Proceedings of the 23rd International Conference on Principles of Knowledge Representation and Reasoning
KR meets Machine Learning and Explanation

Scalable Learning of Challenging Normative Behaviours with Deep RL

Emery Neufeld , Thorsten Engesser , Martin Tappler
TU Wien
{emeric.neufeld, martin.tappler}@tuwien.ac.at, thorsten@logic.at

Abstract

The acceptance of AI agents in daily life hinges on their alignment with social, moral, and legal norms, and in recent years, attempts to build norm-sensitive AI agents — including reinforcement learning (RL) agents — have gained traction. However, while there are many promising approaches, they tend to be geared toward environments with modest state spaces, and adaptations to the *deep RL* context are lacking. In this paper we present a deep learning adaptation of *normative restraining bolts* (NRBs). Our contributions are twofold; first we combine NRBs with deep Q-networks (DQNs) and proximal policy optimization (PPO) to learn optimal behaviour compliant with challenging norms in a complex environment. We demonstrate our agent's ability to learn difficult normative behaviours in the game Pac-Man, which has been used in the literature to benchmark normative RL techniques. Secondly, while past work has assumed a lexicographic ordering over conflicting norms when finding appropriate weights for norms, we discuss the shortcomings of this approach, and provide a more scalable alternative which allows for the selection of policies deemed ideal by more complex metrics.

1 Introduction

As artificial intelligence increasingly pervades daily life, the need for technologies that comply with the norms that govern our society becomes increasingly evident. Norms are rules defining ideal behaviour — often ethical ("you ought to help an injured person"), social ("you ought to stay in the right lane when not passing"), or legal ("you ought to stop at a red light"). It is not uncommon to find that they do not align with the optimal behaviour learned by machine learning approaches like reinforcement learning (RL) (Sutton and Barto 2018), where optimal behaviour is learned via trial and error, by collecting numerical rewards for good behaviour, and punishments for missteps. RL agents have proven effective in mastering a wide variety of tasks, and are a popular choice for constructing control policies in, e.g., robotics (Kober, Bagnell, and Peters 2013), the natural sciences (Degrave et al. 2022), and autonomous vehicles (Sallab et al. 2017), due to the technique's ability to learn efficient and even novel solutions to complex problems. RL from human feedback (Christiano et al. 2017) has paved the way toward effective language models (Stiennon et al. 2020), where the ethical dimension of learning from normative human judgments is clearly relevant. Nevertheless, RL agents prioritize high rewards above all else, so it is not uncommon for them to learn behaviour that violates human norms.

As the role of RL agents in our society evolves, so does the exploration of how RL interacts with machine ethics (Moor 2006; Anderson and Anderson 2007; Wallach and Allen 2008); this vein of study has yielded a growing number of techniques for imposing ethical behaviour on RL agents (Vishwanath, Dennis, and Slavkovik 2024). These include *normative restraining bolts* (NRBs) (Neufeld, Ciabattoni, and Tulcan 2024), which combine formal representations of norm-violating behaviour (namely, a collection of individual violation conditions expressed in linear temporal logic over finite traces (De Giacomo, Vardi, and others 2013)) with numerical punishments to disincentivize this behaviour. This combination allows the linking of precise and understandable representations of desired behaviour and a concrete means of influencing the actions of an RL agent. Neufeld, Ciabattoni, and Tulcan (2025) further developed this approach so that the punishment magnitudes required for enforcing a lexicographic ordering over norms could be algorithmically derived. However, this approach relies strongly on modest state and action spaces, and thus lacks scalability.

In this paper, we apply NRBs to deep Q-networks (Mnih et al. 2015) and proximal policy optimization (Schulman et al. 2017), two comparatively scalable RL approaches which use neural nets to estimate an optimal policy, allowing for applications over much larger and more complex state/action spaces. While the weight-finding algorithm from (Neufeld, Ciabattoni, and Tulcan 2025) cannot be used in this upscaled setting, we employ a hyperparameter optimization (HPO)-based approach that searches via samples for punishment magnitudes that yield a compliant policy. With the observation that the lexicographic ordering over norm priorities employed previously is a precarious choice in the more complex environments characteristic of deep learning problems[1], we propose user-defined *approximately ideal policies*, which can be selected from many possible poli-

[1]Often, in environments characterized by inherent risk of violation of high priority norms, a strict lexicographic approach can result in the agent completely ignoring lower priority norms and its primary objective, due to the inevitable risk of violating a high priority norm; even if the risk increases only slightly, the agent will opt to ignore everything but the high priority norm.

cies based on a tailored metric that can describe more complex conditions than a simple linear ordering. In this paper we make use of a metric based on a weakening of the lexicographic assumption of (Neufeld, Ciabattoni, and Tulcan 2025), employing the results of (Sherali and Soyster 1983). We empirically demonstrate how this approach leads to holistically improved performance.

We demonstrate our approach in the arcade game *Pac-Man*, which has been used to demonstrate the capability of norm-sensitive RL agents to obey norms in a complex, stochastic environment (Noothigattu et al. 2019; Neufeld et al. 2022; Neufeld 2022; Adam and Eiter 2025; Lopez-Miguel et al. 2025). Our experiments cover difficult normative dynamics which are not handled sufficiently by most past approaches, such as temporal obligations and conflicting norms, while accommodating the tackling of large state/action spaces with deep RL.

Related Work. Since (Abel, MacGlashan, and Littman 2016) there have been a variety of approaches presented whose goal is to impose specifically deontic constraints on RL agents, for example using automata (Kasenberg and Scheutz 2018b; Neufeld, Ciabattoni, and Tulcan 2024), inverse RL or demonstrations (Kasenberg and Scheutz 2018a; Wu and Lin 2018; Noothigattu et al. 2019), answer-set programming (ASP) (Adam and Eiter 2025), and multi-objective RL (Rodriguez-Soto et al. 2022; Rodriguez-Soto et al. 2023; Neufeld 2024; Neufeld, Ciabattoni, and Tulcan 2025). For a fairly extensive overview of some of these techniques, we recommend a recent review (Vishwanath, Dennis, and Slavkovik 2024).

However, these approaches are typically implemented within the tabular RL paradigm, where utilities are computed and stored in a table indexed by each state-action pair, greatly limiting the size of feasible state/action spaces. In order to accommodate large, complex state/action spaces, a deep RL technique (such as DQNs (Mnih et al. 2015) or PPO (Schulman et al. 2017)) is typically needed. Approaches attempting to impose norms on deep RL agents do exist (Chen et al. 2017; Peschl 2021; Peschl et al. 2022; Al Nahian et al. 2024); however, most of these focus on shaping behaviour with implicit or observed norms, rather than explicitly defined obligations, permissions, prohibitions, and priorities. With respect to deep RL approaches focusing on imposing explicit, formally-defined constraints — considered by many to be an important goal for responsible autonomous systems (Moor 2006; The IEEE Global Initiative on Ethics of Autonomous and Intelligent Systems 2019) for reasons of, e.g., transparency/auditability — a recent approach, (Xiao, Li, and Wang 2024), imposes an LTL specification on a deep RL agent. While similar in some ways to our technique, this approach rewards compliance instead of punishing violation (and thus is subject to the "procrastination" tendency illustrated by Neufeld, Ciabattoni, and Tulcan (2024)). Additionally, it utilizes a single LTL specification (and as a result could not handle conflicts and contrary-to-duty obligations (Neufeld, Bartocci, and Ciabattoni 2022)). While the original implementation of *policy fixes* (Adam and Eiter 2025) used tabular Q-learning, (Lopez-Miguel et al. 2025) demonstrated an application in deep RL as well as OFTEN-DEEPRL, which integrates them directly into deep RL policies. To frame the problem of minimally changing decisions to enforce norm compliance, i.e., fixing a policy, as a planning problem in ASP, both approaches require an abstract model of the environment, and they are restricted to maintenance obligations. To the best of our knowledge, there are no deep RL approaches supporting explicit representations of norms which can actually handle the difficult normative dynamics studied in the fields of deontic logic and normative reasoning, like *permissions/exceptions*, *contrary-to-duty obligations*, and *conflicts* between norms.

2 Preliminaries

2.1 LTL over Finite Traces

Linear Temporal Logic (LTL) (Pnueli 1977) extends classical propositional logic with temporal operators:

$$\varphi := p \mid \neg\varphi \mid \varphi \vee \varphi \mid \circ\varphi \mid \varphi \mathbf{U} \varphi$$

where $p \in AP$, AP being a set of atomic propositions.

$\circ$ is the "next" operator ($\circ\varphi$, or φ is true in the next state), and $\mathbf{U}$ is the "until" operator ($\varphi \mathbf{U} \psi$, or φ is true until ψ is true). We can also define the operator $\Diamond\varphi \equiv \top\mathbf{U}\varphi$ (φ is eventually true) and the operator $\Box\varphi \equiv \neg\Diamond\neg\varphi$ (φ is always true). $\top$, $\bot$, $\wedge$, and $\rightarrow$ can be derived in the usual way.

LTL formulas are interpreted over infinite traces $\sigma \in (2^{AP})^\omega$; LTL over finite traces (LTLf) (De Giacomo, Vardi, and others 2013) differs from LTL only in that the traces over formulae are interpreted are finite, of length λ.

Deterministic Finite Automata (DFAs). DFAs are finite state machines of the form $\mathcal{A} = \langle \Sigma, \mathcal{Q}, \delta, q_0, F \rangle$, consisting of a finite input alphabet Σ, a finite set of states $\mathcal{Q}$, a deterministic transition function $\delta : \mathcal{Q} \times \Sigma \rightarrow \mathcal{Q}$, an initial state $q_0 \in \mathcal{Q}$, and a set of final states $F \subseteq \mathcal{Q}$. We say $\mathcal{A}$ *accepts* a word $\sigma = \langle \sigma_0, ...\sigma_{\lambda-1} \rangle \in \Sigma^*$ if for the run $q_0, q_1, ..., q_\lambda$ generated by inputting σ to $\mathcal{A}$ (i.e., $q_{i+1} = \delta(q_i, \sigma_i)$ for $i \in \{0, ..., \lambda - 1\}$), $q_\lambda \in F$. LTLf formulae φ over atomic propositions AP can be transformed into equivalent DFAs $\mathcal{A}_\varphi$ with alphabet 2^{AP} such that $\sigma \models \varphi$ (that is, the trace σ satisfies φ) iff $\mathcal{A}_\varphi$ accepts σ (De Giacomo, Vardi, and others 2015).

2.2 Norms

We deal exclusively with regulative norms; i.e., norms that regulate behaviour (as opposed to, e.g., constitutive norms). Regulative norms include obligations $\mathbf{O}\alpha$ (or "α is obligatory"), prohibitions $\mathbf{F}\alpha \equiv \mathbf{O}\neg\alpha$ (we refer to obligations and prohibitions both as obligations), and (strong) permissions $\mathbf{P}\alpha$, which often act as exceptions to obligations to the contrary (and these are the only kind of permissions we consider here). Norms are often conditional, represented as $\mathbf{O}(\alpha|\beta)$ ("α is obligatory when β"); for consistency we will write unconditional norms $\mathbf{O}(\alpha)$ as $\mathbf{O}(\alpha|\top)$. We can also define temporal norms (Governatori et al. 2007) such as maintenance obligations $\mathbf{O}^M_\delta(\alpha|\beta)$ which demand that upon being

triggered with β, α is obligatory in each state until the deadline δ, and achievement obligations $\mathbf{O}^A_\delta(\alpha|\beta)$ which state that after the trigger β, it is obligatory to achieve α before δ.

Normative reasoning is a rich field in which many unique dynamics have been studied, many of them remaining open topics of study. For example, there are many different ways of handling conflicts between norms (Gabbay et al. 2021); in our implementation, we will assume a "fuzzy" preference over conflicting norms in which one norm generally takes priority over another, allowing that significant improvements compliance in lower priority norms can supersede marginal deterioration with respect to high priority norms.

Another dynamic that we model is *contrary-to-duty obligations* (CTDs) (Chisholm 1963), which come into force with the violation of another norm, and can prove difficult for e.g. safe RL approaches (Neufeld, Bartocci, and Ciabattoni 2022). Note that modelling both CTDs and conflicts necessitates some tolerance for violation in a formal language — this is why early straightforward modal logic approaches have failed (Gabbay et al. 2021).

We refer to a set of norms which apply in the same domain (and perhaps interact with each other) as a *normative system*. Information on which normative systems an agent is subject to is contained in what we call the agent's *norm base*.

Violation and Exception Specifications. LTL(f) struggles to model norms directly (Governatori 2015; Neufeld, Bartocci, and Ciabattoni 2022); namely, there is no LTL(f) formula φ that faithfully models an obligation, and LTL(f) on its own does not tolerate conflicts. From here we make use of *violation specifications*, which specify the conditions under which a norm is violated in LTL(f), to represent obligations, and special *exception specifications* to model permissions. This presentation of norms is inspired by the "Andersonian" approach (Anderson 1958) to modelling obligations, by representing a condition for violation and linking it to an individual (in our case numerical) sanction.

Using a temporal logic to implicitly represent norms enables the modelling of temporal obligations. Conditional atemporal obligations (punctual obligations) $\mathbf{O}(\alpha|\beta)$ can be represented through their violation conditions as $\Diamond(\beta \wedge \neg\alpha)$; the obligation is violated when β occurs, but α does not. Strong permissions $\mathbf{P}(\neg\alpha|\beta_1 \wedge \beta_2)$ (an exception to an obligation $\mathbf{O}(\alpha|\beta_1)$) are represented with an *exception specification* (giving the conditions under which the agent is exempted from a specific obligation) $\Diamond(\neg\alpha \wedge \beta_1 \wedge \beta_2)$. Maintenance obligations $\mathbf{O}^M_\delta(\alpha|\beta)$ have in general the violation specification $\Diamond(\beta \wedge \neg\delta\mathbf{U}\neg\alpha)$, and achievement obligations $\mathbf{O}^A_\delta(\alpha|\beta)$ can be modelled with the violation specification $\Diamond(\beta \wedge \neg\alpha\mathbf{U}\delta)$.

2.3 Reinforcement Learning

We assume that the environment in which the agent is learning is a labelled Markov decision process, which assigns labels to each state representing propositions which are true in that state. While this would be a strong assumption in many real-world problems, we argue that the below labelling function could be supplied through some additional technique, e.g., object detection from images. Our assumption of a labelling function enables separating the problems of norm-compliance and extraction of labels from, e.g., images.

Definition 1. *A* labelled MDP *is a tuple* $\langle S, A, Pr, L, R\rangle$ *where* S *is a set of* states*,* A *is a set of* actions*,* $Pr : S \times A \times S \to [0, 1]$ *models the* transition probabilities $Pr(s, a, s')$ *of reaching state* s' *by performing action* a *in* s*,* $L : S \to 2^{AP}$ *is a* labelling function *mapping states to sets of atomic propositions, and* $R : S \times A \times S \to \mathbb{R}$ *is a* reward function *assigning scalar rewards to state transitions.*

The goal of reinforcement learning (RL) is to find an optimal policy $\pi : S \to \Delta(A)$ mapping states to distributions over actions. An optimal policy maximizes the expected *discounted cumulative return* from time step t:

$$G_t = \sum_{i=0}^{\infty} \gamma^i R(s_{t+i}, a_{t+i}, s_{t+i+1}), \tag{1}$$

where $a_i \sim \pi(s_i)$ and $s_{i+1} \sim Pr(s_i, a_i, \cdot)$. The *discount factor* $\gamma \in [0, 1)$ enables prioritizing immediate rewards over future rewards, and ensures that G is finite.

Q-learning (Watkins and Dayan 1992) is a popular *model-free* RL technique (that is, a technique facilitating learning without an explicit model of the environment). It is based on approximating the action value function

$$Q^\pi(s, a) = \mathbb{E}_\pi \left[G_t \mid s_t = s, a_t = a \right],$$

which characterizes the expected return from state s, when taking action a and then following policy π.

The agent explores the environment based on an estimation of the optimal Q-function Q^* (assuming $Q^*(s, a) = Q^\pi(s, a)$ for optimal policy π). Each time it takes action a, transitioning from state s into state s' and receiving reward r, a so-called Q-learning update, based on the *Bellman equation*, is performed to improve the current state-action value estimates $Q(s, a)$:

$$Q_{target}(s, a) \leftarrow r + \gamma \max_{a' \in A} Q(s', a'),$$
$$Q(s, a) \leftarrow Q(s, a) + \alpha(Q_{target}(s, a) - Q(s, a)).$$

The *learning rate* α applied to the *TD error* $Q_{target}(s, a) - Q(s, a)$ balances learning speed and stability. The current values of $Q(s, a)$ are usually stored in a table.

Deep Q-Networks. Tabular Q-learning is not feasible for MDPs with large state spaces due to the necessity of storing Q-values for each individual state-action pair. A common way to address this issue is to estimate the Q-values using *function approximation* with, e.g., neural networks.

A fairly direct adaptation of Q-learning to the deep learning context is the well-known DQN algorithm from (Mnih et al. 2015), which approximates Q-values with *deep Q-networks*. These are feedforward neural networks where the inputs encode the MDP state s and the outputs are the estimated Q-values $Q^\theta(s, a)$ for each (discrete) action a. The network weights θ determine the shape of the Q-function and are adjusted during the training process.

The training process resembles tabular Q-learning. The agent selects actions based on the optimal action predicted by $Q^\theta(s,a)$ (using a strategy such as ϵ-greedy to balance exploitation and exploration) and regularly adjusts θ to account for recent experiences. In contrast to tabular Q-learning this is done by sampling from an extensive set of recently experienced transitions, which are stored in a *replay memory*. This technique, *experience replay*, helps to break the correlation between consecutive experiences and stabilizes training.

Proximal Policy Optimization. Q-learning and DQN are considered *value-based* reinforcement techniques; these stand in contrast to *policy-based* techniques. As the approach we put forward for this paper is based on reward shaping (i.e., changing the reward signal in the MDP to either guide learning or alter behaviour), it is highly flexible, and as we will show in the next section, it can be used on both value-based and policy-based (including actor–critic) techniques. In particular, we will make use of proximal policy optimization (PPO; Schulman et al. 2017) alongside DQN to demonstrate our approach.

PPO is a *policy gradient* method, which learns a *stochastic policy* π_θ. Instead of learning the action-value function Q^π, it directly optimizes the parameters θ of this policy. In particular, an objective function $J(\theta) = \mathbb{E}_\pi[G_0]$ is optimized, which can be estimated using a *policy gradient* (Sutton et al. 1999). PPO is a direct descendent of TRPO (Schulman et al. 2015), which uses a "surrogate objective" by averaging over several trajectories:

$$L^{CPI}(\theta) = \hat{\mathbb{E}}_t\left[r_t(\theta)\hat{A}_t\right],$$

where $r_t(\theta) = \frac{\pi_\theta(a_t|s_t)}{\pi_{\theta_{old}}(a_t|s_t)}$ and $\hat{A}_t$ is an *advantage function* which estimates the advantage of taking action a_t at state s_t (over the other possible actions). PPO replaces this objective with:

$$L^{CLIP}(\theta) = \hat{\mathbb{E}}_t\left[\min(r_t(\theta)\hat{A}_t, \text{clip}(r_t(\theta), 1-\epsilon, 1+\epsilon)\hat{A}_t)\right]$$

which limits the size of policy updates by *clipping* the size of $r_t(\theta)$ to the range $[1-\epsilon, 1+\epsilon]$.

Normative Restraining Bolts. A *restraining bolt* (RB) (De Giacomo et al. 2019) is a collection $\{(\varphi_i, r_i)\}_{i=1}^n$ of individual LTLf specifications φ_i paired with rewards r_i, along with a labelling function L. They can be combined with a labelled MDP $\mathcal{M}$ sharing its labelling function, to create a non-Markovian reward decision process (NMRDP) $\overline{\mathcal{M}}$ whose reward function adds to $R(s_{m-1}, a_{m-1}, s_m)$ (the reward from $\mathcal{M}$ for the transition to s_m) every r_i for which $\langle L(s_0), ..., L(s_m)\rangle \models \varphi_i$. As was established in (Brafman, De Giacomo, and Patrizi 2018), $\overline{\mathcal{M}}$ can be translated into a regular MDP by extending its state space to a product space with the state spaces of the DFAs $\mathcal{A}_{\varphi_i}$ corresponding to each φ_i. This *extended MDP* $\mathcal{M}'$ is *equivalent* (Bacchus, Boutilier, and Grove 1996) to $\overline{\mathcal{M}}$: when $\mathcal{M}'$ is solved with a regular RL algorithm such as Q-learning, the optimal policy is likewise optimal for $\overline{\mathcal{M}}$.

It is easy to see how this technique (essentially non-Markovian reward shaping) could be used to incentivize normative behaviour in RL agents; compliance specifications (logical formulae which specify the compliance conditions of a norm) are an established avenue for using temporal logics to indirectly represent norms (Kasenberg and Scheutz 2018b; Neufeld, Bartocci, and Ciabattoni 2022). Nevertheless, Neufeld, Ciabattoni, and Tulcan (2024) showed that this approach is not ideal, at least for the restraining bolt. The RB rewards the agent when a specification is satisfied; however, a compliance specification will be satisfied even when the norm is not in force (as technically, the agent is still complying with the norm), which can result in the agent trying to extend its runtime in order to reap rewards for compliance. To circumvent this issue, Neufeld, Ciabattoni, and Tulcan (2024) proposed that violation specifications (with corresponding negative rewards) should be used instead.

Definition 2 (Normative Restraining Bolt (NRB) (Neufeld, Ciabattoni, and Tulcan 2025))**.** *An NRB is a tuple* $NRB = \langle L, \{(\varphi_i, -r_i)\}_{i=1}^N\rangle$*, where* $L : S \to 2^{AP}$ *is a labelling function from states to sets of atomic propositions, each* $r_i \in \mathbb{R}^+$*, and* $N = n + m$*. For* $i \in \{1, ..., n\}$*,* φ_i *is the violation specification of a punctual or achievement obligation; for* $i \in \{n+1, ..., n+m\}$*,* φ_i *is the violation specification of a maintenance obligation.*

Like RBs, NRBs can be combined with a labelled MDP $\mathcal{M}$ to produce an NMRDP with an equivalent extended MDP $\mathcal{M}'$. Beyond the expansion of the state space S to $S' = \mathcal{Q}_1 \times ... \times \mathcal{Q}_N \times S$ (where $\mathcal{Q}_i$ is the state space of the automaton $\mathcal{A}_{\varphi_i}$), the only major differences between $\mathcal{M}$ and $\mathcal{M}'$ are the reward and transition functions.

$$R'(q_1, ..., q_N, s, a, q'_1, ..., q'_N, s') = R(s,a,s') + \sum_{i:\delta_i(q_i, L(s'))\in F_i} -r_i \quad (2)$$

is the reward function, with punishments instead of rewards; the transition function is:

$$Pr'(q_1, ..., q_N, s, a, q'_1, ..., q'_N, s') = \begin{cases} Pr(s,a,s') & \text{if } \forall i \in \{1, ..., N\} : q'_i = q'(q_i, s') \\ 0 & \text{otherwise} \end{cases}$$

where

$$q'(q_i, s') = \begin{cases} q'_i & \text{if } \delta_i(q_i, L(s')) = q'_i \notin F_i \\ q_{i,0} & \text{if } i \in \{1, ..., n\} \text{ and } \delta_i(q_i, L(s')) \in F_i \\ q_i & \text{otherwise} \end{cases}$$

The transition function is represented in two parts because the DFAs corresponding to different kinds of obligations require different kinds of "resetting" mechanisms to account for the possibility that a norm is violated more than once. When an obligation is violated, the associated automaton reaches a final state, which is also a sink state. Thus, to both avoid punishments after the violation has been rectified and allow for detecting future violations of the same norm, we "reset" the automaton; for punctual or achievement obligations, the state of the automaton, if it is final, skips without input to the initial state, while for maintenance obligations, it skips to the previous state (as maintenance obligations

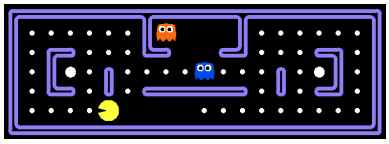


Figure 1: Our Pacman environment, a 5×18 maze with 55 food pellets, 2 ghosts, and 2 power pellets, allowing for a maximum score of 1795.

can be violated repeatedly whilst the obligation remains in force).

We maintain the single-objective framing of the problem of learning with normative restraining bolts from (Neufeld, Ciabattoni, and Tulcan 2024), as opposed to the multi-objective RL framing from (Neufeld, Ciabattoni, and Tulcan 2025). This enables the use of standardized deep RL libraries, which do not, to our knowledge, support multi-objective deep RL algorithms. However, we will refer to the method presented in the latter paper to algorithmically determine which punishments r_i are necessary to enforce a given priority ordering over the corresponding violation specifications, when we propose an alternative method to more feasibly approximate these punishments (which is necessary for the environment presented directly below).

2.4 Evaluation Environment: Pacman

Our environment is inspired by the classic arcade game *Pac-Man*, which takes place in a maze filled with 'food pellets'. The agent, Pacman, must navigate the maze and eat the food pellets it travels over (for 10 points each) using the actions $\{Stop, North, South, East, West\}$; the game is won when Pacman has eaten all the food pellets in the maze, earning 500 points. The goal is to win the game as quickly as possible, with a penalty of -1 point per step. Additionally, there are two ghosts wandering randomly around the maze, with the ability to 'eat' Pacman if it collides with them (in which case Pacman loses and collects -500 points); however, if the ghosts are in a temporary 'scared' state, which is triggered when Pacman eats a special 'power pellet', they can be eaten by Pacman, earning Pacman 200 extra points. The exact layout of the maze we are using is shown in Fig. 1.

Inspired by Noothigattu et al.; Neufeld et al. (2019; 2022), we utilize this setting to train and test the behaviour of Pacman under a variety of normative systems. We start with VEGAN, a very basic norm base containing only the norm

$$vegan: \ \mathbf{F}(eat_{blueGhost} \vee eat_{orangeGhost} | \top)$$

corresponding to violation specification $\Diamond(eat_{blueGhost} \vee eat_{orangeGhost})$; that is, the obligation is violated (once) if the agents eats a blue or orange ghost at some point. We then introduction the strong permission

$$permBlue: \ \mathbf{P}(eat_{blueGhost} | \top),$$

which explicitly allows Pacman to eat a blue ghost. Together with *vegan*, *permBlue* makes up the VEGETARIAN norm base. We further employ an achievement obligation

$$hungry: \ \mathbf{O}^{A}_{score100}(eat_{blueGhost} \vee eat_{orangeGhost} | score0)$$

	vegan	*permBlue* (permission)	*hungry* (temporal)	*penalty* (CTD)
VEGAN	✓			
VEGETARIAN	✓	✓		
HUNGRY			✓	
HUNGRYVEGAN	✓		✓	
HUNGRYVEGETARIAN	✓	✓	✓	
HUNGRYPENALTYVEGAN	✓		✓	✓

Table 1: Overview of Pac-Man norm bases.

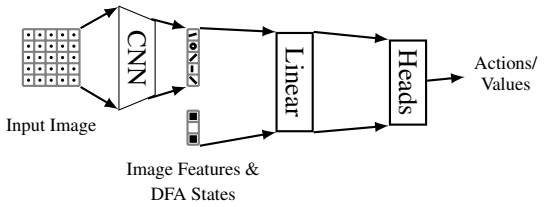


Figure 2: Architecture for deep RL with NRBs, with heads corresponding to a Q-network (DQN) or an actor-critic pair (PPO).

which demands that starting when the game begins at score 0, Pacman is required to eat a ghost before it reaches a score of 100. We use this norm individually in the HUNGRY norm base and conflicting with $vegan$ in HUNGRYVEGAN. HUNGRYVEGETARIAN contains *vegan*, *permBlue*, and *hungry*. Finally, we introduce the CTD obligation

$$penalty: \ \mathbf{O}(stop | eat_{blueGhost} \vee eat_{orangeGhost})$$

which obligates the agent to give up a turn (perform the action $stop$) after violating $vegan$. This occurs with $hungry$ and $vegan$ in HUNGRYPENALTYVEGAN. A summary of the norm bases we employ can be found in Table 1.

3 Shaping Deep RL Policies with NRBs

As we discussed in Sec. 2.3, a labelled MDP $\mathcal{M}$ can be combined with an NRB $\langle L, \{(\varphi_i, -r_i)\}_{i=1}^{N} \rangle$ and translated to an extended MDP $\mathcal{M}'$, for which we can learn an optimal policy with, e.g., Q-learning. However, tabular RL already struggles with large state spaces, and the state space S' of $\mathcal{M}'$ exceeds $|S|$ by a factor that is exponential in the number of norms imposed, presenting an obstacle for implementation in large environments with many norms. A natural solution is to combine NRBs with deep RL, via value-based (e.g., DQN) or policy-based (e.g., PPO) methods.

The input to the deep RL agent is comprised of: (1) d features that represent a state s of the environment MDP. We denote these features as $\{\phi_i(s)\}_{i=1}^{d}$ (in the Pacman setting, this includes the location of the nearest food, the locations of the ghosts, the score, etc.); and (2) each state of each automaton corresponding to a specification in the restraining bolt (i.e., $\bigcup_{1 \leq i \leq N} \mathcal{Q}_{\mathcal{A}_{\varphi_i}}$) as one-hot encoded features, which have a value of 1 when it is a current state, and 0 otherwise. Thus, the set of features for the extended MDP state $(q_1, ..., q_N, s)$ is $\{\phi_i(s)\}_{i=1}^{d} \cup \Big(\bigcup_{1 \leq i \leq N} \{[q_i = q]\}_{q \in \mathcal{Q}_{\mathcal{A}_{\varphi_i}}} \Big)$, where $[\cdot]$ are Iverson brackets. The reward corresponding to each transition of $\mathcal{M}'$ can be computed with Eq. 2.

In a second approach, we follow (Mnih et al. 2015) in processing raw image data from the *Pac-Man* game with three layers of 2D convolutional neural networks (CNNs). To use NRBs with image input, we concatenate the flattened features, extracted using the CNNs, with one-hot encoded DFA states, and pass both through two additional linear layers, as depicted in Fig. 2. Finally, the resulting features are processed by an output layer to estimate Q-values (in the case of DQN) or actor and critic heads (in the case of PPO).

3.1 Approximately Ideal Policies

(Neufeld, Ciabattoni, and Tulcan 2025) resolve norm conflicts under a given strict lexicographic ordering, where the primary objective is always in last place. The multi-objective framing manifested in a Q-function being learned for each norm; ergo, following Rodriguez-Soto et al. (2022; 2023), *convex hull value iteration* (CHVI) (Barrett and Narayanan 2008) can be used to compute the *convex hull* of the multi-objective MDP's policies (the policies optimal for some combination of weights/penalties r_i). Linear programming can compute from the convex hull the required weights r_i for enforcing the given lexicographic ordering over the value functions. Unfortunately, CHVI itself is computationally expensive, even over modest state spaces[2]; in the Pac-Man environment, it would be infeasible.

Size is not the only limiting factor; the stochasticity of an environment might be prohibitive as well. For instance, consider two policies for a self-driving car; one involves reaching the user's destination, and one consists of ignoring the user's commands and staying in the garage. If our highest priority norm is to avoid collisions, then a strict lexicographic ordering will result in the latter policy. Whenever the car leaves the garage, there exists some possibility of a collision; even a small probability of collision will result in the car staying parked if the probability of collisions in the garage is zero. In real-life, there has to be *some* margin for error when it comes to prioritizing norms, and in general, the desired policy for a nuanced and difficult problem might not be so straightforwardly reduced to a strict linear ordering.

With these challenges in mind, we propose that instead of *deriving* weights, we *search* for them. It established that any policy learned with NRBs is in the convex hull (Neufeld, Ciabattoni, and Tulcan 2025); and while using CHVI to compute *all* of these policies is not feasible in our setting, we can compute *some* of them, and observe their characteristics by sampling the distribution induced by the policy on the MDP. Namely, we can sample combinations of weights from some subset of $\mathbb{R}^N$, compute the policy optimal for that combination, and evaluate the resulting behaviour w.r.t. our expectations, for example by scoring with some pre-defined metric ζ. We are then interested in *approximately ideal policies* (AIP), which yield trajectories that optimize for a specific "scoring" function $\zeta : 2^{\mathcal{T}} \to \mathbb{R}$:

$$\pi^*_{AIP} = \arg\max_{\pi \in \Pi} \zeta(\{\tau\}_{\tau \sim \pi}) \tag{3}$$

[2]Around $O(|A|^{|S'|})$; and despite the relatively small size of the environment in (Neufeld, Ciabattoni, and Tulcan 2025), it was reported that up to 256 GB of RAM were necessary to find weights for two norms.

ζ is composed of a number of auxiliary feature functions $f_i(\tau)$, e.g. the average score of a game after several episodes. In the context of learning with NRBs, we are specially interested in ζ which are maximized by policies in the convex hull.

Our proposal, then, is to search for the weights yielding our AIP via *hyperparameter optimization* (HPO); we consider each r_i for the NRB a hyperparameter in our learning problem, and we use the multivariate TPE algorithm (Bergstra et al. 2011) via Optuna (Akiba et al. 2019) to implement the hyperparameter search. The objective of the HPO will be the tailored metric ζ, which will, for our purposes (learning with NRBs), incorporate e.g. the norm violation count over a trajectory. In particular, we consider the normalized violation count: $f_i^{viol} : \mathcal{T} \to \mathbb{R}$ for $i \in \{1, ..., N\}$:

$$f_i^{viol}(\tau) = \frac{maxViol - viol_{\varphi_i}(\tau)}{maxViol}$$

where $maxViol$ is the maximum possible violations in a trajectory, and $viol_{\varphi_i}(\tau)$ is the actual number of violations of the norm with violation specification φ_i over τ; thus, $f_i^{viol}(\tau)$ is 1 when there are no violations, and approaches 0 as more violations occur.

Our Approach. In this paper, we do not stray far from a strict lexicographic ordering. We make two amendments to this approach, however: first, we non-linearly rank norms so that two norms can share the same rank. Additionally, we *weaken* the strict ordering; e.g, if a higher priority norm is violated only slightly more for a given policy, but there is a massive drop in violations for a lower priority norm, we will typically prefer this policy over one where violations of the higher priority norm decrease only slightly, while there is a sharp increase in violations of the lower priority norm.

As for how we achieve this with our ζ: it is known that for some set of lexicographically ordered objectives, given there exists a solution that respects this ordering, there is a linear combination of these objectives whose global optimum is that solution (Sherali and Soyster 1983). Namely, for some lexicographically optimal solution $x^* = [x_0, ..., x_k] \in X$ (where x_0 corresponds to the lowest priority objective, and x_k to the highest), there is a $b_0 \in \mathbb{R}$ such that for all $b \geq b_0$,

$$x^* = \arg\max_{x \in X} \sum_{i=0}^{k} b^i x_i.$$

As we noted, however, we intend to depart from this strict lexicographic ordering. Firstly, instead of a strict linear ordering over objectives, we define a function over each objective x_i that maps the ith objective to some natural number $w(i)$ (its place in the ordering), and assign to each x_i the coefficient $b^{w(i)}$ in the above sum, allowing us to account for objectives x_i and x_j of equal weight (i.e., $w(i) = w(j)$). A second change is to choose a relatively small $b \geq 1$ (significantly smaller than the above-mentioned b_0) in order to "weaken" the lexicographic ordering; this results in the lower priority objectives having a much more comparable weight with respect to higher priority objectives, so

that a large improvement in lower priority violations can be achieved at the expense of a small increase in high priority violations. Our ζ is then

$$\zeta(\{\tau\}_{\tau\sim\pi}) = \hat{\mathbb{E}}_{\tau\sim\pi}\left[\frac{G_0(\tau)}{r_{max}}\right] + \sum_{i=1}^{N} b^{w(i)} \cdot \hat{\mathbb{E}}_{\tau\sim\pi}\left[f_i^{viol}(\tau)\right]$$

where r_{max} is the highest possible reward in the environment. Thus, as b increases, compliance with the highest priority norms become more important, when compared to the agent's reward function or to compliance with less important norms, approaching a lexicographic ordering. However, as b approaches 1, the normalized violations and score become more equally considered in the selection of the AIP. To demonstrate how this acts in practise, our evaluation contains a sensitivity analysis for the parameter b.

4 Experimental Evaluation

We conducted experiments in the Pac-Man environment (DeNero and Klein 2014) shown in Fig. 1, where we trained 7 types of RL agents with PPO and DQN with 2 types of state representations. The 7 agent types range over the six norm bases from Table 1 and a baseline agent, which is not guided by NRBs. The 2 state representations include a *feature-based* state representation, where each state comprises 83 features of the game state and one-hot-encoded representation of the DFA states, and an *image-based* state representation augmented with DFA state information. To create the RL state for the image-based representation, we convert the images to grayscale and downsample them by a factor of three in both height and width, and we then stack two consecutive frames to capture motion. For the training, we used the default hyperparameters of the DQN and PPO implementations provided by STABLE-BASELINES3 (Raffin et al. 2021), except for a reduced buffer size of 100,000 in image-based RL due to memory constraints. Our code and models, as well as some videos, can be found at https://github.com/lexeree/scalable-normative-deep-rl.

Our experiments examine (1) the effect of RL algorithm and required training budget on different norm bases, (2) the viability of HPO for finding suitable NRB weights, (3) the effectiveness of NRBs in combination with image-based state representations, and (4) the effectiveness of NRBs combined with deep RL compared to the state of the art.

To address (1), we create RL policies for the feature-based state representation with varying training budgets for both DQN and PPO. In this first batch of experiments, we manually defined weights for each norm base. For (2), we fixed the training budgets to suitable values found in the training budget experiments and used HPO to identify optimal NRB weights. (3) We then evaluate policies trained from feature-based representations and images. Finally, (4) we compare our results to those reported in the literature.

4.1 Determining Training Budgets

We fixed preliminary weights for the NRBs and ran 12 trials per norm base to determine how many training steps (we tested up to 20 million) are needed to converge to a decent policy. Detailed results can be found in the supplementary material contained in our GitHub repository.

In particular, we observed that for the VEGAN and VEGETARIAN norm bases, DQN performs slightly better than PPO: 2.5 million steps are sufficient for VEGAN, while VEGETARIAN benefits from 5 or even 10 million steps. For the achievement obligation HUNGRY, PPO performed better than DQN with the same training budget, reaching good performance at 10 million steps (vs. 20 million for DQN). Similarly, for HUNGRYVEGAN and HUNGRYPENALTYVEGAN, PPO showed good convergence after 20 million steps, with DQN performing significantly worse. While we observed advantages of PPO for HUNGRYVEGETARIAN with lower numbers of steps, DQN catches up at 20 million steps.

4.2 Optimizing the NRB Weights

Using the optimal algorithm and number of training steps identified above, we then ran a hyperparameter optimization for each norm base to identify appropriate punishments.

We used a total of 100 HPO trials per norm base, with weights sampled from the $[50, 5000]$ range. For each trial, we averaged the score and norm violations over five policies[3] to estimate the metric ζ (with $b = 10$). Each policy was evaluated for 1000 episodes. Our results for VEGAN and HUNGRYVEGAN are visualized in Figure 3.

In particular, we can see that for the VEGAN norm base, as the punishment increases, the number of violations improves while the Pac-Man score worsens. Both curves reach a plateau where norm compliance is near-optimal, and the score appears as high as possible given that compliance.

For HUNGRYVEGAN, we can see that larger punishments for *hungry* generally lead to fewer *hungry* violations, and likewise larger punishments for *vegan* lead to fewer *vegan* violations. However, setting the *vegan* punishment too high can adversely affect the number of *hungry* violations. The score is negatively impacted by both large *hungry* and *vegan* punishments. The identified optimal weight vector strikes a balance by keeping both violation rates low while maintaining a high score. Results for the other norm bases can be found in our GitHub repository and show similar tradeoffs.

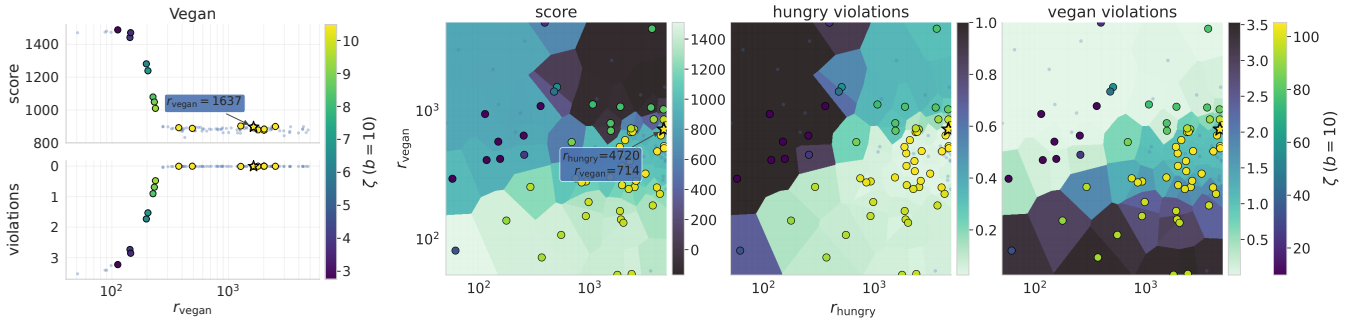
To further validate our approach, we also returned to the "merchant environment" of (Neufeld, Ciabattoni, and Tulcan 2024; Neufeld, Ciabattoni, and Tulcan 2025), and used HPO to find appropriate weights for a regular tabular Q-learning agent. We arrived at different weights (as the approach presented in (Neufeld, Ciabattoni, and Tulcan 2025) specifically looks for the *minimal* weights that will result in the desired policy), but in the end learned the same policy. Details can be found in our repository.

4.3 Sensitivity Analysis of b

The direction in which the optimization progresses, as well as the weights that are ultimately selected, depend on our choice of the parameter b b for the metric ζ.

We provide a brief sensitivity analysis using the HUNGRYVEGETARIAN norm base. In Figure 4, we plot for each

[3]Training a single policy with 20 million steps (DQN/PPO with high-level features) takes around 5 hours on a single Zen 5 core.



(a) VEGAN (b) HUNGRYVEGAN

Figure 3: Estimated performance (Pac-Man score and violation count per episode) by norm penalty weights (r_{vegan} and r_{hungry}) for the single-norm base VEGAN and the dual-norm base HUNGRYVEGAN. Pareto-optimal samples are highlighted and coloured according to our optimization metric ζ, for $b = 10$. The optimal sample according to ζ is marked by a star. For the dual-norm base, each Voronoi cell is coloured according to the score or violation count of its closest sample (lighter colours are better).

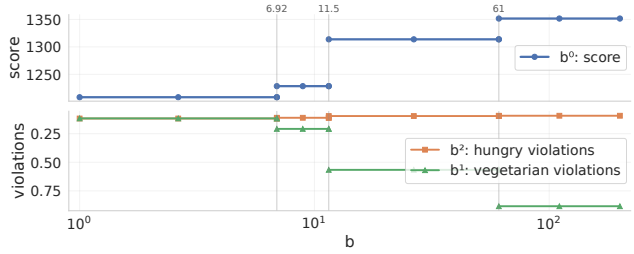


Figure 4: Projected performance of the best found weights for all $b \geq 1$ in HUNGRYVEGETARIAN. Note that while we optimized for a fixed value of $b = 10$, the performance of any given sample can be recomputed for arbitrary values of b.

$b \geq 1$ the expected performance (in terms of score and norm violations) of a policy generated w.r.t. the best weights we identified in our hyperparameter search for that particular b. As we can see, increasing b (i.e., approaching a b which would yield a lexicographic ordering) leads primarily to marginal improvements of adherence to the *hungry* norm (that is, the higher-priority norm). This comes at the cost of a significant degradation of compliance for the lower priority *vegetarian* norm. A secondary effect is that with less compliance to *vegetarian*, the score increases. The choice of a lower b, then, results in a significantly higher rate of compliance to $vegetarian$, coming at the cost of only slightly worse performance for $hungry$, illustrating how relaxing the lexicographic assumption (decreasing the magnitude of b) can result in significantly better performance holistically.

Plots for the other bases can be found in the the supplementary materials contained in the GitHub repository.

4.4 Evaluation of DQN/PPO given the Weights

We evaluated the optimal weights identified in the hyperparameter search using the same settings as in the search (rather than reporting the best observed performance during the search, which would be upward biased by the selection procedure). We also evaluated the weights with the respective other learning algorithm, using the optimal settings identified before. Each result was averaged over 20 independently learned policies, with each policy evaluated for 10,000 episodes.

Learning from High-Level Features The results of our experiments with high-level features are shown in Table 2.

We can see that in general, the baseline policies (generic DQN and PPO) perform decently (winning 84 % and 85 % of games, respectively, and yielding roughly 83 % of the maximum score). Meanwhile, norms are violated quite frequently (between 64 and 94 % of possible violations).

For the agents trained under the influence of the VEGAN norm base, violations are on average less than 0.01 per game: 0.24% and 0.13% of the baseline violations for DQN and PPO respectively. For the VEGETARIAN norm base, the decrease in violations is less sharp, and while the number of times the the permission $permBlue$ is leveraged is very high for the DQN policies, it was quite low for PPO, a more than 60% decrease from baseline (as opposed to the nearly 5% increase for DQN). The likely explanation is that PPO has a significantly more difficult time differentiating between the ghosts due to the choice of features. For HUNGRY, on the other hand, PPO far outperforms DQN; while DQN saw a decrease of about 81% from the baseline $hungry$ violations, PPO resulted in a decrease of nearly 93%.

HUNGRYVEGAN, which contains the conflict between $vegan$ and $hungry$, proved difficult for DQN; for DQN, the higher priority norm, $hungry$, is violated in more than 40% of games (as opposed to the mere 4% for PPO). For DQN, $vegan$ is only violated on average 0.0933 times per game (hence the violation rate for $hungry$), while for PPO it was 1.0319. As for "unnecessary violations" for DQN, in 0.4% of games more than one ghost was eaten, and just under 1% of games a ghost is eaten even though $hugry$ is violated. For PPO, in 5.6% of games more than one ghost is eaten, while a ghost is eaten while $hungry$ is violated in 1.6% of games. When we add $permBlue$ (for HUNGRY VEGETARIAN), violations of $hungry$ decrease by over half for DQN (and decrease to 7.7% for the alternative weights), and increase to 4.8% for PPO (increasing to 6.8% for PPO for the

Norm Base	Settings	hungry viols.	penalty viols.	vegetarian viols.	vegan viols.	perm. levgd.	win %	score
Baseline	DQN, 20 mil. steps	0.6469	3.4327	1.8540	3.7116	1.8540	84.16	1490.06
	PPO, 20 mil. steps	0.7270	3.3875	1.8153	3.6440	1.8153	85.15	**1498.44**
Vegan	DQN, 2.5 mil. steps *	-	-	-	0.0089	-	**93.06**	868.94
	PPO, 5 mil. steps	-	-	-	**0.0048**	-	89.23	838.69
Vegetarian	DQN, 10 mil. steps *	-	-	0.0528	-	**1.9049**	89.92	1213.73
	PPO, 20 mil. steps	-	-	**0.0238**	-	0.7326	87.68	963.06
Hungry	DQN, 20 mil. steps	0.1205	-	-	-	-	82.43	1438.23
	PPO, 10 mil. steps *	0.0563	-	-	-	-	83.17	1441.45
HungryVegan	DQN, 20 mil. steps	0.4060	-	-	0.0933	-	18.093	44.24
	PPO, 20 mil. steps *	0.0362	-	-	1.0319	-	81.67	908.84
HungryVegetarian	DQN, 20 mil. steps *	0.1725	-	0.2060	-	1.8602	89.39	1222.65
	PPO, 10 mil. steps	0.0483	-	0.3998	-	1.7877	86.41	1193.99
	DQN, 20 mil. steps, alt. weights *	0.0776	-	0.0520	-	1.8466	89.55	1182.01
	PPO, 10 mil. steps, alt. weights	0.0675	-	0.3426	-	1.8571	86.41	1200.19
HungryPenaltyVegan	DQN, 20 mil. steps	0.1193	**0.0**	-	0.8600	-	77.43	855.75
	PPO, 20 mil. steps *	**0.0331**	0.0032	-	1.0987	-	89.98	1029.54

Table 2: Experimental results for all norm bases for learning from high-level features; entries with the best performance are bolded.

Norm Base	hungry viols.	vegetarian viols.	vegan viols.	perm. levgd.	win%	score
Baseline	0.7138	1.8248	3.6506	1.8248	83.82	1497.14
Vegan	-	-	0.0009	-	88.14	837.36
Vegetarian	-	0.1005	-	1.5782	89.84	1176.13
Hungry	0.1386	-	-	-	65.23	1260.48
Hungry (best)	0.0740	-	-	-	85.75	1500.26

Table 3: Experimental results for training from images.

Method	Norm Base	Win %	Avg. Score	Viols.
PPO + NRBs (ours)	Vegan	89.23	838.69	**0.005**
	Vegetarian	87.68	963.06	**0.024**
DQN + NRBs (ours)	Vegan	**93.06**	**868.94**	0.009
	Vegetarian	**89.92**	**1213.73**	0.053
OFTEN-DEEPRL	Vegan	89	824.84	0.06
	Vegetarian	88	1139.45	0.03
Policy Fixes	Vegan	88	792.63	0.19
	Vegetarian	74	899.71	0.05

Table 4: Comparative experimental results for Pac-Man.

alternative weights). The average times when $permBlue$ is leveraged sits around 1.8 for all policies. Finally, when we add the norm $penalty$ into the mix in HUNGRYPENALTYVEGAN, we find that performance for $hungry$ actually improves from HUNGRYVEGAN, and $vegan$ violations are quite close for PPO, and closer to what we would expect (0.8600) for DQN. $penalty$ is only rarely violated; not at all for DQN, and only 0.08% all possible times for PPO.

Learning from Images The results for the selected experiments we performed with DQN on image-based states are in Table 3. It can be seen that the policies learned to adhere to the Vegan and Vegetarian norm bases. However, compared to learning from features, the policies incur more violations of the Vegetarian norm base, due to the difficulty of distinguishing ghost colors, which turn very pale orange and blue when scared. The policies also successfully learned to comply with the HUNGRY norm base, approximately matching the average violation ratio of DQN in Table 2, albeit with a lower win percentage overall. In the best trial shown in the last row of Table 3, we can see that policies trained on images can match the performance of feature-based policies.

We have excluded results for the remaining norm bases, since training failed to produce well-performing policies. This is caused by problems related to temporal credit assignment, as the penalty from violating the Hungry norm base is delayed. The connection between the punishment for not eating a ghost and reaching a score of 100 is hard to learn, when in conflict with the vegan norm, which punishes eating ghosts. In future work, we will aim to remedy such issues by adapting techniques like potential-based reward shaping, as used in reward-machine-based RL (Icarte et al. 2023). However, like the original RB technique, these would require adaptation (likely significant, in the case of reward machines) for application in the normative context.

4.5 Comparison to State of the Art

We next present a comparison to policy fixes (Adam and Eiter 2025) and OFTEN-DEEPRL (Lopez-Miguel et al. 2025). To our knowledge, these are the only techniques that have applied comparable norms in a deep RL context, albeit restricted to maintenance obligations. Both require specialized modelling of norms and abstract environments in ASP; therefore, we limit the comparison to *Vegan* and *Vegetarian*, since these norm bases have been modelled by the authors.

Table 4 shows a comparison of our results with feature-based state representation to the reference results presented by Lopez-Miguel et al. (2025). We can see that NRB-based policies mostly produce better results than both existing approaches in terms of score and violations.

In addition to this quantitative evaluation, there are further noteworthy differences. While we used a higher sampling budget, both other approaches rely on ASP-based planning, making individual training steps considerably slower. The planning requires an abstract model of the environment and norms. In contrast, NRBs are simpler to apply, only requiring a violation specification in LTLf and no prior knowledge of the environment dynamics. We also achieve fewer norm violations than previous approaches that used

non-neural function approximation in a larger version of the maze, where avoiding ghosts is easier. (Noothigattu et al. 2019) achieved 0.03 average violations of the vegan norm using inverse RL, while (Neufeld et al. 2022) achieved 0.02 and 0.04 average violations using a *normative supervisor*, which corrects action choices to achieve norm compliance.

5 Conclusion

We have adapted normative restraining bolts (NRBs) to the deep learning context, a necessity for the large and complex state/action spaces characteristic of many real-life applications. We integrated NRBs into two deep RL methods, DQN and PPO, to learn normative behaviour defined by explicitly represented norms. This allowed us to combine precise and understandable formal representations with subsymbolic learning from, e.g., images. We furthermore presented a sample-based method for arriving at the weights needed to enforce norms under customizable conflict-resolution mechanisms via hyperparameter optimization, combining these contributions to learn difficult normative dynamics like contrary-to-duty obligations and conflicting norms in the game *Pac-Man*, finding that our approach surpasses comparable techniques. This weight-finding method, though employed only in the context of NRBs in our paper, should generalize to other deep RL adaptations of reward shaping and multi-objective approaches that require finding weights for punishments and rewards.

In the immediate future, there are several ways in which we intend to develop our framework. For now we assume that DFA inputs (labels) can be derived from a given labelling function mapping states to sets of atoms. However, this may not always be realistic, and adding a component for recognizing labels from, say, images, would be a significant step toward practical implementation. This step is not trivial; learning an incorrect or noisy labelling function can result in unwarranted or missed punishments (due to false positives or negatives of norm violations), which can slow convergence and reduce policy quality. Another area we hope to explore further is selecting AIPs; the method used to estimate AIPs, while feasible, is still computationally expensive. We would like to explore alternative approaches or refinements of our current approach, perhaps returning to the MORL setting, albeit in the deep learning context. Finally, we plan to examine complex real-world scenarios involving a greater number of norms (e.g., autonomous driving) in future work. Bayesian optimization is known to be well-suited for high-dimensional search spaces, so we expect our approach to scale to larger norm bases, perhaps requiring a modest increase in the number of trials.

Acknowledgments

This research was funded in part by the Vienna Science and Technology Fund (WWTF) project 10.47379/ICT22023 and in part by the Austrian Science Fund (FWF) 10.55776/COE12. We thank the TU Wien dataLAB team for their support and for providing the computational resources essential to our experiments.

AI Declaration

Generative AI was used exclusively to assist in the implementation of code for plotting the experimental results.

References

Abel, D.; MacGlashan, J.; and Littman, M. L. 2016. Reinforcement learning as a framework for ethical decision making. In *AAAI Workshop: AI, Ethics, and Society*, volume 16.

Adam, S., and Eiter, T. 2025. Asp-driven emergency planning for norm violations in reinforcement learning. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 39, 14772–14780.

Akiba, T.; Sano, S.; Yanase, T.; Ohta, T.; and Koyama, M. 2019. Optuna: A next-generation hyperparameter optimization framework. In *The 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, 2623–2631.

Al Nahian, M. S.; Frazier, S.; Riedl, M.; and Harrison, B. 2024. Training value-aligned reinforcement learning agents using a normative prior. *IEEE Transactions on Artificial Intelligence*.

Anderson, M., and Anderson, S. L. 2007. Machine ethics: Creating an ethical intelligent agent. *AI Magazine* 28(4):15.

Anderson, A. R. 1958. A reduction of deontic logic to alethic modal logic. *Mind* 67(265):100–103.

Bacchus, F.; Boutilier, C.; and Grove, A. 1996. Rewarding behaviors. In *Proceedings of the National Conference on Artificial Intelligence*, 1160–1167.

Barrett, L., and Narayanan, S. 2008. Learning all optimal policies with multiple criteria. In *Proceedings of the 25th international conference on Machine learning*, 41–47.

Bergstra, J.; Bardenet, R.; Bengio, Y.; and Kégl, B. 2011. Algorithms for hyper-parameter optimization. *Advances in neural information processing systems* 24.

Brafman, R.; De Giacomo, G.; and Patrizi, F. 2018. Ltlf/ldlf non-markovian rewards. In *Proceedings of the AAAI conference on artificial intelligence*, volume 32.

Chen, Y. F.; Everett, M.; Liu, M.; and How, J. P. 2017. Socially aware motion planning with deep reinforcement learning. In *2017 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 1343–1350. IEEE.

Chisholm, R. M. 1963. Contrary-to-duty imperatives and deontic logic. *Analysis* 24(2):33–36.

Christiano, P. F.; Leike, J.; Brown, T. B.; Martic, M.; Legg, S.; and Amodei, D. 2017. Deep reinforcement learning from human preferences. In Guyon, I.; von Luxburg, U.; Bengio, S.; Wallach, H. M.; Fergus, R.; Vishwanathan, S. V. N.; and Garnett, R., eds., *Advances in Neural Information Processing Systems 30: Annual Conference on Neural Information Processing Systems 2017, December 4-9, 2017, Long Beach, CA, USA*, 4299–4307.

De Giacomo, G.; Iocchi, L.; Favorito, M.; and Patrizi, F. 2019. Foundations for restraining bolts: Reinforcement learning with ltlf/ldlf restraining specifications. In *Proceedings of the international conference on automated planning and scheduling*, volume 29, 128–136.

De Giacomo, G.; Vardi, M. Y.; et al. 2013. Linear temporal logic and linear dynamic logic on finite traces. In *IJCAI*, volume 13, 854–860.

De Giacomo, G.; Vardi, M. Y.; et al. 2015. Synthesis for ltl and ldl on finite traces. In *Proceedings of the Twenty-Fourth International Joint Conference on Artificial Intelligence, IJCAI 2015*, 1558–1564. AAAI Press.

Degrave, J.; Felici, F.; Buchli, J.; Neunert, M.; Tracey, B. D.; Carpanese, F.; Ewalds, T.; Hafner, R.; Abdolmaleki, A.; de Las Casas, D.; Donner, C.; Fritz, L.; Galperti, C.; Huber, A.; Keeling, J.; Tsimpoukelli, M.; Kay, J.; Merle, A.; Moret, J.; Noury, S.; Pesamosca, F.; Pfau, D.; Sauter, O.; Sommariva, C.; Coda, S.; Duval, B.; Fasoli, A.; Kohli, P.; Kavukcuoglu, K.; Hassabis, D.; and Riedmiller, M. A. 2022. Magnetic control of tokamak plasmas through deep reinforcement learning. *Nat.* 602(7897):414–419.

DeNero, J., and Klein, D. 2014. UC Berkeley CS188 intro to AI – course materials. We adapted the gym environment available at https://github.com/sohamghosh121/PacmanGym.

Gabbay, D.; Horty, J.; Parent, X.; Van der Meyden, R.; van der Torre, L.; et al. 2021. *Handbook of deontic logic and normative systems*. College Publications, 2021.

Governatori, G.; Hulstijn, J.; Riveret, R.; and Rotolo, A. 2007. Characterising deadlines in temporal modal defeasible logic. In *Proc. of AUSAI*, LNCS, 486–496.

Governatori, G. 2015. Thou shalt is not you will. In *Proceedings of the 15th International Conference on Artificial Intelligence and Law*, 63–68.

Icarte, R. T.; Klassen, T. Q.; Valenzano, R.; Castro, M. P.; Waldie, E.; and McIlraith, S. A. 2023. Learning reward machines: A study in partially observable reinforcement learning. *Artif. Intell.* 323:103989.

Kasenberg, D., and Scheutz, M. 2018a. Inverse norm conflict resolution. In *Proceedings of the 2018 AAAI/ACM Conference on AI, Ethics, and Society*, 178–183.

Kasenberg, D., and Scheutz, M. 2018b. Norm conflict resolution in stochastic domains. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 32.

Kober, J.; Bagnell, J. A.; and Peters, J. 2013. Reinforcement learning in robotics: A survey. *The International Journal of Robotics Research* 32(11):1238–1274.

Lopez-Miguel, I. D.; Adam, S. P.; Bartocci, E.; Eiter, T.; and Tappler, M. 2025. OFTEN-DEEPRL: on-the-fly teaching of ethical norms to deep reinforcement learning agents. In Lynce, I.; Murano, N.; Vallati, M.; Villata, S.; Chesani, F.; Milano, M.; Omicini, A.; and Dastani, M., eds., *ECAI 2025 - 28th European Conference on Artificial Intelligence, 25-30 October 2025, Bologna, Italy - Including 14th Conference on Prestigious Applications of Intelligent Systems (PAIS 2025)*, volume 413 of *Frontiers in Artificial Intelligence and Applications*, 3315–3322. IOS Press.

Mnih, V.; Kavukcuoglu, K.; Silver, D.; Rusu, A. A.; Veness, J.; Bellemare, M. G.; Graves, A.; Riedmiller, M.; Fidjeland, A. K.; Ostrovski, G.; et al. 2015. Human-level control through deep reinforcement learning. *nature* 518(7540):529–533.

Moor, J. 2006. The nature, importance, and difficulty of machine ethics. *IEEE Intelligent Systems* 21:18–21.

Neufeld, E.; Bartocci, E.; and Ciabattoni, A. 2022. On normative reinforcement learning via safe reinforcement learning. In *PRIMA 2022*.

Neufeld, E. A.; Bartocci, E.; Ciabattoni, A.; and Governatori, G. 2022. Enforcing ethical goals over reinforcement-learning policies. *Journal of Ethics and Information Technology*.

Neufeld, E. A.; Ciabattoni, A.; and Tulcan, R. F. 2024. Norm compliance in reinforcement learning agents via restraining bolts. In *Legal Knowledge and Information Systems*. IOS Press. 119–130.

Neufeld, E. A.; Ciabattoni, A.; and Tulcan, R. F. 2025. Combining morl with restraining bolts to learn normative behaviour. In Kwok, J., ed., *Proceedings of the Thirty-Fourth International Joint Conference on Artificial Intelligence, IJCAI-25*, 4615–4623. International Joint Conferences on Artificial Intelligence Organization. Main Track.

Neufeld, E. A. 2022. Reinforcement learning guided by provable normative compliance. In *Proceedings of ICAART 2022*, 444–453.

Neufeld, E. A. 2024. Learning normative behaviour through automated theorem proving. *KI-Künstliche Intelligenz* 1–19.

Noothigattu, R.; Bouneffouf, D.; Mattei, N.; Chandra, R.; Madan, P.; Varshney, K. R.; Campbell, M.; Singh, M.; and Rossi, F. 2019. Teaching ai agents ethical values using reinforcement learning and policy orchestration. In *Proc. of IJCAI: 28th International Joint Conference on Artificial Intelligence*. ijcai.org.

Peschl, M.; Zgonnikov, A.; Oliehoek, F. A.; and Siebert, L. C. 2022. Moral: Aligning ai with human norms through multi-objective reinforced active learning. In *Proceedings of the 21st International Conference on Autonomous Agents and Multiagent Systems*, 1038–1046.

Peschl, M. 2021. Training for implicit norms in deep reinforcement learning agents through adversarial multi-objective reward optimization. In *Proceedings of the 2021 AAAI/ACM Conference on AI, Ethics, and Society*, 275–276.

Pnueli, A. 1977. The temporal logic of programs. In *18th Annual Symposium on Foundations of Computer Science*, 46–57. IEEE.

Raffin, A.; Hill, A.; Gleave, A.; Kanervisto, A.; Ernestus, M.; and Dormann, N. 2021. Stable-baselines3: Reliable reinforcement learning implementations. *Journal of Machine Learning Research* 22(268):1–8.

Rodriguez-Soto, M.; Serramia, M.; Lopez-Sanchez, M.; and Rodriguez-Aguilar, J. A. 2022. Instilling moral value alignment by means of multi-objective reinforcement learning. *Ethics and Information Technology* 24(1):1–17.

Rodriguez-Soto, M.; Rădulescu, R.; Rodriguez-Aguilar, J. A.; Lopez-Sanchez, M.; and Nowé, A. 2023. Multi-objective reinforcement learning for guaranteeing alignment

with multiple values. In *2023 Adaptive and Learning Agents Workshop at AAMAS*.

Sallab, A. E.; Abdou, M.; Perot, E.; and Yogamani, S. 2017. Deep reinforcement learning framework for autonomous driving. *Electronic Imaging* 2017(19):70–76.

Schulman, J.; Levine, S.; Abbeel, P.; Jordan, M.; and Moritz, P. 2015. Trust region policy optimization. In *International conference on machine learning*, 1889–1897. PMLR.

Schulman, J.; Wolski, F.; Dhariwal, P.; Radford, A.; and Klimov, O. 2017. Proximal policy optimization algorithms. *arXiv preprint arXiv:1707.06347*.

Sherali, H., and Soyster, A. 1983. Preemptive and non-preemptive multi-objective programming: Relationship and counterexamples. *Journal of Optimization Theory and Applications* 39(2):173–186.

Stiennon, N.; Ouyang, L.; Wu, J.; Ziegler, D. M.; Lowe, R.; Voss, C.; Radford, A.; Amodei, D.; and Christiano, P. F. 2020. Learning to summarize from human feedback. *CoRR* abs/2009.01325.

Sutton, R. S., and Barto, A. G. 2018. *Reinforcement learning: An introduction*. MIT press.

Sutton, R. S.; McAllester, D.; Singh, S.; and Mansour, Y. 1999. Policy gradient methods for reinforcement learning with function approximation. *Advances in neural information processing systems* 12.

The IEEE Global Initiative on Ethics of Autonomous and Intelligent Systems. 2019. *IEEE standard review — Ethically aligned design: A vision for prioritizing human wellbeing with artificial intelligence and autonomous systems*. IEEE, first edition.

Vishwanath, A.; Dennis, L. A.; and Slavkovik, M. 2024. Reinforcement learning and machine ethics: a systematic review. *arXiv preprint arXiv:2407.02425*.

Wallach, W., and Allen, C. 2008. *Moral machines: Teaching robots right from wrong*. Oxford University Press.

Watkins, C. J., and Dayan, P. 1992. Q-learning. *Machine learning* 8:279–292.

Wu, Y.-H., and Lin, S.-D. 2018. A low-cost ethics shaping approach for designing reinforcement learning agents. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 32.

Xiao, S.; Li, J.; and Wang, Z. 2024. Model-free motion planning of complex tasks subject to ethical constraints. In *International Conference on Human-Computer Interaction*, 116–129.