dynamics of the motion are learned from data obtained from activity using a neural network (NN). Since all the trajectories of the demonstrations would converge to the goal location, one of the ways to model the underlying dynamic system is by using contracting dynamic system. Hence the NN model is trained under constraints given by the contraction analysis of nonlinear systems (Lohmiller and Slotine, 1998). Details pertaining to the learning framework can be found in (Ravichandar and Dani, 2015c). Given the trained network and measurements of the human hand, the problem involves inferring the goal location of the human hand ahead of time. Let a finite set of possible goal locations that the human can reach be $G = \{g_1, g_2, \cdot\cdot\cdot, g_{N_g}\}$. The trained NN is a globally contracting system that generates trajectories leading to the origin given by $x = 0_{n\times1}$. For each goal location g_i, the state vector and the corresponding dynamics are defined as $x^j(t) = \left[[x_{pos}(t) - g_j]^T, x_{vel}^T(t)\right]^T$, $\dot{x}^j(t) = f(x^j(t))$, respectively, where $f(\cdot)$ represents the trained NN model. Similarly, for a set of N_g goal locations, a set of N_g dynamic systems is formed. The discretized versions of these systems are given by

$$x^j(k+1) = x^j(k) + T_s f\left(x^j(k)\right) + T_s w(k) \quad (7)$$

for $j = 1, 2, ..., N_g$, where T_s is the sampling period. The measurement model is given by

$$z(k) = h(x(k)) + v(k), \qquad j = 1, 2, ..., N_g \quad (8)$$

where $z(k) \in \mathbb{R}^3$ is the measurement vector containing the position of the reaching hand at time instant k, $v \sim \mathcal{N}(0, R)$ is a zero mean Gaussian process with covariance R and $h(x(k)) = \left[\begin{smallmatrix} 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 \end{smallmatrix}\right] x(k)$ is the measurement function.

Given that $M_1, M_2, ..., M_{N_g}$ represent the N_g models defined in (7) and (8) for the set of possible goal locations G, the objective is to estimate $p(g_j|Z_{1:k})$. The expression $p(g_j|Z_{1:k})$ denotes the posterior probability of each g_j being the actual goal location given a set of k measurements $Z_{1:k} = [z(1), z(2), \cdot\cdot\cdot, z(k)]^T$. The posterior probability $p(g_j|Z_{1:k})$ is calculated using the Bayes' theorem as follows $p(g_j|Z_{1:k}) = \frac{p(Z_{1:k}|g_j)p(g_j)}{\sum_{j=1}^{N} p(Z_{1:k}|g_j)p(g_j)}$, where $p(Z_{1:k}|g_j)$ is the likelihood of g_j and $p(g_j)$ is the prior probability of g_j. Note that $p(g_j(k)|Z_{1:k}) = p(M_j|Z_{1:k})$ since the models differ only by the goal location. Hence, in order to obtain the posterior probabilities $p(g_j|Z_{1:k}),\ j = 1, .., N_g$, the posterior probabilities of the models $p(M_j|Z_{1:k}),\ j = 1, .., N_g$ is computed. To this end, calculation of the likelihood functions $p(Z_{1:k}|M_j)$ is necessary. In the IMM framework with N_g models, the likelihood functions are computed using N_g filters running in parallel. The G-MMIE algorithm uses extended Kalman filters (EKFs). Each iteration of the IMM filter for intention inference is divided into three main steps (cf. (Bar-Shalom, Li, and Kirubarajan, 2001)). These steps are summarized in the remainder of this subsection.

Interaction/Mixing: At the beginning of each iteration, the initial conditions (state estimate $\hat{x}^{0j}(k-1|k-1)$ and covariance $\hat{P}^{0j}(k-1|k-1)$, where superscript 0 denotes the initial condition, j denotes the number of the filter), at time k, are adjusted by mixing the filter outputs from the previous iteration (time instant $k-1$) in the following way

$$\begin{aligned} \hat{x}^{0j}(k-1|k-1) =& \sum_{i=1}^{N_g} \hat{x}^{i}(k-1|k-1) \times \\ & \mu_{i|j}(k-1|k-1), \quad j = 1, 2, .., N_g \end{aligned} \tag{9}$$

$$\begin{aligned} \hat{P}^{0j}(k-1|k-1) =& \sum_{i=1}^{N_g} \mu_{i|j}(k-1|k-1) \\ \{\hat{P}^{i}(k-1|k-1) + [&\hat{x}^{i}(k-1|k-1) - \hat{x}^{0j}(k-1|k-1)] \\ \times [\hat{x}^{i}(k-1|k-1) &- \hat{x}^{0j}(k-1|k-1)]^T\}, \end{aligned} \tag{10}$$

for $j = 1, 2, ..., N_g$, where $\hat{x}^{i}(k-1|k-1),\ \hat{P}^{i}(k-1|k-1)$ are the state estimate and its covariance, respectively, corresponding to model M_i at time $k-1$ and the mixing probabilities $\mu_{i|j}(k-1|k-1)$ are given by $\mu_{i|j}(k-1|k-1) = \frac{\Pi_{ij}\mu_i(k-1)}{\bar{c}_j},\ i, j = 1, 2, ..., N_g$, where $\Pi_{ij} = p(M(k) = M_j|M(k-1) = M_i)$ is the model transition or jump probability and $\mu_i(k-1) = p(M_i|Z_{1:k-1})$ is the probability of ith model M_i being the right model at time $k-1$ and $\bar{c}_j = \sum_{i=1}^{N} \Pi_{ij}\mu_i(k-1)$ are the normalizing constants.

Model Matched Filtering: With the initial conditions $\hat{x}^{0j}(k-1|k-1)$ and $\hat{P}^{0j}(k-1|k-1)$, the state estimate and its covariance for each model are computed using the EKFs matched to the models. Along with the state estimates and the corresponding covariances, the likelihood functions $\Lambda_j(k)$ are computed using the mixed initial condition (9) and using the corresponding covariance (10). The likelihood function $\Lambda_j(k)$, a Gaussian distribution with the predicted measurement as the (filter initial condition-dependent) mean and the covariance equal to the innovation covariance, is given by

$$\begin{aligned} \Lambda_j(k) =& p(z(k)|M_j(k), \hat{x}^{0j}(k-1|k-1), \\ & \hat{P}^{0j}(k-1|k-1)) \\ \Lambda_j(k) =& \mathcal{N}(z(k); \hat{z}^{j}(k|k-1; \hat{x}^{0j}(k-1|k-1)), \\ & S^{j}(k; \hat{P}^{0j}(k-1|k-1))), \quad j = 1, 2, .., N_g \end{aligned} \tag{11}$$

where $S^j(\cdot)$ is the innovation covariance and $\hat{z}^j(\cdot)$ is the jth filter's predicted measurement at time k.

Model Probability Update: Using $\Lambda_j(k)$, the model posterior probabilities $\mu_j(k)$ are calculated as follows

$$\begin{aligned} \mu_j(k) &= p(g_i|Z_{1:k}) = p(M_j(k)|Z_{1:k}) \\ \mu_j(k) &= p(z(k)|M_j(k), Z_{1:k-1})p(M_j(k)|Z_{1:k-1}) \\ \mu_j(k) &= \frac{\Lambda_j(k)\bar{c}_j}{\sum_{j=1}^{N} \Lambda_j(k)\bar{c}_j} \qquad j = 1, 2, ..., N_g \end{aligned} \tag{12}$$

The goal location estimate $\hat{g}(k) = \arg\max_{g \in G}\ p(g|Z_{1:k})$ is computed by choosing the location $g_i \in G$ corresponding to the model M_i with the highest model probability $\mu_i(k)$ at time k. The implementation details of the proposed algorithm are given in Algorithm 1.

Algorithm 1: Learning and Prediction of Sequential tasks

```
Obtain training sequences of the task;
Choose size of the LSTM-RNN network; Train the
LSTM network with the available demonstrations to
obtain the network weights;
while Task is being performed do
    Predict the goal location of the current reaching
    motion using the G-MMIE Algorithm;
    Based on the predicted goal location, infer the
    on-going step of the task;
    Supply the inferred current step as input to the
    trained LSTM to predict the next step(s) in the
    sequence;
end
```

Experimental Validation

In order to validate the proposed algorithm, an audio amplifier circuit assembly task is considered. Demonstrations of a human subject (Subject 1) building an audio amplifier circuit on a bread board are recorded. The demonstrations contained nine sub-tasks involving 2 steps and one sub-task involving 3 steps. Each sub-task consists of one reaching motion. The sequence of sub-tasks of the demonstration and the corresponding steps and labels are given in Table 1. The same circuit could be built in several ways, each differing in the order in which the components are put together. Indeed,

Table 1: Audio amplifier circuit assembly sequence (* indicates reaching motion).

Sub-task #	Step	Label
1	Pick up op-amp*	a
	Place op-amp	b
2	Pick up wire*	c
	Connect $V_{cc}+$	d
3	Pick up wire*	c
	Connect $V_{cc}-$	e
4	Pick up LED*	f
	Connect LED to V_{out}	g
5	Pick up wire*	c
	Ground other end of LED	h
6	Pick up wire*	c
	Ground non-inverting input port of op-amp	i
7	Pick up resistor*	j
	Connect as feedback resistor	k
8	Pick up resistor*	j
	Connect as input resistor	l
9	Pick up capacitor*	m
	Connect in series with input resistor	n
10	Pick up wire*	c
	Cut the wire	o
	Connect to inverting input port	p

the circuit could be built in $362,880$ ways when the first sub-task is fixed and the remaining nine sub-tasks are allowed to be completed in any order. For instance, once the op-amp is placed, one could choose to connect a wire to $V_{cc}-$ before or after connecting a wire to $V_{cc}+$. All data are collected by using a Microsoft Kinect for Windows VI in the Robotics and Controls lab at UConn. The algorithm is coded in Python 2.7 and Matlab 2014a, and is run on a standard desktop computer running Intel i3 $-$ 3240 processor with 8 Gigabytes of memory.

Experiment 1

The objective of the first experiment is to show that the G-MMIE algorithm can be used to predict the goal locations of reaching motions and thereby identify the ongoing sub-task or activity. A set of 10 reaching motions are collected from another human subject (Subject 2) and used for training the NN under contraction analysis constraints as explained in section on Learning Contracting Nonlinear Dynamics of Reaching Motion. In training, each demonstration is labeled based on the ground truth goal location. The joint position data obtained from the subjects are pre-processed to obtain the velocity and acceleration estimates using a Kalman filter. The goal location of all the 10 reaching motions of the circuit assembly task are successfully inferred by the proposed algorithm. The position and velocity of the hand in 3-dimensional (3D) Cartesian space are considered to be the elements of the state vector $x(k) \in \mathbb{R}^6$. The initial state estimate covariance $\hat{P}^j(0)$, $j = 1, 2, \cdot\cdot, N_g$, the process noise covariance Q^j , $j = 1, 2, \cdot\cdot, N_g$, the number of possible goal locations $N_g = 5$, and the measurement noise covariance R are selected to be $0.2I_{6\times6}$, $0.1I_{6\times6}$, and $0.2I_{6\times6}$, respectively. The state estimates $\hat{x}^j$, $j = 1, 2, \cdot\cdot, N_g$, are initialized using the first two measurements $z(1)$ and $z(2)$ (a finite difference method is used for the velocity initialization). The inferred goal location is used to determine the on-going activity. The average time of inference of the G-MMIE algorithm over all 10 reaching motions is found to be 0.47 seconds. In Fig. 3, the percentages of tests where the intention is correctly inferred by the G-MMIE algorithm, the ANIE algorithm (Ravichandar and Dani, 2015a), and the unsupervised GMM algorithm (Luo and Berenson, 2015) are shown as a function of time. Results pertaining to the G-MMIE algorithm's performance on other datasets can be found in (Ravichandar, Kumar, and Dani, 2016).

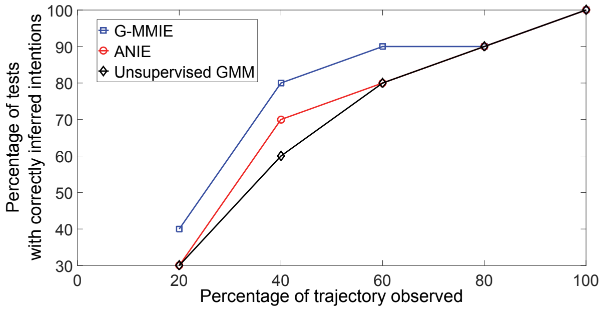

Figure 3: Percentage of tests with correctly inferred intention as a function of the percentage of trajectory observed over 10 reaching trajectories.

Experiment 2

The objective of the second experiment is to learn and predict the circuit assembly sequence. LSTM-RNN is trained from $310,000$ instances of the circuit assembly demonstration. The instances are generated by randomly varying the order of the 10 sub-tasks involved in the circuit assembly task. In order to test the trained network, an additional set of $50,000$ instances of the circuit assembly tasks are created. The LSTM-RNN contains 128 neurons in the hidden layer and is trained for 3 epochs on the training dataset with a constant learning rate of 0.002. The trained network is tested in the following three ways.

1) One step prediction: Given the sequence of steps until the current step, the trained LSTM-RNN is used to predict the next step in the circuit assembly sequence. This test helps understand the predictive or anticipatory performance of the network while the user is performing the circuit assembly task. The ability to predict the next step the user is about to perform can potentially be used in robot's motion planning. The tests involved $100,000$ one-step predictions based on the $50,000$ test sequences (20 predictions per sequence). In Fig. 4, the accuracy of one-step predictions as a function of number of steps observed for LSTM-RNN and RNN are shown. The probability of randomly choosing the next two steps correctly is also included.

2) Two-step prediction: Given the sequence of steps until the current step, the trained LSTM-RNN is used to predict the next two steps in the circuit assembly sequence. The tests involved $950,000$ two-step predictions based on the $50,000$ test sequences (19 predictions per sequence). In Fig. 5, the accuracy of two-step predictions as a function of number of

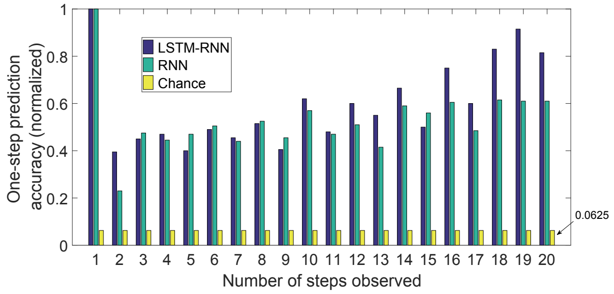

Figure 4: Percentage of correct one-step predictions as a function of number of completed steps.

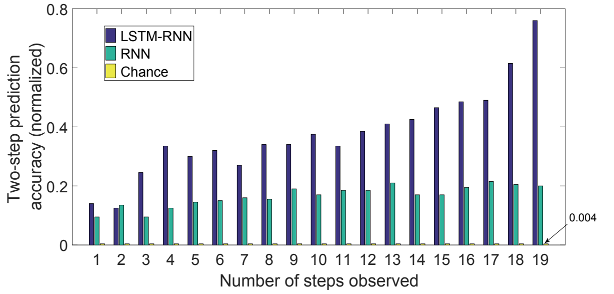

Figure 5: Percentage of correct two-step predictions as a function of number of completed.

steps observed for LSTM-RNN and RNN are shown. The probability of randomly choosing the next two steps correctly is also included. Each predicted step is checked for correctness against the true sequence in order to compute the percentage of correct predictions.

3) Sequence generation accuracy: Given varied number of initial steps, the trained LSTM-RNN is used to predict the rest of the circuit assembly sequence. A generated sequence is deemed to be correct if it will lead to a complete correct audio amplifier circuit. More specifically, the correctness of a generated sequence is determined by checking (1) if the sequence contains all the necessary sub-tasks, (2) if the steps involved in each sub-task is predicted in correct sequence, and (3) if each sub-task is performed only the required number of times. A total of $100,000$ sequence predictions based on the $50,000$ test sequences (20 predictions per sequence) are made using the trained network. In Fig. 6, the accuracy of sequence predictions as a function of number of steps observed for LSTM-RNN and RNN are shown. The probability of randomly choosing the next two steps correctly is also included. It is also observed that the network always generates the steps of any given sub-task in the correct order.

Discussion

The results from Experiment 1 suggest that the G-MMIE algorithm is suitable for providing information about the current step even before it is complete, thereby, providing the LSTM-RNN with the ability to predict the subsequent steps before the current step is complete. In Experiment 2, it is observed that the LSTM-RNN is capable of learning and making prediction about an audio amplifier assembly task. The LSTM-RNN is shown to make predictions about the subsequent steps as early as the first step. The comparisons against the simple RNN and chance-based prediction indicate that the overall performance of LSTM-RNN is better than that of the RNN. This observation is believed to be a result of the LSTM-RNN's ability to learn long-range temporal dependencies. It is also observed RNN performs better than LSTM-RNN at few instances, especially, in the case of one-step predictions. This could be a result of the random initialization of the network parameters before training.

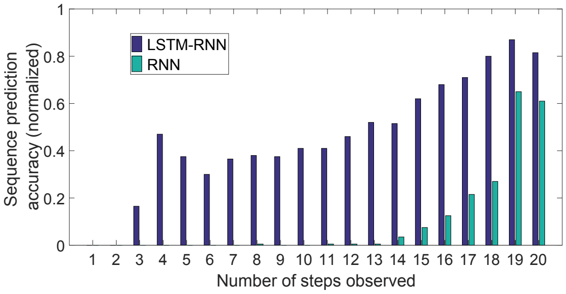

Figure 6: Percentage of correct sequences generated by the LSTM network as a function of number of initial steps.

Conclusion

An algorithm to learn and predict sequential human activities using an LSTM-RNN is presented. The current step is inferred through the early prediction of the goal location (intention) of human reaching motion in a multiple model Bayesian framework. The prior probabilities of the possible goal locations are calculated based on the human eye gaze. Experiment 1 involved the validation of the intention inference algorithm, while Experiment 2 involved the validation of the network's ability to learn sequential tasks. As part of future work, the probability distribution over the future steps will be considered in order provide better predictions and plan appropriate robot response.

Acknowledgments

Pattipati's work was supported by NSF grant CCF-1331850 (NSF CyberSEES), ONR grants N00014-12-1-0238 and N00014-16-1-2036, and QUBE HASI grant N00173-12-2-C902.

References

Baldwin, D. A., and Baird, J. A. 2001. Discerning intentions in dynamic human action. *Trends in Cognitive Sciences* 5(4):171–178.

Bar-Shalom, Y.; Li, X. R.; and Kirubarajan, T. 2001. *Estimation with Applications to Tracking and Navigation*. John Wiley and Sons.

Ding, H.; Reißig, G.; Wijaya, K.; Bortot, D.; Bengler, K.; and Stursberg, O. 2011. Human arm motion modeling and long-term prediction for safe and efficient human-robot-interaction. In *IEEE International Conference on Robotics and Automation (ICRA)*, 5875–5880.

Elfring, J.; Van De Molengraft, R.; and Steinbuch, M. 2014. Learning intentions for improved human motion prediction. *Robotics and Autonomous Systems* 62(4):591–602.

Flanagan, J. R., and Johansson, R. S. 2003. Action plans used in action observation. *Nature* 424(6950):769–771.

Gers, F. A.; Schraudolph, N. N.; and Schmidhuber, J. 2002. Learning precise timing with LSTM recurrent networks. *Journal of Machine Learning Research* 3(Aug):115–143.

Graves, A.; Liwicki, M.; Fernández, S.; Bertolami, R.; Bunke, H.; and Schmidhuber, J. 2009. A novel connectionist system for unconstrained handwriting recognition. *IEEE Transactions on Pattern Analysis and Machine Intelligence* 31(5):855–868.

Gredebäck, G., and Falck-Ytter, T. 2015. Eye movements during action observation. *Perspectives on Psychological Science* 10(5):591–598.

Hart, J. W.; Gleeson, B.; Pan, M.; Moon, A.; MacLean, K.; and Croft, E. 2014. Gesture, gaze, touch, and hesitation: Timing cues for collaborative work. In *HRI Workshop on Timing in Human-Robot Interaction, Bielefeld, Germany*.

Hawkins, K. P.; Vo, N.; Bansal, S.; and Bobick, A. F. 2013. Probabilistic human action prediction and wait-sensitive planning for responsive human-robot collaboration. In *IEEE-RAS International Conference on Humanoid Robots (Humanoids)*, 499–506.

Hochreiter, S., and Schmidhuber, J. 1997. Long short-term memory. *Neural Computation* 9(8):1735–1780.

Jain, A.; Singh, A.; Koppula, H. S.; Soh, S.; and Saxena, A. 2016. Recurrent neural networks for driver activity anticipation via sensory-fusion architecture. In *IEEE International Conference on Robotics and Automation (ICRA)*, 3118–3125.

Kelley, R.; Tavakkoli, A.; King, C.; Nicolescu, M.; Nicolescu, M.; and Bebis, G. 2008. Understanding human intentions via hidden markov models in autonomous mobile robots. In *ACM/IEEE International Conference on Human Robot Interaction*, 367–374.

Klein, G.; Woods, D. D.; Bradshaw, J. M.; Hoffman, R. R.; and Feltovich, P. J. 2004. Ten challenges for making automation a "team player" in joint human-agent activity. *IEEE Intelligent Systems* 19(6):91–95.

Koppula, H. S.; Gupta, R.; and Saxena, A. 2013. Learning human activities and object affordances from rgb-d videos. *The International Journal of Robotics Research* 32(8):951–970.

Lan, T.; Chen, T.-C.; and Savarese, S. 2014. A hierarchical representation for future action prediction. In *European Conference on Computer Vision*, 689–704.

Lohmiller, W., and Slotine, J.-J. E. 1998. On contraction analysis for nonlinear systems. *Automatica* 34(6):683–696.

Luo, R., and Berenson, D. 2015. A framework for unsupervised online human reaching motion recognition and early prediction. In *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2426–2433.

Mainprice, J.; Hayne, R.; and Berenson, D. 2015. Predicting human reaching motion in collaborative tasks using inverse optimal control and iterative re-planning. In *IEEE International Conference on Robotics and Automation (ICRA),*, 885–892.

Pascanu, R.; Mikolov, T.; and Bengio, Y. 2013. On the difficulty of training recurrent neural networks. *International Conferce on Machine Learning* 28:1310–1318.

Ravichandar, H., and Dani, A. P. 2015a. Human intention inference and motion modeling using approximate EM with online learning. In *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 1819–1824.

Ravichandar, H., and Dani, A. P. 2015b. Human intention inference through interacting multiple model filtering. In *IEEE International Conference on Multisensor Fusion and Integration for Intelligent Systems (MFI)*, 220–225.

Ravichandar, H., and Dani, A. P. 2015c. Learning contracting nonlinear dynamics from human demonstration for robot motion planning. In *ASME Dynamic Systems and Control Conference*.

Ravichandar, H.; Kumar, A.; and Dani, A. P. 2016. Bayesian human intention inference through multiple model filtering with gaze-based priors. In *International Conference on Information Fusion*.

Recasens*, A.; Khosla*, A.; Vondrick, C.; and Torralba, A. 2015. Where are they looking? In *Advances in Neural Information Processing Systems (NIPS)*. * indicates equal contribution.

Ryoo, M. S. 2011. Human activity prediction: Early recognition of ongoing activities from streaming videos. In *IEEE International Conference on Computer Vision*, 1036–1043.

Strabala, K. W.; Lee, M. K.; Dragan, A. D.; Forlizzi, J. L.; Srinivasa, S.; Cakmak, M.; and Micelli, V. 2013. Towards seamless human-robot handovers. *Journal of Human-Robot Interaction* 2(1):112–132.

Vondrick, C.; Pirsiavash, H.; and Torralba, A. 2016. Anticipating visual representations with unlabeled video. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*.

Williams, R. J., and Zipser, D. 1995. Gradient-based learning algorithms for recurrent connectionist networks. In Chauvin, Y., and Rumelhart, D. E., eds., *Back-propagation: Theory, Architectures and Applications*. Lawrence Erlbaum Publishers, Hillsdale, N.J. 433–486.