Computational-Statistical Gaps in Reinforcement Learning

Daniel Kane[*]
University of California, San Diego
dakane@eng.ucsd.edu

Sihan Liu
University of California, San Diego
sil046@ucsd.edu

Shachar Lovett[†]
University of California, San Diego
slovett@cs.ucsd.edu

Gaurav Mahajan
University of California, San Diego
gmahajan@eng.ucsd.edu

February 13, 2022

Abstract

Reinforcement learning with function approximation has recently achieved tremendous results in applications with large state spaces. This empirical success has motivated a growing body of theoretical work proposing necessary and sufficient conditions under which efficient reinforcement learning is possible. From this line of work, a remarkably simple minimal sufficient condition has emerged for sample efficient reinforcement learning: MDPs with optimal value function V^* and Q^* linear in some known low-dimensional features. In this setting, recent works have designed sample efficient algorithms which require a number of samples polynomial in the feature dimension and independent of the size of state space. They however leave finding computationally efficient algorithms as future work and this is considered a major open problem in the community.

In this work, we make progress on this open problem by presenting the first computational lower bound for RL with linear function approximation: unless NP=RP, no randomized polynomial time algorithm exists for deterministic transition MDPs with a constant number of actions and linear optimal value functions. To prove this, we show a reduction from UNIQUE-SAT, where we convert a CNF formula into an MDP with deterministic transitions, constant number of actions and low dimensional linear optimal value functions. This result also exhibits the first computational-statistical gap in reinforcement learning with linear function approximation, as the underlying statistical problem is information-theoretically solvable with a polynomial number of queries, but no computationally efficient algorithm exists unless NP=RP. Finally, we also prove a quasi-polynomial time lower bound under the Randomized Exponential Time Hypothesis.

1 Introduction

Function approximation has a long history in reinforcement learning [**???**] and game playing [**??**]. More recently, this merger of reinforcement learning's algorithmic techniques with supervised learning's generalization schemes has achieved tremendous results in various applications with large state spaces, in areas such as game playing [**???**], robotics [**?**] and biology [**?**]. Since, one would expect the statistical and computational demand for these algorithms to grow at least linearly with the size of the state space [**?**], it is quite surprising that these algorithms generalize so well in large state spaces. That said, the computational requirements for existing algorithms have become exceedingly high. For example, AlphaZero was trained on

[*]Supported by NSF Award CCF-1553288 (CAREER) and a Sloan Research Fellowship.
[†]Supported by NSF Award CCF-1909634.

5000 tensor processing units (TPUs) for 13 days [**?**] and OpenAI Five trained its DOTA2 bots using 128000 CPUs [**?**] for 180 days (10 months in real time). This leads to a natural fundamental question: are such data and compute requirements fundamental or can we design efficient algorithms for these applications? More generally: *what minimal properties of environments leads to efficient RL algorithms?*

Over the last decade, this question has driven a growing body of theoretical work showing when *sample efficiency* is possible in RL for particular model classes, such as State Aggregation [**??**], Linear MDPs [**??**], Linear Mixture MDPs [**??**], Reactive POMDPs [**?**], Block MDPs [**?**], FLAMBE [**?**], Reactive PSRs [**?**], Linear Bellman Complete [**??**]. More generally, there are also a few lines of work which propose general frameworks, consisting of structural conditions which permit sample efficient RL; these include the Bellman rank [**?**], Witness rank [**?**], Bilinear Classes [**?**] and Bellman Eluder [**?**]. The goal in these latter works is to develop a unified theory of generalization in RL, analogous to the more classical notions in statistical complexity (e.g. VC-theory and Rademacher complexity) relevant for supervised learning.

A surprisingly minimal assumption which arose from these works is Linear $Q^*\&V^*$ [**?**] where both optimal value function V^* and optimal action-value function Q^* are linear in some known low-dimensional features. **?** showed that in this setting, there exists sample efficient RL algorithms which *regardless of the number of actions* require a number of samples polynomial in the feature dimension and independent of the size of the state space. However, when only either V^* or Q^* are linear, a series of works [**????**] showed that a phase transition occurs as one increases the number of actions: sample efficient algorithms exist for constant number of actions, and quickly transform into information theoretic exponential lower bounds as the number of actions exceeds the dimension of the features underlying Q^* or V^*.

Even though we have made considerable progress in understanding the minimal assumptions from the statistical perspective, the computational aspect of this problem is largely unknown. All the settings mentioned above (except under strong assumptions like linear transitions [**?**] and deterministic rewards [**?**]) do not have computationally efficient algorithms and previous works [**???**] leave designing computationally efficient algorithms as an important open problem. On the other hand, in spite of failed search for such computationally efficient algorithms over the last few years, there are no computational lower bounds for any of these settings (although previous attempts [**?**] have shown inefficiency of specific algorithms) and this is considered a major open problem in the community.

1.1 Our Contributions

In this work, we present the first computational lower bounds for RL with linear function approximation. Before stating our main results, we first need to state some key definitions that we use throughout the paper.

Markov Decision Process (MDP). We first define the framework for reinforcement learning, a Markov Decision Process (MDP). We define a deterministic MDP as a tuple $M = (\mathcal{S}, \mathcal{A}, R, P)$, where $\mathcal{S}$ is the state space, $\mathcal{A}$ is the action space, $R : \mathcal{S} \times \mathcal{A} \mapsto \Delta([0, 1])$ is the stochastic reward function [1], and $P : \mathcal{S} \times \mathcal{A} \mapsto \mathcal{S}$ is the deterministic transition function. An MDP M defines a discrete time sequential decision process where the agent starts from a starting state $s_0 \in \mathcal{S}$. Then, at each time t, the agent at some current state S_t, takes action A_t, receiving reward $R_t \sim R(S_t, A_t)$ and transitions to next state S_{t+1}. This goes on till the agent reaches the end state $\perp$. Each such trajectory/path from starting state s_0 to end state $\perp$ is of length at most horizon H. A deterministic, stationary policy $\pi : \mathcal{S} \mapsto \mathcal{A}$ specifies a decision-making strategy in which the agent chooses actions adaptively based on the current state, i.e. $A_t = \pi(S_t)$. Given a policy π and a state-action pair $(s, a) \in \mathcal{S} \times \mathcal{A}$, the Q-function and V-function under a policy π are defined as

$$V^\pi(s) = \mathbb{E}\left[\sum_{t=0}^{\tau-1} R(S_t, A_t) \mid S_0 = s, \pi\right], \quad Q^\pi(s,a) = \mathbb{E}\left[\sum_{t=0}^{\tau-1} R(S_t, A_t) \mid S_0 = s, A_0 = a, \pi\right], \tag{1}$$

[1] $\Delta([0, 1])$ denotes the set of all distributions over interval $[0, 1]$.

where $S_1, A_1, \ldots S_{\tau-1}, A_{\tau-1}$ are obtained by executing policy π in the MDP M and τ is the first time when policy π reaches the end state $\perp$, that is $S_\tau = \perp$ where it always holds that $\tau \leq H$. We use Q^* and V^* to denote the optimal value functions

$$V^*(s) = \sup_\pi V^\pi(s)\,, \quad Q^*(s,a) = \sup_\pi Q^\pi(s,a)\,, \quad s \in \mathcal{S}, a \in \mathcal{A}$$

We say that the optimal value functions V^* and Q^* can be written as a linear function of d-dimensional features $\psi\colon \mathcal{S} \cup (\mathcal{S} \times \mathcal{A}) \to \mathbb{R}^d$ if for all state s and action a, $V^*(s) = \langle \theta, \psi(s) \rangle$ and $Q^*(s,a) = \langle \theta, \psi(s,a) \rangle$ for some fixed $\theta \in \mathbb{R}^d$ independent of s and a.

Computational Problems. We next introduce 3-SAT, a satisfiability problem for 3-CNF formulas. In a 3-SAT problem, we are given as input, a 3-CNF formula φ with v variables and $O(v)$ clauses and our goal is to decide if φ is satisfiable. Our computational lower bound is based on a reduction from UNIQUE-3-SAT, a variant of 3-SAT. UNIQUE-3-SAT is the promise version of 3-SAT where the given formula is promised to have either 0 or 1 satisfying assignments.

The focus of this work is the computational RL problem, LINEAR-k-RL. In a LINEAR-k-RL problem with feature dimension d, we are given access to a deterministic MDP M with k actions and horizon $H = O(d)$ such that the optimal value functions Q^* and V^* can be written as a linear function of d-dimensional features ψ. Our goal is to output a good policy, which we define as any policy π that satisfies $V^\pi > V^* - 1/4$. Note that here V^π and V^* refers to the value of the policy π and optimal policy respectively at the starting state and is always in $[0, H]$ [2]. Moreover, the constant $1/4$ can be replaced by any arbitrary constant < 1. From now on, we always assume number of actions k is 2 or 3.

Complexity problem LINEAR-k-RL

Oracle: a deterministic MDP M with k actions, optimal value functions V^* and Q^* linear in d dimensional features ψ and horizon $H = O(d)$.

Goal: find policy π such that $V^\pi > V^* - 1/4$.

We now describe how the algorithm interacts with the MDP. We assume that the algorithm has access to the associated (i) reward function R, (ii) transition function P and (iii) features ψ. For all these functions, the algorithm provides a state s and action a (if needed) and receives a random sample from the distribution $R(s,a)$ (for the reward function), the state $P(s,a)$ (for the transition function) or feature $\psi(s)$ or $\psi(s,a)$ (for the features). We assume that each call accrues constant runtime and input/output for these functions are of size polynomial in feature dimension d.

We will often talk about randomized algorithm A solving a problem in time t with error probability p. By this we mean (i) A runs in time $O(t)$; (ii) for satisfiability problems, it returns YES on positive input instances with probability at least $1 - p$ and returns NO on negative input instances with probability 1; and (iii) for RL problem, it returns a good policy with probability at least $1 - p$.

1.1.1 No polynomial time algorithm for LINEAR-2-RL

With these considerations in mind, we present our main result that asserts that unless NP=RP, no randomized polynomial time algorithm can find a good policy in deterministic MDPs with a constant number of actions and linear optimal value functions.

Theorem 1.1 (LINEAR-2-RL $\in$ RP $\implies$ NP=RP). *Unless* NP=RP*, no randomized algorithm can solve* LINEAR-2-RL *with feature dimension* d *in time polynomial in* d *with error probability* $1/10$.

[2] in our constructions, we satisfy the more stringent condition that $V^* \in [0, 1]$.

This resolves the open problem from **?** and **?** by showing that unless RP=NP, no polynomial time randomized algorithm exists for deterministic transition MDPs with a constant number of actions and linear optimal value functions.

Our main technical contribution is a reduction from UNIQUE-3-SAT to LINEAR-3-RL such that a polynomial time algorithm for LINEAR-3-RL implies a polynomial time algorithm for UNIQUE-3-SAT. To achieve this, we use the input for UNIQUE-3-SAT: a 3-CNF formula φ with v variables, to design an input for LINEAR-3-RL: an MDP M_φ with 3 actions and optimal value functions V^* and Q^* linear in d-dimensional features. On a high level, the MDP is constructed such that each state represents an assignment to the UNIQUE-3-SAT variables and the goal is to "search" for the solution to the UNIQUE-3-SAT instance. In particular, at each state, the 3 actions available to the agent correspond to an unsatisfied clause which ensures at least one action available to the agent decreases the distance to the solution. To incentivize finding the solution, a large reward is awarded on reaching the solution and a very small expected reward on reaching the horizon (this reward is small enough that any polynomial time RL algorithm only receives 0 reward with high probability on reaching the horizon). This ensures that (i) finding a good policy also finds the satisfying assignment of φ and (ii) the optimal value functions V^* and Q^* are linear in some low dimensional features. We present this construction in Section 2.

To get lower bounds for LINEAR-2-RL, we use the same construction as above with a small modification. We replace the choice of 3 actions a_1, a_2 and a_3 at every state with a depth-2 binary tree, where the first action is a_1 and the second action leads to a new state which has actions a_2 and a_3. This allows us to simulate the hard 3-action MDP using a 2-action MDP while increasing our feature dimension d by at most a quadratic factor. We present this construction in Section 3.

These reductions allow us to simulate a polynomial time algorithm for UNIQUE-3-SAT on input φ by running the polynomial time algorithm for LINEAR-2-RL on MDP M_φ. More formally, our reduction gives a polynomial relationship between the complexity of UNIQUE-3-SAT and LINEAR-2-RL: a polynomial d^q time algorithm for LINEAR-2-RL implies a polynomial $v^{O(q^2)}$ time algorithm for UNIQUE-3-SAT.

Proposition 1.2. *Suppose* $q \geq 1$. *If* LINEAR-2-RL *with feature dimension* d *can be solved in time* d^q *with error probability* $1/10$*, then* UNIQUE-3-SAT *with* v *variables can be solved in time* $v^{O(q^2)}$ *with error probability* $1/8$*.*

This relates the complexity of UNIQUE-3-SAT to LINEAR-2-RL and LINEAR-3-RL. To relate these problems to complexity class NP, we use a seminal result from **?** which showed that uniqueness of solution can not be used to solve search problems quickly. In particular, they showed a randomized polynomial time reduction from 3-SAT to UNIQUE-3-SAT.

Theorem 1.3 (Valiant-Vazirani Theorem). *Unless* NP=RP*, no polynomial time randomized algorithm can solve* UNIQUE-3-SAT *with error probability* $1/8$*.*

Combining our reduction with Valiant-Vazirani Theorem proves our main result–Theorem 1.1.

1.1.2 Quasi-Polynomial Lower Bound for LINEAR-2-RL

We now present computational lower bound under a strengthening of NP $\neq$ RP conjecture, Randomized Exponential Time Hypothesis (rETH) [**?**], which asserts that probabilistic algorithms can not decide if a given 3-SAT problem with v variables and $O(v)$ clauses is satisfiable in sub-exponential time.

Definition 1.4 (Randomized Exponential Time Hypothesis (rETH)). *There is a constant* $c > 0$ *such that no* randomized *algorithm can decide* 3-SAT *with* v *variables in time* 2^{cv} *with error probability* $1/2$*.*

Randomized Exponential Time Hypothesis along with many variants motivated by Exponential Time Hypothesis [**?**] have been influential in discovering hardness results for a variety of problems see, e.g.

??. Under Randomized Exponential Time Hypothesis, our main result is a quasi-polynomial computational lower bound for learning good policies in deterministic MDPs with linear optimal value functions.

Theorem 1.5 (Quasi-polynomial lower bound for LINEAR-2-RL). *Under rETH, no randomized algorithm can solve* LINEAR-2-RL *with feature dimension d in time $d^{O(\log d/\log\log d)}$ with error probability* $1/10$.

This improves over our super-polynomial lower bound albeit depending on a much stronger hardness assumption. To prove this result, we use a different choice of parameters in our reduction and set the feature dimension d to be sub-exponential in the number of variables v to get the following:

Proposition 1.6. *If* LINEAR-2-RL *with feature dimension d can be solved in time $d^{O(\log d/\log\log d)}$ with error probability* $1/10$*, then* UNIQUE-3-SAT *with v variables can be solved in time $2^{O(v/\log v)}$ with error probability* $1/8$.

Here its important to note that we can not use Valiant-Vazirani Theorem to relate UNIQUE-3-SAT and 3-SAT, since it is consistent with Valiant-Vazirani Theorem that UNIQUE-3-SAT is solvable in $2^{\sqrt{v}}$ time but 3-SAT takes 2^v time. Therefore, we use a more refined lower bound for UNIQUE-3-SAT from **?** which showed that if UNIQUE-3-SAT with v variables can be solved in time $2^{\alpha v}$ for every $\alpha > 0$, then so can k-SAT for all $k \geq 3$.

Theorem 1.7 (?). *Assuming rETH is true, there exists a constant $c > 0$ such that no randomized algorithm can solve* UNIQUE-3-SAT *with v variables in time 2^{cv} with error probability* $1/2$.

In conjunction with our reduction, this gives a quasi-polynomial lower bound for LINEAR-2-RL under rETH. We leave as an open problem if the techniques introduced in this work can be used to prove an exponential lower bound for LINEAR-2-RL under rETH.

Our results give evidence that even though having linear optimal value functions is sufficient for sample efficient reinforcement learning **?**, it is not sufficient for computationally efficient reinforcement learning. More assumptions are required for computationally efficient algorithms, in addition to optimal value functions Q^* and V^* being linear in low-dimensional features, for example sub-optimality gap [**?**]. We hope that this work will open up new research avenues for finding minimal sufficient conditions for computationally efficient reinforcement learning. We now discuss a few further notable implications of this work.

- *Computational-Statistical Gap*: There are many problems which exhibit computational-statistical gaps i.e. regimes where the underlying statistical problem is information theoretically possible but no computationally efficient algorithm exists. Examples include community detection [**???**], planted clique [**??**] and sparse principal component analysis [**??**]. To the best of our knowledge, our computational lower bound is the first computational-statistical gap in reinforcement learning with function approximation. When both optimal value functions Q^* and V^* are linear, MDPs with any number of actions are statistically easy to solve [**?**] but our results show that no polynomial time algorithm can solve these MDPs even with a constant number of actions, unless NP=RP.

- *Natural Problem in* NP $\setminus$ P: There has been quite a lot of recent work in complexity theory literature on proving quasi-polynomial lower bounds based on Exponential Time Hypothesis (for e.g. dense constraint satisfaction problems [**?**], approximating best nash equilibrium [**?**] and approximating densest k-subgraph with perfect completeness [**?**]). This work adds RL with deterministic transition, linear bounded optimal value functions V^*, Q^* and constant number of actions as another natural problem in NP but not in P unless NP=RP.

Remainder of this paper. In Section 2 and Section 3, we present our lower bound constructions for 3 action and 2 action MDPs respectively.

2 Lower Bound for MDPs with 3 actions

In this section, we will prove the reduction, Proposition 2.1 and Proposition 2.2, restated versions of Proposition 1.2 and Proposition 1.6 for LINEAR-3-RL. The overall idea is to first build a randomized algorithm $\mathcal{A}_{SAT}$ which can decide UNIQUE-3-SAT using a randomized algorithm $\mathcal{A}_{RL}$ which solves LINEAR-3-RL. The two reductions only differ in their settings of parameters.

In the first setting, which we use to prove that no polynomial time algorithm exists for LINEAR-3-RL, we set the feature dimension d to be polynomial in the number of variables v. Under this setting, we can build a polynomial time randomized algorithm for UNIQUE-3-SAT using a polynomial time randomized algorithm for LINEAR-3-RL.

Proposition 2.1. *Suppose $q \geq 1$. If* LINEAR-3-RL *with feature dimension d can be solved in time d^q with error probability $1/10$, then* UNIQUE-3-SAT *with v variables can be solved in time $O(v^{8q+16q^2})$ with error probability $1/8$.*

In the second setting, which we use to prove a quasi-polynomial lower bound for LINEAR-3-RL, we set the feature dimension d to be sub-exponential in the number of variables v. This allows us to transform an exponential time lower bound for UNIQUE-3-SAT into a quasi-polynomial lower bound for LINEAR-3-RL.

Proposition 2.2. *If* LINEAR-3-RL *with feature dimension d can be solved in time $d^{\log d/(32 \log\log d)}$ with error probability $1/10$, then* UNIQUE-3-SAT *with v variables can be solved in time $2^{O(v/\log v)}$ with error probability $1/8$.*

Before we prove these results, we give a brief outline of our reduction from UNIQUE-3-SAT to LINEAR-3-RL. On a high level, we construct an MDP where the goal is to "search" for the solution w^* to a UNIQUE-3-SAT instance with v variables. In particular, at each time, the agent is given an unsatisfied clause and asked to flip assignment for a variable present in the clause. Notice that since the clause is unsatisfied, there must be at least one variable whose assignment differs from the solution and therefore, the agent can "reach" the solution in at most $d(w, w^*)$ steps. To incentivize the agent, if the agents at time l finds the solution i.e. $w = w^*$ or reaches the end of the MDP i.e. $l = H$, it receives reward according to the following degree-r polynomial

$$g(l, w) = \left(1 - \frac{l + \text{dist}(w, w^*)}{H + v}\right)^r.$$

We show how to build an MDPs from a UNIQUE-3-SAT instance in Section 2.1. Furthermore, we show that the optimal value functions V^* and Q^* for the constructed MDP are linear in $d = O(v^r)$-dimensional features. Since the expected reward at last layer of the MDP is $O(v^{-r^2})$ (which can be replaced with 0 for any poly(d) time RL algorithm), the only non-zero reward is achieved by solving the underlying UNIQUE-3-SAT instance, proving our reduction. We give a formal argument in Section 2.2, where we show how to build a randomized algorithm for UNIQUE-3-SAT using a randomized algorithm for LINEAR-3-RL. In Section 2.3, we discuss the two different settings of parameters which will prove Proposition 2.1 and Proposition 2.2.

2.1 From 3-CNF formulas to 3-action MDPs

We will start by defining a mapping from an input of UNIQUE-3-SAT problem: 3-CNF formula φ with v variables and $O(v)$ clauses to a MDP M_φ with 3 actions and $H = O(d)$ horizon with optimal value functions linear in d dimensions. Our informal goal is to design an MDP M_φ such that finding a good policy also implies finding the satisfying assignment for the formula φ. We now formally describe the MDP M_φ when the formula φ has a unique satisfying assignment $w^* \in \{-1, 1\}^v$ and later show how the MDP M_φ differs when the formula φ has no solution. See Figure 1 for an example.

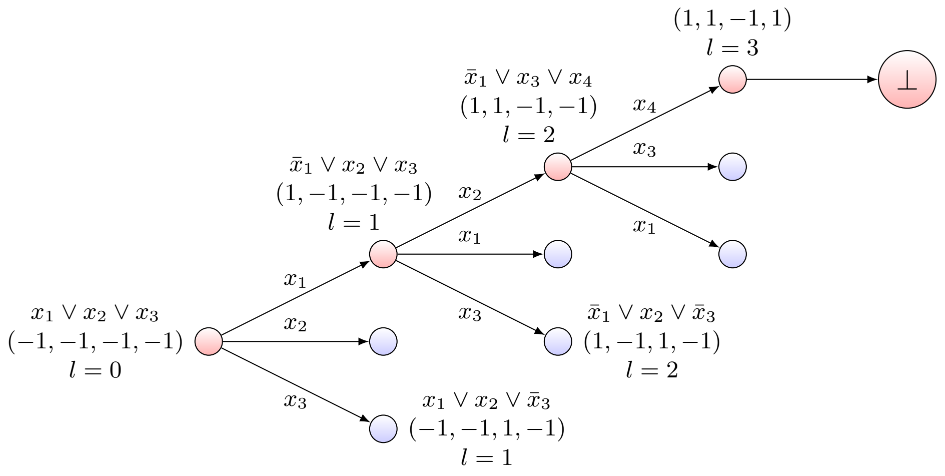

Figure 1: Example construction of 3-action MDP M_φ from a 3-CNF formula $(x_1 \vee x_2 \vee x_3) \wedge (\bar{x}_1 \vee x_2 \vee x_3) \wedge (\bar{x}_1 \vee x_3 \vee x_4) \wedge (x_1 \vee x_2 \vee \bar{x}_3) \wedge (\bar{x}_1 \vee x_2 \vee \bar{x}_3) \wedge (\bar{x}_3 \vee \bar{x}_3 \vee \bar{x}_3) \wedge (x_1 \vee x_1 \vee x_1)$. The only satisfying assignment for this formula is $(1, 1, -1, 1)$. The states are labelled by the corresponding assignment and unsatisfied clause which decides the available actions. The states in the optimal path are colored in red.

Transitions. In our setting, it will be useful to visualize an MDP as a tree, where nodes represent states and edges represent actions. A policy is then a sequence of actions or equivalently a path in the aforementioned tree. The MDP M_φ is a ternary tree i.e. each state/node in the tree has 3 children. The transitions/dynamics are deterministic i.e. the first action goes to first child, the second action goes to second child and so on.

Assignments. Each state is associated with an assignment to the v variables i.e. a binary vector in $\{-1, 1\}^v$ and a natural number l denoting the depth of the state. Our goal here is to choose assignments such that it is always possible to choose an action which decreases the hamming distance to the satisfying assignment. The root in the tree is associated with the all zeroes assignment $(-1, -1, \ldots, -1)$. For any state s with a non-satisfying assignment $w = (w_1, w_2, \ldots, w_v) \neq w^*$, the assignment associated to the three children are as follows. Since w is not a satisfying assignment, consider the first unsatisfied clause with variables $x_{i_1}, x_{i_2}, x_{i_3}$. The first child is associated with the assignment where the i_1-th bit of w is flipped, the second child is associated with vector where i_2-th bit is flipped and so on. More formally, the assignment associated to j-th child is $(w'_1, w'_2, \ldots, w'_v)$ where $w'_k = \neg w_k$ if $k = i_j$ and $w'_k = w_k$ otherwise. The two exceptions to this are (i) states with the satisfying assignment w^* and (ii) states at the last level H. For such states, all actions go to the end state $\perp$.

Rewards. To ensure that finding good policies implies finding the satisfying assignment in our MDP, we will only give rewards when a satisfying assignment is found or at the last layer. More formally, the rewards everywhere are zero except on (i) states with the satisfying assignment w^* and (ii) states on the last level H. In both the cases above, say the state is at level l with assignment w, then the associated reward distribution for any action is a Bernoulli distribution $Ber(g(l, w))$ where

$$g(l, w) = \left(1 - \frac{l + \text{dist}(w, w^*)}{H + v}\right)^r$$

and the Bernoulli distribution $Ber(p)$ is 1 with probability p and 0 with probability $1 - p$. Here r is a parameter which we will specify in Section 2.3. When the formula φ has no satisfying assignment, all rewards are 0. Note that in our simulation (Section 2.2), we don't know/use w^* and instead use an approximate reward function that is easy to compute.

Linear Optimal Value Functions. We next show that in the MDP M_φ, the optimal value functions V^* and Q^* can be written as a linear function of $d = O(v^r)$ dimensional features ψ, where $\psi(s)$ or $\psi(s,a)$ depends only on w, the corresponding assignment, and l, the depth of the state.

Proposition 2.3. *For any state s in level l with assignment w and action a,*

(i) the optimal value function is $V^*(s) = g(l, w)$.

(ii) for large enough v, there exists features $\psi(s), \psi(s,a) \in \mathbb{R}^d$ with feature dimension $d \le 2v^r$ depending only on state s and action a; and $\theta \in \mathbb{R}^d$ depending only on w^* such that V^* and Q^* can be written as a linear function of features ψ i.e. $V^*(s) = \langle \theta, \psi(s) \rangle$ and $Q^*(s,a) = \langle \theta, \psi(s,a) \rangle$.

Proof. To prove our first claim, we start by showing that there exists a policy π that achieves this value for each state. Let π be the policy which for any state s with assignment $w \neq w^*$ chooses the action which decreases the hamming distance $\text{dist}(w, w^*)$ by 1. Note that one such action always exists in our construction, since a satisfying assignment satisfies all clauses. Therefore, from a state s at level l with assignment w, we can reach a state with assignment w_1 such that either (i) w_1 is a satisfying assignment or (ii) w_1 is at the last level and on the optimal path from w to w^* i.e. $\text{dist}(w, w^*) = \text{dist}(w, w_1) + \text{dist}(w_1, w^*)$. In both cases,

$$V^\pi(s) = \left(1 - \frac{l + \text{dist}(w, w_1) + \text{dist}(w_1, w^*)}{H + v}\right)^r = g(l, w)$$

Next, for any other policy π' that ends on state s' at level l' with assignment w' (i.e. either $l' = H$ or $w' = w^*$), we have

$$V^{\pi'}(s) = \left(1 - \frac{l' + \text{dist}(w', w^*)}{H + v}\right)^r \le \left(1 - \frac{l + \text{dist}(w, w') + \text{dist}(w', w^*)}{H + v}\right)^r \le g(l, w)$$

where the first inequality follows from $l' - l \ge \text{dist}(w, w')$. This proves our first claim about V^* i.e. $V^*(s) = g(l, w)$.

To prove our second claim, that V^* and Q^* can be written as a linear function of features ψ, we will show that $V^*(s)$ can be written as a polynomial of degree at most r in w^*. To see why this is enough, we set θ to be all monomials in w^* of degree at most r. That is, each coordinate of θ corresponds to a multiset $S \subset [v]$ of size $|S| \le r$, and its value is $\theta_S = \prod_{i \in S} w_i^*$. We set $\psi(s)$ to be the corresponding coefficients in the polynomial V^*. Then, we can write $V^*(s) = \langle \theta, \psi(s) \rangle$. Since, there are at most $\sum_{i=0}^r v^i \le 2v^r$ many coefficients we can set the feature dimension as $d = 2v^r$.

Finally, we prove that $V^*(s)$ can be written as a polynomial of degree at most r in w and w^*. Firstly hamming distance $\text{dist}(w, w^*)$ is linear in both w and w^* i.e.

$$\text{dist}(w, w^*) = \frac{v - \langle w, w^* \rangle}{2}$$

Our claim follows from noting that $g(l, w)$ is a polynomial of degree r in $\text{dist}(w, w^*)$. Note that linear V^* implies linear Q^* in deterministic MDPs for $\psi(s,a) = \psi(P(s,a))$, since by definition, in MDPs with deterministic transition, $Q^*(s,a) = V^*(P(s,a))$. □

Even though $\psi(s)$ does not depend on w^*, unlike the constructions of **???**, $\psi(s)$ does depend on the MDP M_φ making this construction statistically easy but computationally hard to solve.

2.2 From RL algorithms to 3-SAT algorithms

We now build a randomized algorithm $\mathcal{A}_{SAT}$ for UNIQUE-3-SAT using a randomized algorithm $\mathcal{A}_{RL}$ for the RL problem. However, as mentioned before, since the runtime for $\mathcal{A}_{RL}$ accrues only constant runtime for each call to the MDP oracle, to efficiently build $\mathcal{A}_{SAT}$ using $\mathcal{A}_{RL}$, we need to be able to efficiently simulate the calls to MDP oracle, namely: calls to the reward function, the transition function and the features. To do so, we build an "approximate" simulator $\bar{M}_\varphi$ for the MDP oracle M_φ. The simulator $\bar{M}_\varphi$ is exactly MDP M_φ in terms of transition function and features associated with the MDP M_φ, but differs in the reward function at the last layer which is always 0 for the simulator $\bar{M}_\varphi$. This modification is crucial for an efficient reduction because unlike transitions and features for any state which can be computed in time poly(d) on the MDP M_φ, the rewards at the last layer when dist$(w, w^*) \neq 0$ require access to w^* which can not be done efficiently. With the purposed modification, we can execute each call to simulator $\bar{M}_\varphi$ in time poly(d).

Algorithm. On input 3-CNF formula φ, $\mathcal{A}_{SAT}$ runs the algorithm $\mathcal{A}_{RL}$ replacing each call to MDP oracle M_φ with the corresponding call to simulator $\bar{M}_\varphi$. Recall that the output for the RL algorithm in our setting is a sequence of actions. If the sequence of actions returned by $\mathcal{A}_{RL}$ ends on a state with assignment w, $\mathcal{A}_{SAT}$ outputs YES if w is the satisfying assignment and returns NO otherwise.

Correctness. We set the horizon $H = v^r$. We will assume throughout that $r \geq 2$ and that the runtime of $\mathcal{A}_{RL}$ is $\leq v^{r^2/4}$. Different settings of r satisfying these assumptions will prove Proposition 2.1 and Proposition 2.2 for 3-action MDPs, which we will discuss in Section 2.3. To complete our reduction, we will show the following:

(i) If algorithm $\mathcal{A}_{RL}$ outputs a policy π such that $V^\pi > V^* - 1/4$, then $\mathcal{A}_{SAT}$ on 3-CNF formula φ outputs YES if φ is satisfiable and NO otherwise.

(ii) If $\mathcal{A}_{RL}$ with access to MDP oracle M_φ outputs a policy π such that $V^\pi > V^* - 1/4$ with error probability $1/10$, then $\mathcal{A}_{RL}$ with access to simulator $\bar{M}_\varphi$ outputs a policy π such that $V^\pi > V^* - 1/4$ with error probability $1/8$.

These together will show that $\mathcal{A}_{SAT}$ solves UNIQUE-3-SAT with error probability $\leq 1/8$. We start by proving that if $\mathcal{A}_{RL}$ succeeds on MDP $\bar{M}_\varphi$, then $\mathcal{A}_{SAT}$ succeeds on 3-CNF formula φ. This follows from the fact that any good policy in the MDP M_φ must reach a state with satisfying assignment w^*.

Proposition 2.4. *Suppose $r > 1$ and horizon $H = v^r$. If $\mathcal{A}_{RL}$ outputs a policy π such that $V^\pi > V^* - 1/4$, then $\mathcal{A}_{SAT}$ on* 3-CNF *formula φ outputs YES if φ is satisfiable and NO otherwise.*

Proof. Since algorithm $\mathcal{A}_{SAT}$ always returns NO on an unsatisfiable formula, we restrict our attention to a satisfiable formula φ. In the MDP M_φ, (i) rewards are "very small" everywhere except on reaching the satisfying assignment i.e. the expected reward at the last layer in the MDP M_φ is upper bounded by (for large enough v and $r > 1$)

$$\left(1 - \frac{H}{H+v}\right)^r = \left(\frac{v}{H+v}\right)^r \leq v^{-r^2+r} < 1/4$$

and (ii) the optimal value V^* is large

$$V^* \geq \left(1 - \frac{v}{H+v}\right)^r = \left(1 + \frac{v}{v^r}\right)^{-r} \geq 1 - \frac{rv}{v^r} \geq \frac{1}{2}$$

where the second last inequality follows from Bernoulli's inequality and the last inequality holds for large enough v and $r > 1$. Therefore, if the value of policy is large i.e. $V^\pi > V^* - 1/4$, then the policy π (and therefore the corresponding sequence of actions) has to end on a state with the satisfying assignment w^*. By construction of $\mathcal{A}_{SAT}$, this implies $\mathcal{A}_{SAT}$ will succeed on the formula φ. $\square$

Since we can not simulate the rewards on MDP oracle M_φ efficiently, our reduction runs the algorithm $\mathcal{A}_{RL}$ on an approximate simulator $\bar{M}_\varphi$. However, it's not clear why $\mathcal{A}_{RL}$ would still succeed when each call to MDP oracle is replaced by a call to the simulator $\bar{M}_\varphi$. The following proposition shows that in fact $\mathcal{A}_{RL}$ would succeed on the outputs of simulator $\bar{M}_\varphi$ albeit with a smaller constant probability.

Proposition 2.5. *Suppose $r \geq 2$ and horizon $H = v^r$. Suppose $\mathcal{A}_{RL}$ with access to MDP oracle M_φ runs in time $v^{r^2/4}$ and outputs a policy π such that $V^\pi > V^* - 1/4$ with error probability $1/10$. Then $\mathcal{A}_{RL}$ with access to simulator $\bar{M}_\varphi$, still running in time $v^{r^2/4}$, outputs a policy π such that $V^\pi > V^* - 1/4$ with error probability $1/8$.*

Proof. Let $\Pr_{M_\varphi}$ and $\Pr_{\bar{M}_\varphi}$ denote the distribution on the observed rewards and output policies induced by the algorithm $\mathcal{A}_{RL}$ when running on access to MDP oracle M_φ and simulator $\bar{M}_\varphi$ respectively. Let R_i denote the reward received on the last layer at the end of i-th trajectory. Let T be the total number of trajectories sampled by algorithm $\mathcal{A}_{RL}$ when running on access to MDP oracle M_φ. By our assumption, $\mathcal{A}_{RL}$ runs in time $v^{r^2/4}$ and therefore $T \leq v^{r^2/4}$. Since the expected reward at the last layer in the MDP M_φ is upper bounded by (for large enough v and $r \geq 2$)

$$\left(1 - \frac{H}{H+v}\right)^r = \left(\frac{v}{H+v}\right)^r \leq v^{-r^2+r} \leq v^{-\frac{r^2}{2}}$$

and and the algorithm only visits at most $v^{r^2/4}$ states on last layer, we get by the union bound that with high probability all the rewards at the last level are zero. More precisely (and assuming v is large enough),

$$\Pr_{M_\varphi}\left[R_i = 0\ \forall i \in [T]\right] \geq 1 - v^{-r^2/4} \geq \frac{4}{5}$$

We say $\mathcal{A}_{RL}$ succeeds with access to M_φ (or $\bar{M}_\varphi$) if the output policy π after running for time at most $v^{r^2/4}$ satisfies $V^\pi > V^* - 1/4$. Using the above reasoning and the assumption that $\mathcal{A}_{RL}$ succeeds with access to MDP oracle M_φ with probability $9/10$ implies

$$\Pr_{M_\varphi}\left[\mathcal{A}_{RL} \text{ succeeds with access to } M_\varphi \mid R_i = 0\ \forall i \in [T]\right] \geq \frac{\frac{9}{10} - \frac{1}{5}}{\frac{4}{5}} = \frac{7}{8}$$

Note that the marginal distributions $\Pr_{M_\varphi}$ and $\Pr_{\bar{M}_\varphi}$ on output policy π given $R_i = 0\ \forall i \in [T]$ are exactly the same because MDP oracle $\bar{M}_\varphi$ and simulator M_φ only differ on last layer rewards. This implies

$$\begin{aligned}&\Pr_{\bar{M}_\varphi}\left[\mathcal{A}_{RL} \text{ succeeds with access to } \bar{M}_\varphi \mid R_i = 0\ \forall i \in [T]\right]\\ &= \Pr_{M_\varphi}\left[\mathcal{A}_{RL} \text{ succeeds with access to } M_\varphi \mid R_i = 0\ \forall i \in [T]\right]\end{aligned}$$

Since, $\Pr_{\bar{M}_\varphi}\left[R_i = 0\ \forall i \in [T]\right] = 1$, we conclude that

$$\Pr_{\bar{M}_\varphi}\left[\mathcal{A}_{RL} \text{ succeeds with access to } \bar{M}_\varphi\right] \geq \frac{7}{8}$$

$\square$

2.3 Setting of Parameters

It follows from Propositions 2.3 to 2.5 that if LINEAR-3-RL with feature dimension $d = 2v^r$ can be solved in time $v^{r^2/4}$ with error probability $1/10$, then UNIQUE-3-SAT with v variables can be solved in time $d \cdot v^{r^2/4}$ with error probability $1/8$ (here the extra d factor is because each call to the simulator $\bar{M}_\varphi$ takes d time). In this section, we discuss the two different settings of r we use to prove our lower bounds. As we increase r, we decrease the expected reward available to the algorithm at the last layer on the order of $v^{-O(r^2)}$, making the problem harder. However, increasing r also increases the feature dimension on the order of v^r. This non-polynomial gap in the feature dimension and expected reward at the last layer will give our main reduction.

In the first setting, we will set r to be a constant wrt number of variables v and prove that a polynomial algorithm for LINEAR-3-RL implies a polynomial algorithm for UNIQUE-3-SAT.

Proof of Proposition 2.1. For any $q \geq 1$, we set

$$r = 8q\,. \tag{2}$$

Note that $q \geq 1$ implies $r \geq 2$. Therefore, to prove our proposition, we just need to show

$$d^q \leq v^{r^2/4} \tag{3}$$

$$d \cdot v^{r^2/4} \leq v^{8q+16q^2+1} \tag{4}$$

under this setting of d and r. Here the first equation bounds the time complexity of LINEAR-3-RL in terms of feature dimension d and the second equation bounds the time complexity of UNIQUE-3-SAT in terms of the number of variables v. Equation (3) is true as

$$v^{\frac{r^2}{4}} = (v^r)^{\frac{r}{4}} \geq d^{\frac{r}{8}} = d^q$$

where the first inequality follows from $d \leq v^{2r}$ for large enough v and the last equality follows from Equation (2) above. Equation (4) holds since

$$d \cdot v^{r^2/4} = 2v^{r+r^2/4} = O(v^{8q+16q^2}),$$

where the first equality follows from $d = 2v^r$ and the last equality follows from Equation (2) for large enough v. $\square$

In Appendix A, we prove a more general version, Proposition A.1, which shows that a quasi-polynomial algorithm for LINEAR-3-RL implies a quasi-polynomial algorithm for UNIQUE-3-SAT.

In the second setting, we set r^2 to be almost linear in the number of variables v. This will prove Proposition 2.2 for 3-action MDPs.

Proof of Proposition 2.2. This follows exactly as proof of Proposition 2.1. We set

$$r = \left\lceil \frac{\sqrt{v}}{\log v} \right\rceil .$$

We proceed to show that (i) the time complexity of LINEAR-3-RL can be bounded by $v^{r^2/4}$ (ii) $d \cdot v^{r^2/4}$, which is the time complexity of UNIQUE-3-SAT if (i) holds, can be bounded by $2^{O(v/\log v)}$.

With this setting, the time complexity of LINEAR-3-RL simplifies to

$$d^{\frac{\log d}{32 \log\log d}} \leq v^{\frac{2r \log d}{32 \log\log d}} \leq v^{\frac{4r^2 \log v}{32 \log(r \log v)}} \leq v^{\frac{8r^2 \log v}{32 \log v}} = v^{\frac{r^2}{4}}$$

where the first and second inequality follows from $v^r \leq d \leq v^{2r}$ and third inequality follows from our setting of r.

Similarly, the time complexity of UNIQUE-3-SAT simplifies to

$$d \cdot v^{r^2/4} = 2v^{r+\frac{r^2}{4}} \leq v^{r^2} \leq v^{\frac{4v}{\log^2 v}} = 2^{\frac{4v}{\log v}}$$

where the first equality follows from $d = 2v^r$, first inequality follows for $r \geq 2$ and large enough v and the second inequality follows from our setting of r above. □

3 Lower Bound for MDPs with 2 actions

In this section, we prove computational lower bound for LINEAR-2-RL. Similar to Section 2, our proof is based on reduction from UNIQUE-3-SAT. We will modify the MDP M_φ with three actions into M_φ by introducing some intermediate states. See Figure 3 for an example of this modification for a single state.

Intermediate states. Recall that in M_φ each state is associated with an assignment. Let the i-th clause, which consists of three variables $x_{i_1}, x_{i_2}, x_{i_3}$, be the first unsatisfied clause. Then, the three actions available each correspond to flipping one of the variable in the clause. We will replace them by two actions: while one action still flips the last variable x_{i_3}, the other action leads to an intermediate state $s_{[i_1,i_2]}$. At the state $s_{[i_1,i_2]}$, two actions are available: one flips x_{i_1} and the other flips x_{i_2}.

Depth of state. In the 3 action MDP M_φ, the depth of a state is simply the length of the path that ends at the state. Here, we define the depth to be the number of non-intermediate states included in the path. That being said, the intermediate states will have the same depth as their parents.

Rewards. The rewards are the same as those in the 3 action MDP. Namely, rewards are only given at last layer or when the assignment is w^*. In particular, for a state with assignment w and depth l, the reward distribution is $Ber(g(l, w))$ where

$$g(l, w) = \left(1 - \frac{l + \text{dist}(w, w^*)}{H + v}\right)^r.$$

We now show that, even with this modification, the optimal value functions V^* and Q^* can still be written as a linear function of some low dimensional features.

Proposition 3.1. *For any state s in level l with assignment w and action a,*

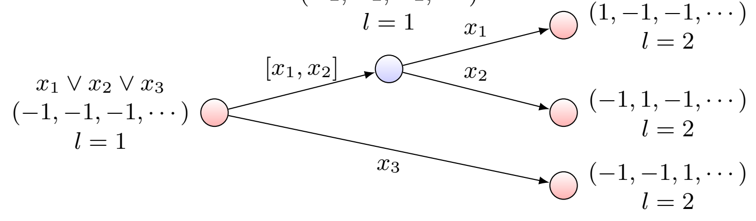

Figure 2: Part of a 2-action MDP corresponding to the CNF clause $(x_1 \vee x_2 \vee x_3)$. The non-intermediate states are colored red and the intermediate states are colored blue.

(i) If s is a non-intermediate state, then the optimal value function is $V^*(s) = g(l, w)$.

(ii) If s is an intermediate state that leads to actions which flip coordinates i_1 and i_2, then the optimal value function is

$$V^*(s_{[i_1,i_2]}) = \left(1 - \frac{l + \mathit{dist}(w, w^*) + 2 \cdot \mathbb{1}\{w_{i_1} = w^*_{i_1}\} \cdot \mathbb{1}\{w_{i_2} = w^*_{i_2}\}}{H + v}\right)^r .$$

(iii) for feature dimension $d = 2v^{2r}$, there exists features $\psi(s), \psi(s, a) \in \mathbb{R}^d$ depending only on state s and action a; and $\theta \in \mathbb{R}^d$ depending only on w^* such that V^* and Q^* can be written as a linear function of features ψ i.e. $V^*(s) = \langle\theta, \psi(s)\rangle$ and $Q^*(s, a) = \langle\theta, \psi(s, a)\rangle$.

Proof. The proof for the value function of non-intermediate state is identical to that in the 3-action MDP. We proceed to argue the second claim. For an intermediate state, the value function will be identical to its parent if the two actions available include a wrong bit that ought to be flipped in the optimal assignment w^*. Otherwise, no matter what action the agent takes, it will reach a non-intermediate state whose depth is $l + 1$ and hamming distance is $\text{dist}(w, w^*) + 1$. Compared to the value function of its parent, such intermediate state will have an extra 2 in the numerator. We encode the situation with the indicator term $\mathbb{1}\{w_{i_1} = w^*_{i_1}\} \cdot \mathbb{1}\{w_{i_2} = w^*_{i_2}\}$. This then gives the value function for these intermediate states.

Lastly, like in the proof of Proposition 2.3, it suffices to argue the value function is a degree $2r$ polynomial in w and w^*. This is by noticing that (i) $\text{dist}(w, w^*)$ is linear in w and w^*; and (ii) $\mathbb{1}\{w_{i_1} = w^*_{i_1}\} \cdot \mathbb{1}\{w_{i_2} = w^*_{i_2}\}$ is quadratic in w and w^* i.e.

$$\mathbb{1}\{w_{i_1} = w^*_{i_1}\} \cdot \mathbb{1}\{w_{i_2} = w^*_{i_2}\} = \frac{1}{4} \cdot \left(1 - w_{i_1} \cdot w^*_{i_1}\right)\left(1 - w_{i_2} \cdot w^*_{i_2}\right) .$$

Thus, the value function is overall a polynomial of degree $2r$ in w^*. As in Proposition 2.3, we can set θ to be all monomials in w^* of degree at most $2r$ and $\phi(s)$ to be the corresponding coefficients. Since there are at most $2v^{2r}$ such monomials, this concludes the proof. □

By Proposition 3.1, the feature dimension d of the 2 action MDP and the number of variables v in the UNIQUE-3-SAT instance are related by $d = 2v^{2r} \leq v^{3r}$. We are now ready to prove Proposition 1.2 and Proposition 1.6.

Proof of Proposition 1.2. Fix $q \geq 1$, we set $r = 12q$. Under this setting, we have

$$d^q \leq d^{r/12} \leq \left(v^{3r}\right)^{r/12} = v^{r^2/4} ,$$

where the first inequality follows from the setting of r and the second inequality follows from $d \leq v^{3r}$. The reduction then allows us to upper bound the complexity of UNIQUE-3-SAT by

$$d \cdot v^{r^2/4} \leq v^{r^2/4+3r} = v^{O(q^2)} ,$$

where the inequality again follows from $d \leq v^{3r}$. □

Proof of Proposition 1.6. The proof follows similarly as proof of Proposition 2.2. The only difference is that since d is now bounded by v^{3r} instead of v^{2r}, we need the runtime of LINEAR-2-RL in the assumption to also have a different constant in the exponent i.e. $d^{\log d/(72 \log\log d)}$. □

Acknowledgements

The authors would like to thank Sham Kakade, Akshay Krishnamurthy, Ayush Sekhari, Wen Sun, Csaba Szepesvari and Gellert Weisz for enlightening discussions and comments on initial draft.

A General Reduction from UNIQUE-3-SAT to LINEAR-3-RL

Proposition A.1. *Suppose $m \geq 0$ and $q \geq 1$. If* LINEAR-3-RL *with feature dimension d can be solved in time $d^{q\cdot(\log d)^{m/(m+2)}}$ with error probability $1/10$, then* UNIQUE-3-SAT *with v variables can be solved in time $v^{O(16^{m+2}q^{m+2})\cdot(\log v)^m}$ with error probability $1/8$.*

Proof. It follows from Proposition 2.3, Proposition 2.4 and Proposition 2.5 that for $2 \leq r < v$, if LINEAR-3-RL with feature dimension $d = 2v^r$ can be solved in time $v^{r^2/4}$ with error probability $1/10$, then UNIQUE-3-SAT with v variables can be solved in time $d \cdot v^{r^2/4}$ with error probability $1/8$ (as each call to simulator $\bar{M}_\varphi$ takes d time). For any $m \geq 0$ and $q \geq 1$, we set

$$r = \left\lceil \sqrt{(16q)^{m+2} \log^m v} \right\rceil . \tag{5}$$

Note that $m \geq 0$ and $q \geq 1$ implies $r \geq 2$. Therefore, to prove our claim, we just need to show the following equations hold for our setting of d and r:

$$v^{r^2/4} \geq d^{q\cdot\log^{\frac{m}{m+2}} d} \tag{6}$$

$$d \cdot v^{r^2/4} = v^{O((16q)^{m+2})\log^m v} \tag{7}$$

Here the first equation bounds the time complexity of LINEAR-3-RL in terms of feature dimension d and the second equation bounds the time complexity of UNIQUE-3-SAT in terms of the number of variables v.

Proof of Equation (6): To prove the first inequality, we lower bound r in terms of feature dimension d as

$$r \geq 8q\,(\log d)^{m/(m+2)} . \tag{8}$$

which can be proved by lower bounding r^{m+2} as follows

$$\begin{aligned} r^{m+2} = r^2 \cdot r^m &\geq (16q)^{m+2} \log^m v \cdot r^m \\ &= (16q)^{m+2} \log^m(v^r) \geq (16q)^{m+2} \log^m(\sqrt{d}) \geq (8q)^{m+2} \log^m d \end{aligned}$$

where the first inequality follows from our setting of r and the second inequality follows from $d \leq 2v^r \leq v^{2r}$ for $r > 0$ and large enough v. Substituting the lower bound in $v^{r^2/4}$, we can write the time complexity of LINEAR-3-RL in terms of feature dimension d as

$$v^{\frac{r^2}{4}} = (v^r)^{\frac{r}{4}} \geq d^{\frac{r}{8}} \geq d^{q\cdot\log^{\frac{m}{m+2}} d}$$

where the first inequality follows again from $d \leq v^{2r}$ and the second inequality follows from Equation (8) above.

Proof of Equation (7): The second equation follows by substituting our setting of r (Equation (2)) in $d \cdot v^{r^2/4}$,

$$d \cdot v^{r^2/4} = 2v^{r+r^2/4} = v^{O((16q)^{m+2})\log^m v}$$

where the first equality follows from $d = 2v^r$. $\square$