Online Probabilistic Goal Recognition over Nominal Models

Ramon Fraga Pereira1*, Mor Vered2, Felipe Meneguzzi1 and Miquel Ramírez2

1Pontifical Catholic University of Rio Grande do Sul, Brazil
2Monash University, Australia
3The University of Melbourne, Australia
ramon.pereira@edu.pucrs.br, mor.vered@monash.edu
felipe.meneguzzi@pucrs.br, miquel.ramirez@unimelb.edu.au

Abstract
This paper revisits probabilistic, model–based goal recognition to study the implications of the use of nominal models to estimate the posterior probability distribution over a finite set of hypothetical goals. Existing model–based approaches rely on expert knowledge to produce symbolic descriptions of the dynamic constraints domain objects are subject to, and these are assumed to produce correct predictions. We abandon this assumption to consider the use of nominal models that are learnt from observations on transitions of systems with unknown dynamics. Leveraging existing work on the acquisition of domain models via Deep Learning for Hybrid Planning we adapt and evaluate existing goal recognition approaches to analyse how prediction errors, inherent to system dynamics identification and model learning techniques have an impact over recognition error rates.

1 Introduction
Goal recognition consists of identifying the correct goal of an observed agent, given a model of the environment dynamics in which the agent operates and a sample of observations about its behaviour [Sukthankar et al., 2014]. Most approaches to goal recognition rely on carefully engineered domain models, often in the form of a plan library [Sukthankar et al., 2014, Chapter 1] or over a domain theory and a given set of hypothetical goals [Ramírez and Geffner, 2009; Ramírez and Geffner, 2010]. While much effort focuses on improving the recognition algorithms themselves [Martin et al., 2015; Pereira et al., 2017], recent research has looked critically into the availability and quality of the domain models [Pereira and Meneguzzi, 2018; Amado et al., 2018] used to drive such algorithms. In this paper, we explore how existing and suitable reformulations of well known approaches to goal recognition perform over domain models which have been obtained with the application of machine learning techniques to datasets consisting of transitions between states annotated with actions. For this, we leverage recent results on Planning [Say et al., 2017; Wu et al., 2017] that combine Deep Learning [Goodfellow et al., 2016] to obtain a nominal model [Ljung, 1998], and apply gradient–based optimisation [Calafiore and El-Ghaoui, 2014] algorithms using such models to compute plans.

This paper makes three main contributions. First, we propose a general framework for goal and plan recognition based on a computational model, Finite–Horizon Optimal Control (FHOC) problems [Bertsekas, 2017], that support continuous action spaces and uses arbitrary cost functions to define plans, offering more expressiveness that previous models [Baker et al., 2009; Ramírez and Geffner, 2010] yet remaining computationally feasible when optimal solutions are approximated. The two other contributions we make, discussed in Sections 4.1 and 4.2, reformulate two well–known recent approaches to goal recognition [Ramírez and Geffner, 2010; Vered et al., 2016; Kaminka et al., 2018] that seek alternative ways to enforce constraints on the transition function, which in our setting, is not directly accessible and represented with a “black box” neural network.

We evaluate the proposed recognition algorithms empirically in Section 5, using three benchmark domains based on the constrained Linear–Quadratic Regulator (LQR) problem [Bemporad et al., 2002], with increasing dimensions of state and action spaces. We build synthetic datasets for these domains and show that the first of the proposed algorithms performs quite well and infers the correct hidden goals when on the actual transition function of the domain, gracefully degrading over nominal models, due to the sometimes poor generalisation ability of the neural networks obtained.

2 Background
We now introduce the Finite–Horizon Optimal Control (FHOC) problems, whose solutions we use to model the range of possible agent behaviour, and discuss how we use Deep Neural Networks (DNNs) to approximate the dynamics constraints in FHOCs.

2.1 Finite Horizon Optimal Control Problems
We follow for the most part Bertsekas’ [2017] presentation of FHOC problems, incorporating some elements typically used by the literature on Control [Borrelli et al., 2017] and Planning [Bonet and Geffner, 2013] to account for constraints and

*This work was developed during an internship at The University of Melbourne under the supervision of Dr. Miquel Ramírez.
goals\textsuperscript{1}. Transitions between states are described by a stationary, discrete–time dynamical system
\begin{equation}
 x_{k+1} = f(x_k, u_k, w_k)
\end{equation}
where for each time point \( k \in [0, N] \), \( x_k \) is the state, \( u_k \) is the control input and \( w_k \) is a random variable with a probability distribution that does not depend on past \( w_j \), \( j < k \). For now, we make no further assumptions on the specific ways states, inputs and perturbations interact. States \( x_k \), controls \( u_k \), and disturbances \( w_k \) are required to be part of spaces \( S \subseteq \mathbb{R}^d \), \( C \subseteq \mathbb{R}^p \), and \( D \subseteq \mathbb{R}^{d \times p} \). Controls \( u_k \) are further required to belong to the set \( U(x_k) \subseteq C \), for each state \( x_k \) and time step \( k \). We note that the latter accounts for both the notion of preconditions and bounds on inputs. Observed agents seek to transform initial states \( x_0 \) into states \( x_N \) with both specific properties. These properties are given as logical formulas over the components of states \( x_k \), and the set of states \( S_G \subseteq S \) are those where the desired property \( G \), or goal, holds. The preferences of observed agents to pursue specific trajectories are accounted for with cost functions of the form
\begin{equation}
 J(x_0) = \mathbb{E}\{g(x_N) + \sum_{k=0}^{N-1} g(x_k, u_k, w_k)\}
\end{equation}
\( g(x_N) \) is the terminal cost, and \( g(x_k, u_k, w_k) \) is the stage cost. We define Finite Horizon Optimal Control problems as an optimisation problem whose solutions describe the range of possible optimal behaviours of observed agents
\begin{equation}
 \min_{\pi \in \Pi} \{ J_\pi (x_0) \} \tag{3}
\end{equation}
subject to
\begin{align}
 u_k &= \mu_k(x_k) \tag{4} \\
 x_{k+1} &= f(x_k, u_k, w_k) \tag{5} \\
 u_k &\in U(x_k), x_k \in S, x_N \in S_G \tag{6}
\end{align}
where \( \Pi \), the initial state, is an arbitrary element of \( S \). Solutions to Equations 3–6 are policies \( \pi \)
\begin{equation}
 \pi = \{\mu_0, \mu_1, \ldots, \mu_k, \ldots, \mu_{N-1}\}
\end{equation}
and \( \mu_k \) is a function mapping states \( x_k \) into controls \( u_k \) in \( C \). When \( \pi \) is such that \( \mu_i = \mu_j \) for every \( i, j \in [0, N] \), we say \( \pi \) is stationary. We note that terminal constraints \( x_N \in S_G \) can be dropped, replacing them by terms in \( g(x_k, u_k, w_k) \) that encode some measure of distance to \( S_G \). Costs \( g(x_N) \) are typically set to 0 when terminal constraints are enforced, yet this is a convention, and establishing preferences for specific states in \( S_G \) over others is perfectly possible.

\subsection{DNNs as Nominal Models}

Learning the dynamics or the transition between states of a domain model from data can be formalised as the problem of finding the parameters \( \theta \) for a function
\[ f(x_k, u_k; \theta) \]
that minimise a given loss function \( L(D, \theta) \) over a dataset \( D = \{ (x, u, y) \mid y = f(x, u, w), y \in S \} \).

\textsuperscript{1}Referred to as target regions in control.

reported on in this paper, we chose to use the procedure and neural architecture reported by Say et al. [2017] to acquire \( \hat{f} \), a DNN using Rectified Linear Units (ReLUs) [Nair and Hinton, 2010] as the activation function, given by \( h(x) = \max(x, 0) \). The DNN is densely connected, as recommended by [Say et al., 2017], so for a DNN consisting of \( L \) layers, \( \theta = (W, b) \), where \( W \in \mathbb{R}^{d \times d \times L} \) and \( b \in \mathbb{R}^{1 \times L} \). We use the loss function proposed by Say et al. [2017]
\[ \sum_j ||\hat{y}_j - y_j|| + \lambda \sum_l ||W_l||^2 \]
where \( \hat{y}_j = \hat{f}(x_j, u_j; \theta) \) and \( \lambda \) is a Tikhonov \( L^2 \) regularisation hyper-parameter [Goodfellow et al., 2016]. In the context of optimisation for Machine Learning, using this regularisation technique induces the optimisation algorithm to overestimate the variance of the dataset, so weights associated with unimportant directions of the gradient of \( L \) decay away during training. As also noted by Goodfellow et al. [2016], RELU networks represent very succinctly a number of linear approximation surfaces that is exponential in the number of layers \( L \) [Montufar et al., 2014]. This strongly suggests that RELU networks can displace Gaussian process estimation [Rasmussen and Williams, 2006] as a good initial choice to approximate complex non-linear stationary random processes, such as those in Equation 1, with the further advantage that, as demonstrated by [Yamaguchi and Atkeson, 2016; Say et al., 2017], DNNs can be directly used in Equation 5, so existing optimisation algorithms can be used off-the–shelf.

\section{Goal Recognition over Nominal Models}

Most existing approaches to goal and plan recognition assume that the domain model is both complete and correct, relying on what we call actual models. Such assumptions are widely considered by the literature on Control and Robotics [Mitrovic et al., 2010] to be too strong to hold in real-world, practical applications, where uncertainty on key model parameters is unknown, or these change over time due to wear and tear. We adopt this stance to define the task of goal and plan recognition over nominal models, that are estimated or recovered from past observed state transitions. These measurements of the underlying system dynamics can be obtained from observations on the behaviour of other agents, random excitation, or the simulation of plans and control trajectories derived from idealised models. We formally define the task of goal recognition over FOHCs and nominal models as follows.

\textbf{Definition 1 (Goal Recognition Problem).} A goal recognition problem over nominal models is given by an estimated transition function \( \hat{f}(x_k, u_k, w_k) \), s.t. \( \hat{f}(x_k, u_k, w_k) = \hat{x}_{k+1} \), where \( x_k \) is a state, \( u_k \) a control input, and \( w_k \) is a random variable; a cost function \( J \); an initial state \( I \) (i.e., an arbitrary element of \( S \)); a set of hypothetical goals \( G \); the hidden goal \( G^* \in G \); a sequence of observations \( O = \{o_1, o_2, \ldots, o_m\} \); and a horizon \( H \).

We further define the sequence of observations \( O \) to be a partially observed trajectory of states \( x \in S \) induced by a policy \( \pi \) that minimises \( J \). In general, a finite but indeterminate

\phantom{1}
number of intermediate states may be missing between any two observations \(o_i, o_{i+1} \in O\). We draw a distinction between online [Baker et al., 2009; Vered et al., 2016] and offline goal recognition, in which the former is a sequence of \(m\) Goal Recognition problems where \(O\) is obtained incrementally, while in the later \(O\) is available up front. We borrow the term judgement point from [Baker et al., 2009] to refer to the act of solving each of the \(m\) goal recognition problems that follow from the arrival of each new observation.

Informally, solving a goal recognition problem requires to select a candidate goal \(G \in \mathcal{G}\) such that \(\mathcal{G} = G^*\), on the basis of how well \(G\) predicts or explains the observation sequence \(O\) [Baker et al., 2009; Ramírez and Geffner, 2010]. Typically, this cannot be done exactly, but it is possible to produce a probability distribution over the goals \(G \in \mathcal{G}\) and \(O\), where the goals that best explain \(O\) are the most probable ones.

### 3.1 Probabilistic Goal Recognition

We follow Ramirez and Geffner [2010] (RG10) and adopt the modern probabilistic interpretation of Dennet’s principle of rationality [1983], the so-called Bayesian Theory of Mind, as introduced by a series of ground-breaking cognitive science studies by Baker et al. [2009][2014, Chapter 7]. RG10 set the probability distribution over \(G \in \mathcal{G}\) and \(O\) introduced above to be the Bayesian posterior conditional probability

\[
P(G|O) = \alpha P(O|G) P(G) \tag{7}
\]

where \(P(G)\) is the probability assigned a priori to goal \(G\), \(\alpha\) is a normalisation factor inversely proportional to the probability of \(O\), and \(P(O|G)\) is

\[
P(O|G) = \sum_{\pi} P(O|\pi) P(\pi|G) \tag{8}
\]

\(P(O|\pi)\) is the probability of obtaining \(O\) by executing a policy \(\pi\) and \(P(\pi|G)\) is the probability of an agent pursuing \(G\) to select \(\pi\). In the next section, we discuss two well-known existing approaches to approximate Equation 8, both reasoning over counterfactuals [Pearl, 2009] in different ways. These approaches frame in a probabilistic setting the so-called but-for test of causality [Halpern, 2016]. That is, if a candidate goal \(G\) is to be considered the cause for observations \(O\) to happen, evidence of \(G\) being necessary for \(O\) to happen is required. The changes to existing approaches are motivated by us wanting to retain the ability to compute counterfactual trajectories when transition functions cannot be directly manipulated.

### 4 Adapting Goal Recognition to Nominal Models

#### 4.1 Goal Recognition as Goal Mirroring

Mirroring [Vered et al., 2016] is an online goal recognition approach that works on both continuous and discrete domain models. For each of the candidate goals in \(\mathcal{G}\), Halpern’s but-for test is implemented by comparing two plans: an ideal plan and the observation-matching plan (\(O\)-plan). Ideal plans are optimal ones computed for every pairing of \(I\) and candidate goals \(G\), which are pre-computed before the recognition process starts. The \(O\)-plan is also computed for every pair \(I, G\), and it is required to visit every state in \(O\). \(O\)-plans are made of a prefix that results from concatenating the \(O\)-plans computed for previous judgement points [Baker et al., 2009], and a suffix, a plan computed from the last observed state to each candidate goal \(G\). The but-for test is implemented by making use of Theorem 7 in [Ramírez and Geffner, 2009], that amounts to consider a candidate \(G\) to be necessary for \(O\) to happen, if the cost of optimal plans and those consistent with the observation \(O\) are the same. Vered et al. [2016] show that \(O\)-plans are indeed consistent with \(O\) so Ramírez and Geffner’s results apply. The test was later cast in a probabilistic framework by Kaminka et al. [2018], with Equation 8 becoming

\[
P(O|G) = \left[1 + \epsilon(\pi_{O,G}, \pi_G)\right]^{-1} \tag{9}
\]

\(\epsilon(\pi_{O,G}, \pi_G)\) above is the matching error of \(\pi_G\), the ideal plan for \(G\) w.r.t. \(\pi_{O,G}\), the \(O\)-plan for the observations. Under the assumption that \(w\) is a random variable \(w \sim N(0, \sigma)\) with values given by a Gaussian distribution with mean 0 and standard deviation \(\sigma\), \(\epsilon\) can be used to account for the influence of \(w\) as long as \(\sigma\) remains an order of magnitude smaller than the values given by \(f(x, u, 0)\). Kaminka et al. [2018] define \(\epsilon\) as the sum of the squared errors between states in the trajectory of \(\pi_G\), and those found along the trajectory of \(\pi_{O,G}\). Under the second assumption that the selected ideal plans for a goal \(G\) are the most likely too, Kaminka et al. \(\epsilon\) is an unbiased estimator for the likelihood of \(\pi_{O,G}\).

Having established the suitability of Kaminka et al. means to bring about the but-for test to FHOCS, we now describe how we depart from their method to obtain \(O\)-plans \(\pi_{O,G}\). In this paper, we construct \(\pi_{O,G}\) by calling a planner once for each new observation \(o\) added to \(O\) and candidate goal \(G\), rather than just once per candidate goal \(G\) as Kaminka et al. proposed. Doing so allows us to “enforce” consistency with observations \(O\), since the couplings between states, inputs and perturbation in \(f(x, u, w)\) are no longer available so we can influence them with additional constraints, but are rather “hidden” in the network parameters. As the first observation \(o_1\) is obtained, we call a planner to solve Equations 3–6 setting the initial state \(x_0\) to \(I\) and \(x_{N} \rightarrow x_{o_1}\), the state embedded in \(o_1\). The resulting trajectory \(m_{O_1}^1\) is then used to initialise \(\pi_{O,G} = \langle m_1 \rangle\). We then invoke the planner again, this time setting \(x_0 = x_{o_1}\) and some suitably defined constraints s.t. \(x_{N} \in S_G\) for every candidate goal \(G\). The resulting trajectories \(m_{O_i}^1\) are used to define the \(O\)-plans \(\pi_{O,G} = \pi_{O,G} \oplus m_{O_i}^1\), which are compared with the pre-computed ideal plans \(\pi_G\) to evaluate \(P(O|G)\) as per Equation 9. As further observations \(o_i\), \(i > 1\), are received, we obtain trajectories \(m_{O_i}^1\) as above but setting \(x_0 = x_{o_{i-1}}\) and \(x_{N} \rightarrow x_{o_{i}}\), which are used to update \(\pi_{O,G}\) setting \(\pi_{O,G} = \pi_{O,G} \oplus m_{O_i}^1\). Trajectories \(m_{O_i}^1\) are obtained by setting initial states to \(x_{o_i}\), and concatenated to the updated \(\pi_{O,G}\) to obtain the \(O\)-plan for the \(i\)-th judgement point. We use as a baseline Vered et al.’s [2016] original mirroring algorithm, which requires \(|\mathcal{G}|\) calls to a planner per observation, and \(|\mathcal{G}| + (|\mathcal{G}| + |O|)\) calls overall. The approach above requires \(|\mathcal{G}| + 1\) calls to a planner per observation, and
be sets of \( r \) best trajectories \( t_i^+, t_i^- \) for either cost function, we introduce

\[
\Delta(O, G) = \frac{1}{r} \sum_{i=1}^{r} J_{\pi}^-(x_0; G) - J_{\pi}^+(x_0; G)
\]

where \( J_{\pi}^-(x_0; G) \) (resp. \( J_{\pi}^+(x_0; G) \)) is the result of evaluating the original cost function \( J_\pi(x_0; G) \) setting \( \pi \) to be the deterministic policy that follows from trajectories \( t_i^+ \) and \( t_i^- \).

We define the likelihood of \( O \) given \( G \) as RGI0 do

\[
P(O|G) = [1 + \exp(-\beta \Delta(O, G))]^{-1}
\]

with the proviso that \( \beta \) needs to be adjusted so as to be the inverse of the order of magnitude of \( \Delta(O, G) \). In comparison to the previous approach, our formulation of goal recognition over cost differences requires \( 2|G| \) calls per observation, and \( 2|G||O| \) calls overall. While all methods require a number of calls linear on \( |G||O| \), evaluating Equation 14 tends to be more expensive, as \( J^+ \) and \( J^- \) contain several non-linear terms and their derivatives are also costlier to compute. This is relevant as most if not all of the optimisation algorithms that we can use to solve Equation 3–6 are based on gradient-based techniques [Calafiore and El-Ghaoui, 2014].

5 Experiments and Evaluation

We now present the empirical evaluations we carried out on the algorithms proposed in the previous section. Sections 5.1 and 5.2 introduce the benchmark domains we used and describe how we generated the datasets for learning the transition function and the goal recognition tasks. Sections 5.3 and 5.4 report the quality of nominal models obtained and the performance of our goal recognition algorithms over both nominal and actual models.

5.1 Domains

For experiments and evaluation, we use three benchmark domains based on the constrained LQR problem [Bemporad et al., 2002], a general and well-understood class of optimal control problems with countless practical applications. Specifically, we use a discrete-time deterministic, linear dynamical system:

\[
x_{k+1} = Ax_k + Bu_k
\]

and trajectories must minimise the quadratic cost function

\[
J = x_N^TQx_N + \sum_{k=0}^{N-1} (x_k^TQ_kx_k + u_k^TRu_k^T)
\]

where \( Q \in \mathbb{R}^{d \times d} \) and \( R \in \mathbb{R}^{p \times p} \). All matrices are set to \( I \) of appropriate dimensions, but \( R \) which is set to \( 10^{-2}I \). Actions (inputs) \( u_k \) are subject to simple “box” constraints of the form \( lb(u) \leq u_k \leq ub(u) \). We note that the unconstrained LQR problem has an analytical solution [Bertsekas, 2017] as the cost function is globally convex and dynamics are linear, this simplifies the analysis of the behaviour of optimisation algorithms for training DNNs and computing trajectories. We consider two types of tasks. In the first domain, which we call 1D–Navigation, states \( x_k \in \mathbb{R}^2 \) represent the position

\[
|\mathcal{G}| + |\mathcal{O}| + (|\mathcal{G}||\mathcal{O}|) \text{ calls overall. This is a slight overhead which, on the basis of the results in Section 5, seems to be amortised enough by the accuracy and robustness of the new method.}

4.2 Goal Recognition Based on Cost Differences

We now present a novel goal recognition algorithm based on cost differences, inspired by RGI0’s probabilistic framework. RGI0 implement Halpern’s but-for test [2016] by determining whether plans exist that, while achieving \( G \), either guarantee that \( O \) happens, or prevent it from happening, the later being the counterfactual plan [Pearl, 2009]. When no such plans exist, a proof of \( G \) (not) being sufficient cause for \( O \) is obtained. Typically though, goals remain feasible, yet costs of plans change, making \( G \) less likely to be the cause of \( O \) whenever the cost of achieving \( G \) is lesser when \( O \) does not take place. We retain this definition of the test, yet we do not obtain such plans from the solution of a suitably modified version of \( f(x, u, w) \), as RGI0 does, by manipulating preconditions and effects. That is not possible in our setting, since couplings between state variables, actions and perturbation are not represented explicitly. Instead, we modify the cost function \( J \) by introducing artificial potential fields [Warren, 1989] centered on each observation in \( O \) that increase or decrease costs for valid trajectories.

Let \( J_\pi(x_0; G) \) be the cost functions for each of the hypothetical goals \( G \in \mathcal{G} \). For a given observation sequence \( O = (o_1, \ldots, o_n) \), we define cost functions

\[
J_{\pi}^+(x_0; G, O) = g(x_N) + \sum_{k=0}^{N-1} \left(g(x_k, \pi(x_k)) + \sum_{j=1}^{m} h(x_k, o_j)\right)
\]

\[
J_{\pi}^-(x_0; G, O) = g(x_N) + \sum_{k=0}^{N-1} \left(g(x_k, \pi(x_k)) - \sum_{j=1}^{m} h(x_k, o_j)\right)
\]

where the exponent is given as some suitably defined function over the difference of vectors \( x \) and \( o \). We note that both \( x \) and \( o \in \mathbb{R}^d \). For this paper, we have chosen the sum smooth abs functions

\[
\ell(u) = \sum_{i=1}^{d} \sqrt{u_i^2 + p^2} + p
\]

where \( u_i \) is the \( i \)-th component of the vector \( x - o \) and \( p \) is a parameter we set to 1. These functions have been reported by Tassa et al. [2012] to avoid numeric issues in trajectory optimisation over long horizons. The potential is used in Equation 10 to increase, w.r.t \( J_{\pi}^+(x_0; G) \), the cost of those trajectories that stay away from \( O \). Conversely, in Equation 11 it reduces the cost for trajectories that avoid \( O \). Let \( T^+ \) and \( T^- \)
and velocity of a particle, control inputs $u_k \in \mathbb{R}$ represent instant acceleration. Goal states require reaching a given position, yet leave terminal velocities unconstrained. The second domain, 2D Navigation, has higher dimensionality as states $x_k \in \mathbb{R}^{4n}$ represent position and velocities of $n$ vehicles on a plane, and control inputs $u_k \in \mathbb{R}^{2n}$ represent instant accelerations along the $x$ and $y$ axis. As in the previous domain, goal states only require reaching specific positions.

### 5.2 Learning and Recognition Datasets
To build the datasets and learn the system dynamics for the domains discussed previously, we generated 500 different navigation tasks i.e. pairs of states $x_0$ and $x_G$. We set the horizon $H = 100$, resulting in three different datasets with 50,000 transitions each. To generate the trajectories for each of the tasks, we first encoded the FHOCs for each of the domains using the RDDL domain description language [Sanner, 2011]. This enabled us to use the suite of hybrid planners recently proposed by Bueno et al. [2019], which we refer to as Tf-PLAN. These planners rely on state-of-the-art gradient-based optimisation techniques that obtain the gradients directly from the symbolic RDDL encoding, and have been shown to offer good performance on linear and non-linear dynamics with complex non-linear cost functions.

We use a benchmark consisting of 30 problems for 1D-Navigation and 30 each for 2D-Navigation setting $n$ to 1 and 2. We ensure the initial states and hypothetical goals are significantly different from the ones used to obtain the nominal models. Each task considers at most 5 goals hypotheses $G$, and observations $O$ comprise either 5 or 10 states, i.e., 5% and 10% of trajectories when $H = 100$. To generate $G$, we used Tf-PLAN to find a best trajectory from each initial state $I$ to the hidden goal $G^* \in G$. All states in the trajectories found have the same probability of appearing in any given $O$.

### 5.3 Learning Results
We use the approach of Say et al. [2017] to learn models of the system dynamics (transition function) of our domains from data. For the training stage, we configured the DNN proposed by [Say et al., 2017] to use the same hyper-parameters. Namely, 1 hidden layer, a batch size of 128 transitions, we set the learning rate to 0.01 and dropout rate to 0.1. Training was stopped for all domains after 300 epochs. We used exactly the same DNN configuration to learn the system dynamics for all of our domains. The Mean Squared Error (MSE) for the best DNN out of 10 trials, using 300 epochs for training, were $4.5 \cdot 10^{-5}$ for 1D-Navigation, $1.7 \cdot 10^{-4}$ for 2D-Navigation with $n = 1$, and $9.6 \cdot 10^{-6}$ for the same domain but $n = 2$. From these errors we conclude that using off-the-shelf the learning approach of Say et al. [2017] results in nominal models of very high quality, as judged by the loss function they propose.

### 5.4 Goal Recognition Results
We now show the experimental results of our recognition approaches over both actual and nominal models. For recognising goals over actual models we used the implementation of the Tf-PLAN planner used in [Bueno et al., 2019] that takes as input a domain model formalised in RDDL. For nominal models we used the implementation of Tf-PLAN in [Wu et al., 2017] that takes as input a domain model represented as a DNN. For both planners we set the learning rate to 0.01, batch size equals to 128, and the number of epochs to 300.

To evaluate our recognition approaches, we use metrics already used in the literature in goal recognition [Pereira et al., 2017; Pereira and Meneguzzi, 2018]. These are the average True Positive Rate (TPR) and average False Positive Rate (FPR). TPR is given by the number of true positives (1 when $G^*$ maximises $P(G|O)$, 0 otherwise) over the sum of true positives and false positives (the number of candidate goals maximising $P(G|O)$). A higher TPR indicates better performance, as it measures how often the true goal is calculated reliably. FPR is the average number of candidate goals $G \neq G^*$ which maximise $P(G|O)$, measuring how often goals other than the true one are found to be as good as or better explanation for $O$ than $G^*$. We also use the Top-k metric, typically used in machine learning to evaluate classifiers, setting $k$ to 2, to measure the frequency in which $G^*$ was amongst the top $k$ candidate goals as ranked by $P(G|O)$, and complements the two previous measures.

<table>
<thead>
<tr>
<th>Approach</th>
<th>$\eta$</th>
<th>$\Delta(G, O)$</th>
<th>Online TPR</th>
<th>Online FPR</th>
<th>Offline TPR</th>
<th>Offline FPR</th>
</tr>
</thead>
<tbody>
<tr>
<td>MIRRORING A 5</td>
<td>450.87</td>
<td>0.77</td>
<td>0.05</td>
<td>90.07</td>
<td>0.93</td>
<td>0.01</td>
</tr>
<tr>
<td>MIRRORING A 10</td>
<td>900.90</td>
<td>0.78</td>
<td>0.05</td>
<td>90.98</td>
<td>0.96</td>
<td>0.01</td>
</tr>
<tr>
<td>MIRRORING N 5</td>
<td>450.04</td>
<td>0.45</td>
<td>0.16</td>
<td>90.04</td>
<td>0.28</td>
<td>0.15</td>
</tr>
<tr>
<td>MIRRORING N 10</td>
<td>900.45</td>
<td>0.22</td>
<td>0.17</td>
<td>90.43</td>
<td>0.26</td>
<td>0.15</td>
</tr>
</tbody>
</table>

Table 1: Experimental results of our approaches over both actual and nominal models. $\eta$ represents the model type, $\%$ is observation level, and $N$ is the number of observed states. Note that the average number of goal hypothesis $|G|$ in the datasets is 5.

5 Results and Jupyter notebooks are available in https://github.com/authors-ijcai19-3244/ijcai19-paper3244-results
to the Top–2 measure, in 37% of the judgement points considered, $\Delta(O,G)$ is superior in 8.9% of the cases and both approaches are in agreement and correct in 42.1% of cases. This suggested to us that $\Delta(O,G)$ could be sensitive to one of the parameters used to calculate trajectories. Our ablation study detected that the number of epochs is the key parameter, as it directly affects how far from optimal are the trajectories found. We also observed that varying the number of epochs had counter-intuitive results, as the approximations to the optimal values of $J^+$ and $J^−$ do not get better or worse in a linear fashion. Instead, we observed costs improve (or worsen) for either cost functions at different rates, sometimes changing the sign of $\Delta(O,G)$. We ran the $\Delta(O,G)$ recognising over a limited number of instances, setting the number of epochs to 3,000 and we observed a significant improvement which brought it to be in agreement with $\pi$MIRRORING if not sometimes superior. Of course, this entailed an increase of run times by a roughly an order of magnitude. This leads us to conclude that the relatively good results of $\Delta(O,G)$ in Table 1 are due to the fact that $J^+$ and $J^−$ are closer to the convex ideal in Equation 16, as they include less non-linear terms $h(x, o)$, so TF-PLAN is less likely to get trapped in a local minima with adverse results for recognition accuracy early on.

6 Related Work

Most model–based approaches to goal and plan recognition rely on complete and accurate models of actions, transitions or other constraints. Pereira and Meneguzzi [2018] tackles the issue of model uncertainty, and enhances landmark–based heuristics to work with Weber and Bryce’s notion of incomplete models [2011], whereby uncertainty manifests in the preconditions and effects of the discrete planning operators. A transition function is still provided and fully accessible to the planner, yet in an incomplete form and possibly being incorrect, and this is indeed shown to have an impact on the accuracy of model–based goal recognition approaches.

Recent work such as that of Say et al. [2017] and Sanchez-Gonzalez et al. [2018] use DNNs [Goodfellow et al., 2016] to directly acquire models of dynamic constraints. These learnt constraints can then be readily used to formulate FHOC problems on top of them, which can be solved with a variety of dynamic programming or optimisation algorithms [Yamaguchi and Atkeson, 2016; Say et al., 2017; Wu et al., 2017; Bueno et al., 2019]. Compared to previous approaches combining dynamic programming and system identification [Mitrovic et al., 2010] based on Gaussian Process estimation [Rasmussen and Williams, 2006], DNNs have gradients which may be more straightforward to use for optimisation, yet show very quick rates of growth when it comes to their size.

The FHOC model we define in Section 2.1 is a generalization [Bertsekas, 2017] of the Markov Decision Processes (MDPs) used in seminal work on model–based goal and plan recognition [Baker et al., 2009]. As noted in Section 3, we too adopt Bayesian probabilities as the language to express the degree of certainty on causal relationships between goals $G$ and observations $O$. Our work separates from theirs in three respects. First, Baker et al. [2009] MDPs are defined over discrete action spaces, and we use continuous ones. Second, they assume observations $O$ to be prefixes of trajectories, while we allow intermediate states to be missing. Finally, we explicitly perform a check of sufficiency to determine causal relations between goals and observations [Halpern, 2016], by computing counterfactual trajectories from models. These we use to derive probability measures of certainty of causal relation, rather than requiring explicit distributions relating goals and observations as Baker et al. [2009] do.

7 Discussion

Model–based goal and plan recognition is a real–world, non–trivial and challenging application of causal reasoning, and this paper casts past approaches to model–based goal recognition as different implementations of Halpern’s but–for test of sufficient causality. We also show that machine learning systems can be used to generate predictions which are good enough to enable the generation of meaningful counterfactuals and by extension “true” causal reasoning, in contrast with recent statements to the contrary [Pearl, 2019].

We look forward to further investigating three questions this paper leaves open. First, to what extent the proposed algorithms can handle increasing variance for $\pi$. Informal tests on Linear–Quadratic Gaussian [Bertsekas, 2017] problems derived from the ones presented in this paper show promise but we have yet to test these observations for significance. Second, we need to determine whether it is possible to modify the loss function used for training the nominal models in a way that takes into account the accumulated error along trajectories, rather than just the errors in predicting the next state. Last, as discussed in Section 5.4, cost based goal recognition is very sensitive of planners converging to an unhelpful local minimum, which seems to us an inherent characteristic of stochastic optimisation algorithms. We look forward to evaluating our recognition algorithms over planners relying on Differential Dynamic Programming (DDP) [Mitrovic et al., 2010; Yamaguchi and Atkeson, 2016], which may converge faster to better local minima of the cost function.

We expect this paper to expand the applicability of model–based goal and plan recognition by replacing carefully engineered models for carefully curated datasets. This work is also an example of how to exploit latent synergies between Planning, Control, Optimisation and Machine Learning, as we mix together algorithms, techniques and concepts to address in novel ways a high–level, transversal problem relevant to many fields in Artificial Intelligence.

Acknowledgements

We want to thank Prof. Judea Pearl and Prof. Benjamin Recht for their lively exchanges on Twitter which have provided significant inspiration to write this paper. We also thank João Paulo Aires for the invaluable discussions about DNNs. This material is based upon work partially supported by the Australian DST Group, ID8332. This work is also financed by the Coordenação de Aperfeiçoamento de Pessoal de Nível Superior (Brazil, Finance Code 001). Felipe acknowledges support from CNPq under project numbers 407058/2018-4 and 305969/2016-1.
References


