Interpretable Policies from Formally-Specified Temporal Properties

Jonathan DeCastro¹, Karen Leung², Nikos Aréchiga³, Marco Pavone²

Abstract—We present an approach for interpreting parameterized policies based on a formally-specified abstract description of the importance of certain behaviors or observed outcomes of a policy. The standard way to deploy data-driven policies usually involves sampling from the set of outcomes produced by the policy. Our approach leverages parametric signal temporal logic (pSTL) formulas to construct an interpretable view on the modeling parameters via a sequence of variational inference problems; one to solve for the pSTL parameters and another to construct a new parameterization satisfying the specification. We perform clustering using a finite set of examples, either real or simulated, and combine computational graph learning and normalizing flows to form a relationship between these parameters and pSTL formulas either derived by hand or inferred from data. We illustrate the utility of our approach to model selection for validation of the safety properties of an autonomous driving system, using a learned generative model of the surrounding agents.

I. INTRODUCTION

Parameterized policies, widely prevalent in robot decision-making, motion prediction, and scenario-generation, are often black-box, having been learned from data. However, for a general data-driven policy, it is unclear how the choice of parameters will affect the outcome of the policy when it is deployed as an agent in an environment. A user of a policy, for instance, may wish to produce a style of behavior or evoke a certain outcome upon interacting with the environment. More specifically for the context of autonomous driving, the user may want to generate a suite of challenging driving scenarios in a simulation environment in order to stress test their newly developed autonomous driving policy and evaluate its performance. Thus, a challenge is in quantifying a relationship between parameters of data-driven policies and the emergent behaviors from deploying that policy.

For safety-critical systems, such as for autonomous cars, it is important to quantify the relationship between the parameters of a policy and the resulting behaviors of using that policy because it can (i) help with the verification of data-driven policies [1], [2] which is currently a bottleneck in the wide-spread adoption of learning-based components in safety-critical systems, (ii) provide interpretability and thus transparency which can potentially improve performance in quantifying a relationship between parameters of data-driven policies and the emergent behaviors from deploying that policy. Because more challenging, or rare, scenarios can now be known until placed in the context of a dataset of driver interactions. The process of finding such mapping, i.e., executing the policy and measuring the outcome, is typically expensive, even for simulation environments. Though having this mapping can widen the bottleneck in the verification and validation of robotic systems such as autonomous cars, because more challenging, or rare, scenarios can now be generated automatically.

To quantitatively characterize a relationship between parameters of a parameterized policy and resulting behaviors from running that policy in a computationally efficient way, we adopt variational inference to connect parameters from parameterized policies to parameters from parametric signal temporal logic (pSTL) formulas [5], [6]. Signal temporal logic (STL) is a temporal logic that is specified over dense-time real-valued signals, such as time-series data produced from continuous and hybrid systems. STL provides a con-

¹Toyota Research Institute, Cambridge, MA 02139, USA. jonathan.decastro@tri.global
²Department of Aeronautics and Astronautics, Stanford University, Stanford, CA 94305, USA. {karenl7, pavone}@stanford.edu
³Toyota Research Institute, Los Altos, CA 94022, USA. nikos.arechiga@tri.global

Fig. 1: Tailoring the learned driver behavior model according to outcomes as experienced by a vehicle behind it.

"Give me examples where two vehicles eventually come within X meters, and where the lead car experiences a sharp deceleration of at least Y"
cise language to construct specifications (i.e., formulas) that
describe relationships between the spatial (e.g., states of a
robotic system) and temporal properties of a signal. pSTL is
a parametric extension of STL where the parameters of the
STL formula are unknown and need to be determined from
input signals, and can be used as a form of feature extraction
for time-series data [7].

**Related work:** The problem of constructing interpretable
views and formally-specifiable models has ties to ongoing
work within machine learning and formal methods. In cre-
ating policy models that achieve a given objective, there
are several learning-based approaches exist. For instance,
variational autoencoders (VAE) and conditional VAE (cVAE)
have proven successful of multi-agent imitation learning
with a compact parameter representation exist, e.g. [8], [9].
Where there exists a well-defined terminal goal or outcome,
reinforcement learning techniques have been employed to
configure agents that achieve certain goals [10]. Such goals
can be made more expressive by leveraging temporal log-
ics [11], and symbolic abstractions that can be learned [12].
However, it is in general difficult to configure policies that
generate certain time-series properties through rewards and,
moreover, goal adherence often comes at the expense of
reconstruction (e.g., imitation learning).

Within the context of latent models, the concept of dis-
entanglement [13], [14] has become a popular means to
enforce a direct relationship between metrics and latent
variable models. Techniques such as [15] provide semantic
meaning in the latent structure using unsupervised learning.
The compactness of the representation and ability to seman-
tically cluster the behavior of a deep network model is useful
when a metric exists, but inferring such metrics from data
while respecting a user-defined logical structure remains a
challenge.

Bridging metrics with an expressive set of logical spec-
ifications is a well-studied area. In particular, [16] studies
computation graph learning over fixed predicates, and use
formula robustness, describing the degree to which a property
is satisfied, to guide the search of a decision tree that fits
features of a set of measured traces. The work of [17]
exploits logical structure of a problem to inform testing of
a given cyber-physical system. In [18], the authors propose
an approach to infer temporal logic formulas and impose a
tightness requirement to learned predicates that precisely fit
features of a corpus of time-series data. [19] introduces a
technique that allows construction of sparse formulas in a
semi-supervised manner where it is assumed an oracle can
provide positive or negative labels. The work of [20] use
semi-supervised manner where it is assumed an oracle can
provide a measure of

Within the context of latent models, the concept of dis-
entanglement [13], [14] has become a popular means to
enforce a direct relationship between metrics and latent
variable models. Techniques such as [15] provide semantic
meaning in the latent structure using unsupervised learning.
The compactness of the representation and ability to seman-
tically cluster the behavior of a deep network model is useful
when a metric exists, but inferring such metrics from data
while respecting a user-defined logical structure remains a
challenge.

Bridging metrics with an expressive set of logical spec-
ifications is a well-studied area. In particular, [16] studies
computation graph learning over fixed predicates, and use
formula robustness, describing the degree to which a property
is satisfied, to guide the search of a decision tree that fits
features of a set of measured traces. The work of [17]
exploits logical structure of a problem to inform testing of
a given cyber-physical system. In [18], the authors propose
an approach to infer temporal logic formulas and impose a
tightness requirement to learned predicates that precisely fit
features of a corpus of time-series data. [19] introduces a
technique that allows construction of sparse formulas in a
semi-supervised manner where it is assumed an oracle can
provide positive or negative labels. The work of [20] use
semi-supervised manner where it is assumed an oracle can
provide a measure of

Within the context of latent models, the concept of dis-
entanglement [13], [14] has become a popular means to
enforce a direct relationship between metrics and latent
variable models. Techniques such as [15] provide semantic
meaning in the latent structure using unsupervised learning.
The compactness of the representation and ability to seman-
tically cluster the behavior of a deep network model is useful
when a metric exists, but inferring such metrics from data
while respecting a user-defined logical structure remains a
challenge.

Bridging metrics with an expressive set of logical spec-
ifications is a well-studied area. In particular, [16] studies
computation graph learning over fixed predicates, and use
formula robustness, describing the degree to which a property
is satisfied, to guide the search of a decision tree that fits
features of a set of measured traces. The work of [17]
exploits logical structure of a problem to inform testing of
a given cyber-physical system. In [18], the authors propose
an approach to infer temporal logic formulas and impose a
tightness requirement to learned predicates that precisely fit
features of a corpus of time-series data. [19] introduces a
technique that allows construction of sparse formulas in a
semi-supervised manner where it is assumed an oracle can
provide positive or negative labels. The work of [20] use
semi-supervised manner where it is assumed an oracle can
provide a measure of

Within the context of latent models, the concept of dis-
entanglement [13], [14] has become a popular means to
enforce a direct relationship between metrics and latent
variable models. Techniques such as [15] provide semantic
meaning in the latent structure using unsupervised learning.
The compactness of the representation and ability to seman-
tically cluster the behavior of a deep network model is useful
when a metric exists, but inferring such metrics from data
while respecting a user-defined logical structure remains a
challenge.

Bridging metrics with an expressive set of logical spec-
ifications is a well-studied area. In particular, [16] studies
computation graph learning over fixed predicates, and use
formula robustness, describing the degree to which a property
is satisfied, to guide the search of a decision tree that fits
features of a set of measured traces. The work of [17]
exploits logical structure of a problem to inform testing of
a given cyber-physical system. In [18], the authors propose
an approach to infer temporal logic formulas and impose a
tightness requirement to learned predicates that precisely fit
features of a corpus of time-series data. [19] introduces a
technique that allows construction of sparse formulas in a
semi-supervised manner where it is assumed an oracle can
provide positive or negative labels. The work of [20] use
semi-supervised manner where it is assumed an oracle can
provide a measure of

Within the context of latent models, the concept of dis-
entanglement [13], [14] has become a popular means to
enforce a direct relationship between metrics and latent
variable models. Techniques such as [15] provide semantic
meaning in the latent structure using unsupervised learning.
The compactness of the representation and ability to seman-
tically cluster the behavior of a deep network model is useful
when a metric exists, but inferring such metrics from data
while respecting a user-defined logical structure remains a
challenge.

Bridging metrics with an expressive set of logical spec-
ifications is a well-studied area. In particular, [16] studies
computation graph learning over fixed predicates, and use
formula robustness, describing the degree to which a property
is satisfied, to guide the search of a decision tree that fits
features of a set of measured traces. The work of [17]
exploits logical structure of a problem to inform testing of
a given cyber-physical system. In [18], the authors propose
an approach to infer temporal logic formulas and impose a
tightness requirement to learned predicates that precisely fit
features of a corpus of time-series data. [19] introduces a
technique that allows construction of sparse formulas in a
semi-supervised manner where it is assumed an oracle can
provide positive or negative labels. The work of [20] use
semi-supervised manner where it is assumed an oracle can
provide a measure of

Within the context of latent models, the concept of dis-
entanglement [13], [14] has become a popular means to
enforce a direct relationship between metrics and latent
variable models. Techniques such as [15] provide semantic
meaning in the latent structure using unsupervised learning.
The compactness of the representation and ability to seman-
tically cluster the behavior of a deep network model is useful
when a metric exists, but inferring such metrics from data
while respecting a user-defined logical structure remains a
challenge.

Bridging metrics with an expressive set of logical spec-
ifications is a well-studied area. In particular, [16] studies
computation graph learning over fixed predicates, and use
formula robustness, describing the degree to which a property
is satisfied, to guide the search of a decision tree that fits
features of a set of measured traces. The work of [17]
exploits logical structure of a problem to inform testing of
a given cyber-physical system. In [18], the authors propose
an approach to infer temporal logic formulas and impose a
tightness requirement to learned predicates that precisely fit
features of a corpus of time-series data. [19] introduces a
technique that allows construction of sparse formulas in a
semi-supervised manner where it is assumed an oracle can
provide positive or negative labels. The work of [20] use
semi-supervised manner where it is assumed an oracle can
provide a measure of

Within the context of latent models, the concept of dis-
entanglement [13], [14] has become a popular means to
enforce a direct relationship between metrics and latent
variable models. Techniques such as [15] provide semantic
meaning in the latent structure using unsupervised learning.
The compactness of the representation and ability to seman-
tically cluster the behavior of a deep network model is useful
when a metric exists, but inferring such metrics from data
while respecting a user-defined logical structure remains a
challenge.

Bridging metrics with an expressive set of logical spec-
ifications is a well-studied area. In particular, [16] studies
computation graph learning over fixed predicates, and use
formula robustness, describing the degree to which a property
is satisfied, to guide the search of a decision tree that fits
features of a set of measured traces. The work of [17]
exploits logical structure of a problem to inform testing of
a given cyber-physical system. In [18], the authors propose
an approach to infer temporal logic formulas and impose a
tightness requirement to learned predicates that precisely fit
features of a corpus of time-series data. [19] introduces a
technique that allows construction of sparse formulas in a
semi-supervised manner where it is assumed an oracle can
provide positive or negative labels. The work of [20] use
semi-supervised manner where it is assumed an oracle can
provide a measure of

Within the context of latent models, the concept of dis-
entanglement [13], [14] has become a popular means to
enforce a direct relationship between metrics and latent
variable models. Techniques such as [15] provide semantic
meaning in the latent structure using unsupervised learning.
The compactness of the representation and ability to seman-
tically cluster the behavior of a deep network model is useful
when a metric exists, but inferring such metrics from data
while respecting a user-defined logical structure remains a
challenge.

Bridging metrics with an expressive set of logical spec-
ifications is a well-studied area. In particular, [16] studies
computation graph learning over fixed predicates, and use
formula robustness, describing the degree to which a property
is satisfied, to guide the search of a decision tree that fits
features of a set of measured traces. The work of [17]
exploits logical structure of a problem to inform testing of
a given cyber-physical system. In [18], the authors propose
an approach to infer temporal logic formulas and impose a
tightness requirement to learned predicates that precisely fit
features of a corpus of time-series data. [19] introduces a
technique that allows construction of sparse formulas in a
semi-supervised manner where it is assumed an oracle can
provide positive or negative labels. The work of [20] use
semi-supervised manner where it is assumed an oracle can
provide a measure of
the likelihood of a parameter given the prediction and the context of its environment is expected to be encountered, while the likelihood \( p(\alpha_t \mid X^t, \xi^t) \) informs the confidence of the prediction under the parameterization and environment. Often, we assume the conditional distributions to be Gaussian, with the mean and variance represented by the nonlinear mappings \( X^t \times Y^t \mapsto q_\theta \mathcal{U} \) and \( X^t \times Y^t \mapsto q_\sigma \mathcal{U} \), respectively, where \( q_\theta \) and \( q_\sigma \) are neural network weights. The latent representation is useful for sample efficiency and lends to interpretability when the dimension of \( z \) is small. While interpretation of \( z \) can often be found by clustering, e.g. according to semantics [9], the objective of this work is to make such policies more guided, more principled, and human-understandable.

As we detail in Sec. IV, while our formulation supports \( q(z \mid X^t, \xi^t) \) when available, it is not strictly required; e.g. analytically-derived policies such as ordinary differential equations parameterized by lookup tables. Also, note that policies on actions are not limiting. Without loss of generality, we may subsume dynamics and adopt the same approach to policies of the form \( p(\mathbf{x}_{t+1} \mid \xi^t, \xi^t, z) \), with \( \mathbf{x}_{t+1} \in \mathcal{X} \), the state at timestep \( t + 1 \).

B. Parameterizable Temporal Logics

To make the policies introduced above interpretable by a human, we introduce the parametric extension of the formal specification language signal temporal logic.

Definition II.1 (Parametric Signal Temporal Logic). Parametric STL (pSTL) formulas are defined recursively as follows:

\[ \varphi ::= \mu \mid \neg \varphi \mid \varphi \land \psi \mid \varphi \lor \psi \mid [\square_{[a,b]} \varphi] \mid [\diamond_{[a,b]} \varphi] \mid U_{[a,b]} \psi \]

where \( \mu \) is an atomic predicate, whose truth value is determined by the inequality \( \mu(s) < c \), for some parameter \( c \), and \( \varphi \) is an STL formula. We write \( \varphi_c \) to denote a pSTL formula parameterized by some vector \( c \). A trace \( \xi^T = (\xi^0, \xi^1, ..., \xi^T) \) satisfies \( \varphi \) if \( \xi^T \) satisfies \( \varphi \) at time \( t = 0 \). In other words, \( \xi^T \) satisfies \( [\square_{[a,b]} \varphi] \) if \( \varphi \) holds at every time step between \( a \) and \( b \), and \( \xi^T \) satisfies \( [\diamond_{[a,b]} \varphi] \) if, between time steps \( a \) and \( b \), \( \varphi \) is true at every time step up to some step when \( \psi \) holds. Additionally, let \( H_{[a,b]} \varphi_c := \mathcal{T}(U_{[a,b]} \varphi) \) such that \( \xi^T \) satisfies \( H_{[a,b]} \varphi_c \) if \( \varphi \) is true at some time step between \( a \) and \( b \).

The predicates are assumed to be of the form \( \mu(\xi^T) \gg c \), with \( \gg \in \{ >, = \} \). In this work, we adopt the quantitative semantics of pSTL, wherein, for a given formula \( \varphi_c \), each trace of length \( T \) admits a robustness value \( \rho_{\varphi_c} : \mathbb{R}^T \times \mathbb{R}^T \mapsto \mathbb{R} \), as follows.

- \( \rho_{\varphi_c}(\xi^T, t) \geq 0 \) if \( \xi^T \) satisfies \( \varphi_c \) at time \( t \), and
- \( \rho_{\varphi_c}(\xi^T, t) < 0 \) if \( \xi^T \) does not satisfy \( \varphi_c \) at time \( t \).

For a given STL formula, the robustness value is calculated by recursion on the parse tree of the formula as follows:

\[ \rho_{\varphi_c}(\xi^T, t) = \max(\rho_{\varphi_c}(\xi^T, t), \rho_{\psi_c}(\xi^T, t)) \]

\[ \rho_{[\square_{[a,b]} \varphi]}(\xi^T, t) = \max_{t' \in [t \cup a, t+b]} \rho_{\varphi_c}(\xi^T, t') \]

\[ \rho_{[\diamond_{[a,b]} \varphi]}(\xi^T, t) = \min_{t' \in [t \cup a, t+b]} \rho_{\varphi_c}(\xi^T, t') \]

\[ \rho_{U_{[a,b]} \psi_c}(\xi^T, t) = \max_{t' \in [t \cup a, t+b]} \min(\rho_{\psi_c}(\xi^T, t'), \min \rho_{\varphi_c}(\xi^T, t'')) \]

Definitions corresponding to the operators “\( \neg \)”, “\( \cap \)”, and “\( \Rightarrow \)” can be constructed accordingly. For brevity, we use the shorthand \( \rho_c(\xi^T) := \rho_{\varphi_c}(\xi^T, 0) \).

III. PROBLEM STATEMENT

We are concerned with instantiating parameterized policies that interact with an environment involving possibly many different agents. Due to the causal nature of the execution of a policy, it is not always clear what the outcome of a given parameterization will be. For instance, in the running example of Fig. 1, different behaviors and driving styles may emerge with different parameterizations of a given driving policy. However, there may be no easy way to select these parameters allowing for exploration of the spectrum between “easy” and “challenging” cases from the perspective of the vehicle behind. However, we may be able to observe interactions in data and infer the parameters of a formal specification to best match the observed outcomes in data corresponding to a user’s interpretation of “easy” or “challenging” cases.

More precisely, we posit that a pSTL specification \( \varphi_c \), referred to as a pSTL template, is both human-interpretable and able to encode desirable characteristics of a signal. Our goal is to: (1) learn parameters of \( \varphi_c \) yielding clusters of \( z \) that best fit the data, and (2) use the learned specification to construct an interpretable parameterization \( x \in \mathcal{Z} \) whose parameters are commensurate with satisfaction of \( \varphi_c \). A fixed policy can have several interpretations. As such, we call this approach an interpretable view on a parameterized policy.

IV. INTERPRETABLE VIEWS ON POLICIES

Given a pSTL formula parameterized by \( c \), we infer \( c \) under a given policy and data corpus. Consider a dataset of trajectories \( D = \{ \xi_t^0, ..., \xi_t^N \} \) produced by a policy in a particular multi-agent environment. Our goal is to find a set of parameters for a given pSTL template that best characterizes \( D \) with respect to the logical structure provided by the template. The approach is outlined in Fig. 2.

A. Variational Inference

Since the problem of inferring \( c \) is intractable to solve in general, we draw upon variational inference [22]. We discuss the case where a policy exists in latent variable form, i.e. we consider the problem of optimizing \( \rho(\alpha_t \mid \xi^t, z) \) and \( q(z \mid \xi^t) \) given (e.g. cVAE), then discuss the standard form where only \( p(\alpha_t \mid \xi^t, z) \) is given (e.g. \( z \) is hyperparameters).

Suppose we are given a distribution of a batch (vector) of modeling parameters \( z \) conditioned on a batch of trajectories \( \xi \), \( q(z \mid \xi) \). In our derivation, we are instead interested in
the joint distribution covering the parameterizations across
the entire dataset, i.e. a batch vector \( Z \in \mathbb{R}^{n_z \times |D|} \). We may
exploit independence so that \( q(Z \mid D) = \prod_t q(Z \mid \xi_t^D) \). Our
objective is to find an estimate \( q(c \mid D) \) of the true posterior
\( p(c \mid D) \) of pSTL parameters \( c \) via
\[
\mathbb{D}_KL[q(c \mid D) \parallel p(c \mid D)]
\]
\[
= \int q(c \mid D) \log \frac{q(c \mid D)p(D)}{p(c, D)} dc
\]
\[
= \int \mathbb{E}_{q(Z \mid D)} \log \frac{\mathbb{E}_{q(Z \mid D)} q(c \mid Z)p(D)}{p(D \mid c)p(c)} dc
\]
\[
\geq \mathbb{E}_{q(c \mid D)} \left[ \log \frac{\mathbb{E}_{q(Z \mid D)} q(c \mid Z)}{p(c)} - \log p(D \mid c) \right]
\]
\[
= \mathbb{D}_KL[\mathbb{E}_{q(Z \mid D)} q(c \mid Z) \parallel p(c)] - \mathbb{E}_{q(c \mid D)} \log p(D \mid c) \quad (1)
\]
where \( p(D) \) is the true (unknown) distribution of the data,
hence the introduction of the inequality.

The expression on the right hand side of the inequality is commonly referred to as the expectation lower bound, or ELBO. We assume the conditional distributions are normally distributed with mean and variance given by neural networks. The expectation appearing within the KL-divergence is approximated via Monte Carlo sampling of the batch-vectorized parameters \( Z \). Hence, \( q(Z \mid D) \) consists of a Monte Carlo approximation to the mean and variance. We generate the samples via the reparameterization trick using the provided mean and variance. We found that a standard Normal distribution for the prior \( p(c) \) yielded good results. The likelihood \( p(D \mid c) \) is discussed in the next section.

It is well-known that the evidence ELBO in (1) suffers from approximation errors, due to the Gaussian distribution assumed in the conditional distributions, we adopt normalizing flows to enrich the distributions. Normalizing flows are invertible, distribution-preserving transformations that enable representation of a rich set of distributions. In similar fashion to [23], we let \( c_t \) denote the output of the \( t \)th layer of the flow \( f_t \), and treat \( c_0 \) as the input and \( c = c_K \) as the final layer of a sequence of \( K \) flows.

Equation (1) may then then re-written as:
\[
\mathbb{D}_KL[\mathbb{E}_{q(Z \mid D)} q(c_0 \mid Z) \parallel p(c_K)] - \mathbb{E}_{q(c_0 \mid D)} \log p(D \mid c_K)
\]
\[
+ \mathbb{E}_{q(c_0 \mid D)} \left[ \log \prod_{t=1}^K \left[ \det \frac{\partial f_t}{\partial f_{t-1}} \right] \right] \quad (2)
\]
which gives an arbitrarily rich distributional representation of the ELBO. We choose inverse autoregressive flows (IAF) [24] within the inference scheme to improve the expressive power of each introduced transformation layer.

For policies not in variational form and lacking an encoder, one may train (2) directly on traces by replacing \( \mathbb{E}_{q(Z \mid D)} q(c_0 \mid Z) \) with \( q(c_0 \mid D) \) and using a recurrent structure similarly to standard trajectory-based cVAEs.

B. pSTL Semantics via Likelihood Tailoring

The likelihood term \( p(D \mid c) \) describes a criterion we wish to assert in order to establish concrete semantics. As with existing data clustering problems, there exist a number of possible approaches. Below we outline two approaches, which have commonly-used analogs in standard data analysis, for interpreting a given dataset.

- **Discriminative Clustering.** A discriminative clustering model seeks a description that best distinguishes between two types of data traces. In similar fashion to binary classification, the objective is to obtain a pSTL formula whose parameterization minimizes the mean square error of the robustness values across the population. To cast this problem into the probabilistic setting, we use a softmax likelihood, i.e.
\[
p(\xi_i \mid c) = \frac{\exp(-\rho^c_i(\xi_i)/\tau)}{\sum_c \exp(-\rho^c_i(\xi_i)/\tau)}
\]
where \( \tau \) is a softness parameter. The choice provides a semantic for best discriminator between satisfying and non-satisfying examples under the template.

- **Absolute Clustering.** This type of model seeks to cluster together common traits shared among a set of
solving an additional inference problem. Similar to [13], we
an interpretable space under the learned formula requires
modeling parameters \( \rho \) to a linear mapping to the robustness value
\( D \). Interpretable Views on Latent Encodings
the loss function to prevent degeneracy.
\[ L(z) = (1 + \rho_c(z))|\rho_c(z)| + \rho_c(z)^2 \] is a modified
hinge loss. That is, those data that satisfy the formula
are given priority over those that do not, and those
that do are made to be only marginally satisfied. The
loss improves as formulas are discovered that collect
more marginally-satisfiable data. Such models can give
a semantic for best marginal fit of satisfying examples
under the template.
In both cases, we assume that likelihoods are independent for
each trace, such that jointly we have
\[ p(D | c) = \prod_i p(\xi_i | c) \]. Notice that, given the same dataset, the semantics of each are different. The purpose of the discriminative model is
to form a logical relationship to the data that minimizes
the differences in robustness values over the population of
traces. The absolute clustering model, on the other hand, is
a regression problem; it constructs a logical relationship that
maximizes the number of marginally-robust examples.

C. Training Procedure
To capture the discrete nature of the formula learning
problem, we solve the variational inference approach using
backpropagation. As in previous works [25], [26], we use
the softmax/softmin trick in order to cast the pSTL formulas as
smooth, differentiable functions over the parameters \( c \). During training, we anneal both the softness parameter and
the weighting between reconstruction and KL divergence in
the loss function to prevent degeneracy.

D. Interpretable Views on Latent Encodings
Transforming the existing policy parameter space \( Z \) into
an interpretable space under the learned formula requires
solving an additional inference problem. Similar to [13], we
use distribution-preserving normalizing flows to construct a
mapping between \( z \) and \( \hat{z} \). We learn flows that map the
modeling parameters \( z \) to some space \( \hat{z} \), constrained to have
a linear mapping to the robustness value \( \rho_c(\xi) \) (see Fig. 2).

We choose the following objective:
\[
D_{KL}(q(z_0 | z, m) || p(z_{K'})) - E_{q(z_0 | z)} \log p(z | z_{K'})
- E_{q(z_0 | z)} \log p(m | z_{K'}) + E_{q(z_0 | z)} \left[ \log \prod_{t=1}^{K'} \left| \det \frac{\partial f_t}{\partial f_{t-1}} \right| \right]
\]
where we define the interpretability metric \( m = \tanh(\rho_c(z)) \), and where the mean and variance of \( p(m | z_{K'}) \) are fully-connected linear layers. We find planar flows to be sufficient to performing inference. For vector-valued
metrics, we wish to form linear relationships between each
element of the vector and individual elements of the new
parameter space by training individual, hence independent,
linear decoder layers for each metric.

It is useful to point out that solving (2) provides a
clustering of the policy parameter space according to the
formula, while the interpretable view in (3) provides a
selection mechanism for parameters, as depicted in Fig. 2.
Note that since the data we source is incomplete and possibly
changes with different interactions with the environment, the
parameterization is only locally valid in the neighborhood of
the data we train on.

V. CASE STUDY: AGENT BEHAVIOR MODELING
We examine the case of simulating driver behavior, where
the use case can be intent prediction, simulation, and vali-
dation of an autonomous ego car interacting with different
styles of drivers. We consider these agents to be human-
driven vehicles driving in tandem, as pictured in Fig. 1.

A. Agent Policy
We use the Argoverse dataset [27] to build a behavior
model of longitudinal behaviors, a policy taking in features
and conditioning variables and producing acceleration as an
output. We train a long short-term memory (LSTM) encoder and decoder as a cVAE with the position and velocity of the
car ahead treated as features \( x \), and the position and velocity
of the car behind treated as the conditioning variables \( y \). We
model the driver’s style using a two-dimensional latent vector
\( z \). The decoder LSTM transforms \( x, y, z \) into deterministic
acceleration commands. The future states are determined by
a point-mass model approximating the car’s physics.

In training the cVAE, we harvest data that only produces
in-lane examples where there exists a vehicle ahead. Since
the formula propositions require position, velocity and accel-
erations of both cars, we use a Kalman smoother to compute
the derivatives from positional data, before transforming each
trace into the lane-relative reference frame of the ego car.

B. Construction of a Policy Selection Mechanism
We next synthesize an interpretable view from a pSTL
formula to construct a new, compact parameterization where
TABLE I: Learned pSTL parameters for the cVAE.

<table>
<thead>
<tr>
<th>$c_{A_{\Delta x}}$</th>
<th>$c_{\Delta x}$</th>
<th>$c_{\xi}$</th>
<th>$c_{\Delta \dot{x}}$</th>
<th>$c_{\dot{x}}$</th>
<th>$c_{c}$</th>
</tr>
</thead>
<tbody>
<tr>
<td>91.36</td>
<td>-2.31</td>
<td>-3.08</td>
<td>-10.63</td>
<td>-3.75</td>
<td>0.1981</td>
</tr>
</tbody>
</table>

individual parameters are made to allow for adjustment of increasing or decreasing satisfaction of the formula. This can be useful, for instance, in validation of an autonomous driving system, where it is often important to stress-test the autonomy against more- or less-difficult agents. Our metric is constructed from a pair of STL declarative specifications defining outcomes where the interaction with the vehicle behind are deemed most or least severe:

Severe: $\mathcal{O}(\Delta x \leq c_{\Delta x}) \land \mathcal{O}(\Delta \dot{x} \leq c_{\Delta \dot{x}}) \land \mathcal{O}(\dot{x}_{lead} \leq c_{\xi})$

Benign: $\Box(\Delta x \geq c_{\Delta x}) \land \Box(\Delta \dot{x} \geq c_{\Delta \dot{x}}) \land \Box(\dot{x}_{lead} \geq \dot{x}_{c})$

where $\Delta x = x_{lead} - x_{trailing}$ is the distance between vehicles, $\Delta \dot{x} = \dot{x}_{lead} - \dot{x}_{trailing}$ is the relative speed, and $\dot{x}_{lead}$ is the lead vehicle’s acceleration. The first formula describes a preference for obtaining collisions and near-collisions that the driver in back could possibly have avoided. That is, at some point, the lead car had experienced sharp deceleration but at some other point there was a significant difference in speeds. The second formula describes the opposite behavior; those behaviors which we consider benign; i.e. cases where the distance, relative speeds, and accelerations are always kept high. Note that the benign formula is a stronger requirement than the negation of the severe formula, and hence will represent a well-defined, compact cluster of behaviors.

pSTL parameter learning: We train the pSTL parameters over the same dataset used to train the cVAE. To be able to form a regression from one formula to the other, we adopt the metric $m = \exp(\rho_{severe}(\xi)) - \exp(-\rho_{benign}(\xi))$, and assign $\hat{z}^0$ to this metric, reserving $\hat{z}^1$ to represent behaviors that have not been expressed by the formula.

We use the tool STLCLG for differentiable pSTL computation graph learning described in [25]. For the parameter inference model, we use a 16-dimensional encoder with three fully-connected layers and tanh activations. We use five IAF layers for the clustering model. To estimate the log-likelihood, we use 50 Monte Carlo samples within the inference network.

The learned pSTL parameters are shown in Table I. These are chosen as the empirical maximum likelihood values. In the plot of the original parameterization in Fig. 3, it is clear that there exists pockets of greater and lower severity of the model throughout the latent space.

Interpretable view learning: We construct an interpretable view using a composition of 15 planar flows to provide a two-dimensional mapping $\hat{z} = (\hat{z}^0, \hat{z}^1)^T \mapsto z$. For both the encoder and decoder for the latent variables, three 16-dimensional fully-connected tanh layers are used. In Fig. 4, the metric associated with the pair of formulas is seen to align with the $\hat{z}^0$-axis, producing a new parameterization that offers the progression of trajectories that span from benign to severe as the principal axis $\hat{z}^0$ is adjusted.

C. Discriminative Clustering

To illustrate the utility in discriminating between types of behavior of a given policy, we introduce a formula that is able to discriminate between different types of agent behavior that are interpreted as benign, while preserving the ability to easily progress from any of the benign cases to more severe behavior.

We base our discriminative model on acceleration:

$\Box(\dot{x}_{lead} \leq \dot{x}_{accel})$

That is, we wish to separate examples that have demonstrated the consistent deceleration from examples that may have accelerated at some point along the trace. We form an interpretable view using both the metric from Sec. V-B and the one learned here to form a parameter axis allowing progression from severe to benign with $\Box(\dot{x}_{lead} \leq \dot{x}_{accel})$ and, on the other axis, a progression from severe to benign with $\Box(\dot{x}_{lead} \leq \dot{x}_{accel}) = \mathcal{O}(\dot{x}_{lead} > \dot{x}_{accel})$. The resulting parameter set is two-dimensional $\hat{z} = (\hat{z}^0, \hat{z}^1)^T$.

The corresponding behaviors shown in Fig. 5 illustrate the resulting trajectories for each case. Clearly the two cases

Fig. 4: Left: the interpretable parameterization, where the parameter $\hat{z}^0$ clearly can be used to adjust the style of the behavior according to the severe-benign pair of formulas. Right: The progression of trajectories realized when adjusting $\hat{z}^0$. 
show different speed profiles, and these can be configured independently with $\tilde{z}^0$ and $\tilde{z}^1$.

VI. CONCLUSIONS

We address the problem of making policies more transparent and configurable to a user by constructing an interpretable view that follows a user’s formal specification, and learning a new parameterization that best matches outcomes revealed in a corpus of data. Our proposed approach is applicable to general data-driven policies where upon interacting with an environment, we are interested in the resulting spatial and temporal properties. As demonstrated with our case study, we find that this approach is particularly beneficial for scenario generation in the testing and validation of autonomous driving policies in simulation, but may also be useful for on-line intent prediction and social awareness.

Future work includes performing inference over the logical structure of formulas, and learning over integer-valued parameters. Currently, the challenge of learning formulas from data that are sufficiently expressive while remaining interpretable to humans is an interesting topic that should be further explored. Such tradeoffs are practically interesting in multi-agent settings, especially as the number of agents in the environment grows.

REFERENCES


