Sample Efficient Learning with Feature Selection for Factored MDPs

Zhaohan Daniel Guo ZGUO@CS.CMU.EDU
Carnegie Mellon University
Pittsburgh, PA, USA

Emma Brunskill EBRUN@CS.STANFORD.EDU
Stanford University
Stanford, CA, USA

Editor:

Abstract

In reinforcement learning, state is often represented by feature vectors. Prior sample complexity bounds scale with the complexity of all features. However, not all features may be necessary for learning a good policy. Therefore it is of significant interest to understand if the sample complexity can scale with the complexity of necessary features instead of all features. We answer this in the affirmative for at least one important case of interest: factored Markov Decision Processes. We show that is possible to eliminate unnecessary features by using directed exploration and leveraging the negative information from failing to reach desired states. Under mild assumptions, this is sufficient to show there exists an RL algorithm whose sample complexity scales with the cardinality of the parent sets of the necessary features, rather than the parent sets of all features. This yields an exponential improvement in sample complexity bounds when the maximum cardinality of the parent sets of the necessary features is smaller than for all features.

1. Introduction

In many machine learning and AI control problems, choosing which features to represent the state of the domain is critical and has significant impact on sample efficiency and performance. Though deep learning avoids the direct need for feature selection, it often requires an enormous amount of data. In supervised learning there has been significant interest in when a sparse subset of the input feature set is needed to predict the output variable, and the resulting theoretical analysis on the amount of data required. In this paper we are interested in formally analyzing a related setting of sparse feature dependence in reinforcement learning. RL introduces additional difficulties due to the fact that the data acquired is not independent and identically distributed, but instead is determined by the actions taken by the agent. Though there has been some promising results that sparsity can be exploited in contextual bandit problems Abbasi-Yadkori et al. (2012), to our knowledge there has been little formal work demonstrating a similar benefit for Markov Decision Processes (MDPs).

As a concrete first step, we focus on RL in factored Markov Decision Processes, where the state consists of a set of features, and the dynamics model of each feature depends on a specific subset of the features, called its parent set. Prior theoretical analysis has demonstrated that this factored structure allows the sample complexity (the number of steps on which an algorithm may take an

action that is not near optimal) to scale exponentially with the size of the largest parent set, rather than the number of features in total (Strehl et al., 2007; Diuk et al., 2009; Chakraborty and Stone, 2011). However, we are interested in the case where the value function (of any policy) depends only on a single subset of all features, but it is unknown which features are necessary. Our aim is to investigate whether it is possible to take advantage of the additional sparse structure to further reduce the sample complexity for learning factored MDPs.

We show in the affirmative, that it is possible to do Probably Approximately Correct (PAC) RL where the sample complexity scales as an exponential function of only the in-degree (size of largest parent set) of the necessary features. Our result is an exponential improvement in the sample complexity over prior PAC algorithms for factored MDPs (Chakraborty and Stone, 2011) that do no explicit feature selection, if the in-degree of the necessary features is smaller than the in-degree of the full feature set. We prove this result by presenting an algorithm that achieves this bound with *no knowledge of which nor how many features are necessary*, nor any knowledge of the structure of the underlying dynamics of the factored MDP.

Our key insight is to make two assumptions that then enable us to leverage *negative* information to (a) identify when our estimated dynamics model must be wrong, and to (b) pinpoint at least one aspect of the model that is wrong. This idea allows us to elimate features for which we have poor models when computing our optimal policy. Our algorithm uses directed exploration to repeatedly try to visit specific states. Prior work (Guo and Brunskill, 2015; Liu et al., 2016) has shown that directed exploration can yield PAC RL algorithms that are competitive with or in some cases improve upon prior PAC results. Our first assumption is that the domain has a known bound on the diameter (diameter has been used in other work, e.g. Jaksch et al. (2010); Bartlett and Tewari (2009)), and this enables the algorithm to detect when our dynamics model is incorrect ((a)) from the failure to reach a state in a bounded number of steps. Our second assumption, which we call the Superset assumption, states that if the parent set for a state variable is too small, including additional parents will yield a detectable difference in at least one of the dynamics parameter estimates for that variable, even if we do not consider the full set of that feature's true parents. The superset assumption allows the algorithm to detect which feature has an incorrect sized set of parents ((b)). The second assumption seems somewhat stronger than desired but we suspect empirically it is likely to always be satisfied (indeed it was satisfied in our preliminary toy simulations). Our algorithm can be viewed as a tool in support of this analysis, and is not designed to be practical. We hope that some of our insights, in particular the idea of leveraging negative information, may have potential benefit for the future design of empirically-oriented algorithms.

2. Setting

The state s in a factored finite MDP is a feature vector $(x_1, x_2, \ldots, x_n)$ where $x_i \in Dom_i$ and $|Dom_i| = d$ (Kearns and Koller, 1999). Each P_i is the transition probability for feature x_i, dependent on its parent set of features Par_i. The reward is defined as $R(s,a) = \sum_j^{|R|} R_j(s,a)$ and each reward function R_j is a function of a set of features Par_j. The in-degree of a factored MDP is the size of the largest parent set over all features/rewards $\max_i |Par_i|$. Let F' denote the set of all features. There exists a set of necessary features F, that includes all features for the rewards and the parents of the dynamics models for those features. This implies that a factored MDP defined

only over the set F has an identical value function to the value function in the original space F' for any policy, and the original optimal value function is constant over the unnecessary features. In factored finite MDP RL the transition/reward dynamics are unknown (i.e. parent sets are unknown), the in-degree is unknown, and the necessary features are unknown.

Algorithm 1

Input: m, H, M
for $K = 1$ to $|F'|$ **do**
  $F, \widehat{T}$=LearnAndSelect(K,m,H)
  Compute π^* for $F, \widehat{T}$
  Execute π^* for M steps
end for

Algorithm 2 LearnAndSelect

Input: K, m, H
Initialize F to all features
$G \leftarrow$ all possible feature-value vectors of size $2K$
while Exists a element g of G not visited m times **do**
  $\forall s\ R_{tmp}(s) = 0$, $R_{tmp}(g) = 1$
  Compute optimistic policy π_o using R_{tmp}
  stuck=True
  for $t = 1$ to H **do**
    Execute π_o
    If a feature-value vector that has not yet been visited m times is visited, set stuck=False and break out of loop
  end for
  if stuck **then**
    Run Superset Test to eliminate incorrect parent sets
    Eliminate f from F if all possible parent sets for it are eliminated
    Remove feature-value vectors from G whose features are no longer in F
  end if
end while
Return remaining F

3. Theoretical Analysis

Our main result is to prove (by construction) that there exists an RL algorithm for factored MDPs that is PAC with a sample complexity that is bounded as a function of the max in degree (cardinality of the parent set) of the necessary features, rather than the max in degree of all features.

As in past factored MDP RL work (Chakraborty and Stone, 2011) we measure the performance of π using average reward, where $U^\pi(s) = \lim_{T\to\infty} U^\pi(s,T) = \lim_{T\to\infty} \frac{1}{T}\mathbb{E}(\sum_{t=1}^{T} r_t | s_1 = s, \pi)$ (Kearns and Singh, 2002), assume $U^\pi(s)$ is independent of s and can be denoted as just U^π, and

assume the ϵ-return mixing time T_ϵ is given. T_ϵ is defined as that for any policy π and $T' \geq T_\epsilon$, $|U^\pi - U^\pi(T')| \leq \epsilon$ i.e. T_ϵ is long enough to see ϵ-optimal average reward.

Theorem 1 *For any ϵ and δ, let $T_{\epsilon,D} = \max(D, T_\epsilon)$, J be the in-degree of the necessary features, and n be the total number of features. Then there exists an algorithm such that with probability at least $1-\delta$, for all $K \geq J$ i.e. at least as large as the in-degree of the necessary features, the average per-step reward is ϵ-optimal i.e. $|U - U^*| \leq \epsilon$, where the total number of time steps to reach $K = J$ (sample complexity) is*

$$O\left(\frac{J^2 D^2 (dn)^{11J+10}}{|A|^{-10} R_{\max}^{-10} T_{\epsilon,D}^{-16} \epsilon^{10}} \log^2(dn T_{\epsilon,D} R_{\max} |A|/\delta)\right) \quad (1)$$

.

Before proceeding with a brief description of the algorithm that enables the above theorem, we first state our two key assumptions.

Assumption 1 *(Diameter Assumption) Let s, s' be any two states. Let $D_\pi(s, s')$ be the random variable for the number of steps it takes policy π to start at s and reach s' the first time. Let $D(s, s') = \min_\pi \mathbb{E}(D_\pi(s, s'))$. The MDP diameter $D \geq \max_{s,s'} D(s, s')$ is known.*

Assumption 2 *(Superset Assumption) Assume a state feature x_i has an incorrect parent set W of size K under a dynamics model that is ϵ accurate for correct parent sets. Let U be the true parent set for x_i. Consider supersets of parent features of size $2K$ which can be smaller than U. Then for at least one of those supersets, the dynamics model for x_i differs, $|P(x_i|W', a) - P(x_i|W, a)|_1 \geq O(\epsilon)$ for some action a and all possible values of those parent set variables.*

We now describe an algorithm that will satisfy our theorem. We emphasize that this algorithm is used to provide an existence proof, and is not intended to be used experimentally. However we think it is possible that some of the ideas in it may be useful for the design of algorithms used for practical use. Due to space limitations, details are in the appendix.

Our algorithm 1 proceeds by fixing a possible in-degree K and then executing a subprocedure (LearnAndSelect, Algorithm 2). The idea is that the algorithm will try to learn the dynamics model and reward model of the MDP, assuming the maximum in degree of all possible necessary state and reward features is K. To do so, the LearnAndSelect algorithm uses directed exploration to try to reach different target combinations of state features values. If $K >= J$ (the true in-degree of necessary features) this direct exploration will succeed, the algorithm will learn good models of the domain, and can use these to compute a near optimal policy. To complete this part, we leverage prior results for factored MDP RL which does not perform feature selection and that relies on knowing the correct in-degree, but that can learn the correct parent sets and dyanmics models, the Adaptive k-Meteorologists Algorithm (Diuk et al., 2009). The optimistic policy (line 6) is then computed by picking a parent set out of the remaining possible parent sets for every feature such that results in the largest optimal value.

The key issue is to identify when K is insufficiently small for some features. To do so the LearnAndSelect (Algorithm 2) subprocedure leverages our two key assumptions to make progress. It repeatedly tries to reach a particular state tuple of $2K$ features. If it cannot succeed, this indicates that it has a poor dynamics model for at least one of the state features: otherwise by Assumption

1, we would be able to compute a policy that could reach a state with the targeted feature set. The second issue is to then try to eliminate the feature that has a poor model under the current in degree K. To do so we perform the superset test, which essentially detects which state feature(s) violate the superset assumption for the current K. This allows us to shrink the considered set of features. This subroutine can thus make progress and eventually learns a good model for all features that it can with an in degree of K before terminating.

We now provide a very brief overview of the main technical results. Details are in the appendix.

First, consider when $K \geq J$, i.e. we guess an in-degree that is at least as large as the in-degree of the necessary features. Recall that throughout LearnAndSelect, we are accumulating samples (s, a, s', r). We use the Adaptive k-Meteorologists Algorithm (Diuk et al., 2009) to learn a model from the samples. Lemma 2 tells us that as long as we can get enough samples from visiting feature-value targets, we can learn an accurate dynamics model.

The main result is Lemma 3, which ensures that we make progress either towards learning a more accurate dynamics model, or towards eliminating possible parent sets for features. We rely on the diameter assumption and the superset assumption to make progress. The diameter assumption ensures we either reach our target with high probability, or our dynamics model has some place it is incorrect. If there is some place incorrect, then we continue with the superset test by gathering many more samples from running the policy we know is failing in order to get samples from states where our dynamics model is predicting incorrectly. Then we use the superset assumption in order to identify which parent sets in our dynamics model are incorrect, and eliminate them. Since there are a finite number of possible parent sets, we will eventually eliminate all of the incorrect ones and end up with a dynamics model that is accurate for the remining state features.

Finally, when LearnAndSelect finishes, we can use our accurate dynamics model to compute a near-optimal policy that we then execute for a long enough time to offset the sub-optimal performance during LearnAndSelect. Note that in the earlier stages when $K < J$, we have no guarantees on performance. It is only for all $K \geq J$ when we can show that we will get an accurate dynamics model. Unfortunately we cannot detect when $K \geq J$, so we have to continue to run LearnAndSelect.

Lemma 2 *Let F be a subset of features with in-degree K. Suppose we have $m = O\left(|A|K\frac{d}{\epsilon_1^2}\log(d|F|/\delta_1)\right)$ samples from every feature-value target of size $2K$ of F. Then we can construct a dynamics model for F with an L_1 accuracy of ϵ_1 with probability $1 - \delta_1$.*

Lemma 3 *(Exploration Episode Lemma) The following holds w.p. $1 - \delta_1$. At the end of each iteration of the while loop (line 4 – line 17) in the LearnAndSelect algorithm (Algorithm 2), one of two things will happen: either the target g or another feature-value vector that has not been visited m times will be visited, or some potential parent set will be eliminated as a possible parent for some feature.*

Lemma 4 *(LearnAndSelect Lemma) The following holds w.p. $1 - \delta_1$. After LearnAndSelect (Algorithm 2) is finished, all targets g will either have been visited m times, or one of its features will have been eliminated. If $K \geq J$ where J is the in-degree of the necessary features, all necessary features will remain in F. This will take $O\left(\frac{D^2K(dn)^{3K}m^2R_{\max}}{\epsilon_2}\log(dn/\delta_1)\right)$ steps.*

We now present a brief sketch of our main theorem. It assumes a value for m,

$$m = \tilde{O}\left(\frac{KdR_{\max}^4 n^4 |A|^5 d^{4K} \max(D, T_\epsilon)^8}{\epsilon_2^4} \log(1/\delta_2)\right) \tag{2}$$

which is further justified in the appendix.

Proof (Sketch)

From Lemma 7, LearnAndSelect will take $O\left(\frac{D^2 K (dn)^{3K} m^2 R_{\max}}{\epsilon_2} \log(dn/\delta_1)\right)$ steps. Once $K \geq J$, the necessary features will remain in the returned F, and by Lemma 2, the returned dynamics of the necessary features will be ϵ_1 accurate. Due to the value of m (eqn 2), the optimal policy computed from the learned dynamics will be ϵ_2-optimal, since we can ignore the dynamics of any unnecessary features in F by definition.

Thus after LearnAndSelect, Algorithm 1 will execute an ϵ_2-optimal policy for M steps. In order to get an average error of only ϵ, we let $\epsilon_2 = \epsilon/2$, and let M be large enough so the error in the steps in LearnAndSelect, which are counted as mistakes, are spread over the M steps.

Finally plugging in m (equation 2) results in the final sample complexity where we count all the steps for $K < J$ as mistakes and also use a union bound to bound the error for each K with $\delta_1 = \delta/n$. ■

4. Related Work

Prior work on factored MDP RL with formal theoretical bounds include Met-RMax (Diuk et al., 2009) and LSE-RMax (Chakraborty and Stone, 2011); however, such work does not perform feature selection. More recent work has significantly reduced the sample complexity of learning factored MDPs (Hallak et al., 2015), but requires strong structural assumptions and only apply to the batch setting.

Prior work for feature selection for factored MDPs performs it as a post-processing step after solving it and uses the learned features for transfer learning (Kroon and Whiteson, 2009). In contrast, our algorithm learns the necessary features while doing online reinforcement learning, and is more sample efficient as there is no need to fully learn the dynamics for all features. We also provide a formal theoretical analysis which is the first, to our knowledge, for this setting. Other prior work focus more on practical algorithms without formal guarantees such as using multinomial regression with LASSO (Nguyen et al., 2013).

There also exists work for feature selection for value function estimation but under the assumption of a linear value function (Painter-Wakefield and Parr, 2012; Geramifard et al., 2011). Feature selection can also be viewed as a form of model selection, where each model is a particular selection of features. Prior work has theoretical bounds for model selection such as the OAMS algorithm (Ortner et al., 2014); however those bounds depend on the square root of the number of models. For factored MDPs the number of models grows doubly exponential with the number of features.

5. Conclusion

We presented an algorithm for performing feature selection while solving factored MDPs whose sample complexity scales exponentially with the in-degree of the necessary features, potentially an exponential improvement over scaling with the in-degree of all features. Interesting theoretical questions include whether we can further relax our assumptions, or whether we can detect when we know the correct in-degree and stop exploration. Though our algorithm is an existence proof, we believe that the ideas in this algorithm, such as leveraging negative information through the failure to reach expected reachable goals, might be helpful for practical algorithms to help diagnose an error in its model model class, an error in an assumption, or perhaps that its model does not have the capacity to accurately represent the environment.

References

Yasin Abbasi-Yadkori, David Pal, and Csaba Szepesvari. Online-to-confidence-set conversions and application to sparse stochastic bandits. In *AISTATS*, volume 22, pages 1–9, 2012.

Peter L Bartlett and Ambuj Tewari. Regal: A regularization based algorithm for reinforcement learning in weakly communicating mdps. In *Proceedings of the Twenty-Fifth Conference on Uncertainty in Artificial Intelligence*, pages 35–42. AUAI Press, 2009.

Doran Chakraborty and Peter Stone. Structure learning in ergodic factored MDPs without knowledge of the transition function's in-degree. In *Proceedings of the 28th International Conference on Machine Learning (ICML-11)*, pages 737–744, 2011.

Thomas G Dietterich. Hierarchical reinforcement learning with the maxq value function decomposition. *J. Artif. Intell. Res.(JAIR)*, 13:227–303, 2000.

Carlos Diuk, Lihong Li, and Bethany R Leffler. The adaptive k-meteorologists problem and its application to structure learning and feature selection in reinforcement learning. In *Proceedings of the 26th Annual International Conference on Machine Learning*, pages 249–256. ACM, 2009.

Alborz Geramifard, Finale Doshi, Joshua Redding, Nicholas Roy, and Jonathan How. Online discovery of feature dependencies. In *Proceedings of the 28th International Conference on Machine Learning (ICML-11)*, pages 881–888, 2011.

Zhaohan Guo and Emma Brunskill. Concurrent pac rl. In *Twenty-Ninth AAAI Conference on Artificial Intelligence*, 2015.

Assaf Hallak, COM François Schnitzler, Timothy Mann, and Shie Mannor. Off-policy model-based learning under unknown factored dynamics. In *Proceedings of the 32nd International Conference on Machine Learning (ICML-15)*, pages 711–719, 2015.

Thomas Jaksch, Ronald Ortner, and Peter Auer. Near-optimal regret bounds for reinforcement learning. *Journal of Machine Learning Research*, 11(Apr):1563–1600, 2010.

Michael Kearns and Daphne Koller. Efficient reinforcement learning in factored mdps. In *IJCAI*, volume 16, pages 740–747, 1999.

Michael Kearns and Satinder Singh. Near-optimal reinforcement learning in polynomial time. *Machine Learning*, 49(2-3):209–232, 2002.

Mark Kroon and Shimon Whiteson. Automatic feature selection for model-based reinforcement learning in factored MDPs. In *Machine Learning and Applications, 2009. ICMLA'09. International Conference on*, pages 324–330. IEEE, 2009.

Yao Liu, Zhaohan Guo, and Emma Brunskill. Pac continuous state online multitask reinforcement learning with identification. In *Proceedings of the 2016 International Conference on Autonomous Agents & Multiagent Systems*, pages 438–446. International Foundation for Autonomous Agents and Multiagent Systems, 2016.

Trung Thanh Nguyen, Zhuoru Li, Tomi Silander, and Tze-Yun Leong. Online feature selection for model-based reinforcement learning. In *ICML (1)*, pages 498–506, 2013.

Ronald Ortner, Odalric-Ambrym Maillard, and Daniil Ryabko. Selecting near-optimal approximate state representations in reinforcement learning. In *Algorithmic Learning Theory*, pages 140–154. Springer, 2014.

Christopher Painter-Wakefield and Ronald Parr. Greedy algorithms for sparse reinforcement learning. In John Langford and Joelle Pineau, editors, *Proceedings of the 29th International Conference on Machine Learning (ICML-12)*, ICML '12, pages 1391–1398, New York, NY, USA, July 2012. Omnipress. ISBN 978-1-4503-1285-1.

Alexander L Strehl, Carlos Diuk, and Michael L Littman. Efficient structure learning in factored-state mdps. In *AAAI*, volume 7, pages 645–650, 2007.

Appendix A. Algorithm

The algorithm pseudocode is provided for reference.

Algorithm 1

Input: m, H, M
for $K = 1$ to $|F'|$ **do**
  $F, \widehat{T}$=LearnAndSelect(K,m,H)
  Compute π^* for $F, \widehat{T}$
  Execute π^* for M steps
end for

Algorithm 2 LearnAndSelect

Input: K, m, H
Initialize F to all features
$G \leftarrow$ all possible feature-value vectors of size $2K$
while Exists a element g of G not visited m times **do**
  $\forall s\ R_{tmp}(s) = 0$, $R_{tmp}(g) = 1$
  Compute optimistic policy π_o using R_{tmp}
  stuck=True
  for $t = 1$ to H **do**
    Execute π_o
    If a feature-value vector that has not yet been visited m times is visited, set stuck=False and break out of loop
  end for
  if stuck **then**
    Run Superset Test to eliminate incorrect parent sets
    Eliminate f from F if all possible parent sets for it are eliminated
    Remove feature-value vectors from G whose features are no longer in F
  end if
end while
Return remaining F

Appendix B. Theoretical Analysis

This section presents the supporting lemmas and main theorem for the performance of Algorithm 1. In the first section (Section B.1), we present the assumptions we make.

B.1 Assumptions

Assumption 1 *(Diameter Assumption) Let s, s' be any two states. Let $D_\pi(s, s')$ be the random variable for the number of steps it takes policy π to start at s and reach s' the first time. Let $D(s, s') = \min_\pi \mathbb{E}(D_\pi(s, s'))$. A diameter D means that $D \geq \max_{s,s'} D(s, s')$. It is an upper*

bound on the expected number of steps it takes to go between any two states. We assume D is known.

Assumption 2 *(Superset Assumption) Let π be a policy computed from assuming a particular parent set for every feature, where some of the parent sets are incorrect. Suppose π visits a state–action pair for which the predicted dynamics of an incorrect parent set are 2ϵ off from the true dynamics in $O(D^2)$ steps, with probability $1-\delta$. This would be the case if we get stuck (line 13) in Algorithm 2.*

Let that feature with an incorrect parent set be x_i. Let W of size K be the wrong parent set (of features) for feature f_i. Let U be the true parent set (of features). Consider supersets of W of size $2K$. There are up to $\binom{n}{K} \leq O(n^K)$ different superset sets of size $2K$. Each superset has d^{2K} instantiations of values. By the pigeonhole principle, after $d^{2K}m$ steps, at least one instantiation of every possible superset has been visited at least m times, since we gather data for all supersets at every step.

The assumption we make is that after $O(d^{2K}m)$ steps, there exists some superset W' out of all possible supersets of size $2K$ such that one of its instantiations has been visited at least m times and for that instantiation, $|P(x_i|W', a) - P(x_i|W, a)|_1 \geq O(\epsilon)$ for some action a.

B.2 Discussion of Assumption 2

The challenge with creating a Superset-like assumption comes from the intricate dependence on the policy. When the parent set for a feature is incorrect, the MLE estimate according to that incorrect parent set is completely determined by the state distribution induced by the policy. Because of this, it is extremely difficult to make a Superset-like assumption that only depends on the factored MDP. There can exist adversarial policies for which it is near impossible to detect that a parent set is incorrect without a massive amount of data. Thus our assumption attempts to use a favorable policy where we are repeatedly visiting informative states where the incorrect parent set has an incorrect transition model.

B.2.1 Comparison to G–SCOPE

For the G–SCOPE algorithm (Hallak et al., 2015), 3 assumptions are made: Strong Parent Superiority, Non-Parent Conditional Weakness, and Conditional Diminishing returns. Our Superset Assumption is similar to Strong Parent Superiority, and we do not make the other 2 assumptions. Because G–SCOPE is an offline algorithm, a suitable behavior policy as well as suitable domain dynamics are needed to satisfy these 3 assumptions. For our algorithm, the Superset Assumption is similarly dependent on the policy and dynamics, but the dependence on the policy is weaker because we have control over the policy; we are using a policy in which we know we are visiting state–action pairs for which our dynamics has incorrect predictions, gathering samples from those pairs, and so we expect in practice that this exploration policy is sufficient to satisfy the Superset Assumption.

B.2.2 SATISFYING THE SUPERSET ASSUMPTION

Here, we give some intuition for how it can be satisfied for simple settings.

The Stock Trading domain used in many prior PAC-FMDP algorithms (Strehl et al., 2007) satisfies the Superset Assumption. In the Stock Trading domain, the state is specified by a set of sectors, each sector having a number of stocks. Stocks are binary features indicating whether they are rising or falling. Sectors are binary variables indicating whether they have been bought or not. The probability of a stock rising is $P(rising) = 0.1 + 0.8 \times$ (fraction of stocks rising in same sector). The dynamics of the stocks are not affected by actions. Thus in this domain, no matter what the policy is, it will be possible to reach any stock state. Suppose there are 3 sectors and 3 stocks per sector. Suppose we mistakenly assume the in-degree is 1, whereas the true in-degree is 3. Then we will only be able to try parent sets with one stock, and end up aliasing multiple states. There is a significant difference in $P(rising)$ when the number of rising stocks is off by one; that difference is exactly $0.8/3$. Consider a parent set where the stock is currently not rising. The prediction of this parent would be the average of the predictions from 4 aliased states (the state of the other 2 stocks in the sector, which could have 0, 1, or 2 stocks rising). Comparing this to a superset of size 2, where both stocks are current not rising; in this case there are 2 aliased states, which is just whether the 3rd stock is rising or not. The probability of rising predicted by the superset of size 2 will be a fixed quantity lower than what is predicted by the parent set of size 1. If the requested accuracy of near-optimality, ϵ is small, then we will be able to detect this difference, no matter the policy, and thus rely on the superset assumption to pinpoint that this parent set is incorrect. Generalizing to more stocks and sectors with higher in-degree (the in-degree is the number of stocks per sector), the superset test will consider parent sets of k stocks as well as supersets of $2k$ stocks. There is a large difference in $P(rising)$ between k stocks and $2k$ stocks, with the additional k stocks all being in the rising state, and we will be able to detect that difference.

The Taxi domain (Dietterich, 2000; Hallak et al., 2015) also satisfies the Superset Assumption under many policies. Consider a uniform random exploration policy. Then all 25 locations in the 5x5 grid world would be visited with reasonable probability (there is no location that is particularly hard to get to). Movement actions have in-degree 2, so comparing a parent set of size 1 to supersets of size 2 will definitely work, since the true parent set is also size 2. The pick up and drop off actions have in-degree 3; they depend on both the row and column of the location of the taxi, as well as another feature representing the location of the passenger. With an incorrect parent set of size 1, we would be aliasing states for which the pickup/dropoff actions are both successful and not. However if we consider supersets of size 2, then there are feature-values where pickup/dropoff always fail. Thus we can detect the difference between sometimes being successful and always failing, and learn that the parent set of size 1 is incorrect.

B.3 Small Lemmas

In this section, we present some small lemmas that will be used later on.

Lemma 1 *(Necessary Feature Lemma) For any policy π*

$$Q^{\pi}(s, a, T) = Q^{\pi}(z, a, T) \tag{3}$$

where z is the state s restricted to only the necessary features from F i.e. Q-functions only depend on the necessary features. Furthermore, the transition dynamics of the unnecessary features have no effect on Q-functions. This means that as long as the dynamics for the necessary features are learned accurately, dynamics for unnecessary features may be arbitrary.

Proof

Let π be given. Let $s = (z, y)$ be the state decomposed into necessary features z and unnecessary features y. Initialize $Q(\cdot, \cdot, 0)$ to 0. We will perform induction. The base case for $Q(\cdot, \cdot, 0)$ is trivially true since it is a constant.

By induction

$$
\begin{aligned}
& Q(s, a, T+1) && (4)\\
& = R(s, a) + \sum_{s'} P(s'|s, a) \max_{a'} Q(s', a', T) && (5)\\
& = \sum_i R_{i,a}(s) && (6)\\
& + \sum_{s'} \prod_i P_{i,a}(s'|Par_{i,a}(s)) \max_{a'} Q(s', a', T) && (7)\\
& = \sum_i R_{i,a}(z) && (8)\\
& + \sum_{s'} \prod_i P_{i,a}(s'|Par_{i,a}(s)) \max_{a'} Q(z', a', T) && (9)\\
& = \sum_i R_{i,a}(z) && (10)\\
& + (\sum_{z'} \prod_{i \in z'} P_{i,a}(s'|Par_{i,a}(s)) \max_{a'} Q(z', a', T) && (11)\\
& \cdot \sum_{y'} \prod_{i \in y'} P_{i,a}(s'|Par_{i,a}(s))) && (12)\\
& = \sum_i R_{i,a}(z) && (13)\\
& + \sum_{z'} \prod_{i \in z'} P_{i,a}(s'|Par_{i,a}(s)) \max_{a'} Q(z', a', T) && (14)\\
& = Q(z', a', T) && (15)
\end{aligned}
$$

Note that the dynamics of the unnecessary features make no difference.

■

Lemma 2 *(Simulation Lemma) Kearns and Koller (1999) Let M be a factored MDP over n state variables with l entries in conditional probability table of the transition model. Let M' be an approximation to M where all the CPTs differ by at most $\alpha = O((\epsilon/T^2 l R_{\max})^2)$. Then for any policy π, $|U^\pi_M(T) - U^\pi_{M'}(T)| \leq \epsilon$.*

Lemma 3 *(Explore or Exploit Lemma) Fix a policy π. Let M and M_K be MDPs such that M and M_K agree on some states, but differ in dynamics and rewards for other states. Then $|U^{\pi}_{M_K}(T) - U^{\pi}_{M}(T)| \leq TR_{\max}P(escape)$ where $P(escape)$ is the probability of visiting a state in which the two models differ.*

Proof

Let τ_1 denote trajectories that stay within states where the two models agree and τ_2 denote trajectories where there are escapes to other states. Then

$$|U^{\pi}_{M_K}(T) - U^{\pi}_{M}(T)| \tag{16}$$
$$= \frac{1}{T}|\sum_{\tau,|\tau|=T} P_M(\tau)R(\tau) - \sum_{\tau,|\tau|=T} P_{M_K}(\tau)R(\tau)| \tag{17}$$
$$\leq \frac{1}{T}|\sum_{\tau_1} P_M(\tau_1)R(\tau_1) - \sum_{\tau_1} P_{M_K}(\tau_1)R(\tau_1)| \tag{18}$$
$$+ \frac{1}{T}|\sum_{\tau_2} P_M(\tau_2)R(\tau_2) - \sum_{\tau_2} P_{M_K}(\tau_2)R(\tau_2)| \tag{19}$$
$$\leq \frac{1}{T}|\sum_{\tau_2} P_M(\tau_2)R(\tau_2) - \sum_{\tau_2} P_{M_K}(\tau_2)R(\tau_2)| \tag{20}$$
$$\leq \frac{1}{T}\sum_{\tau_2} |P_M(\tau_2)R(\tau_2) - P_{M_K}(\tau_2)R(\tau_2)| \tag{21}$$
$$\leq \frac{1}{T}TR_{\max}\sum_{\tau_2} |P_M(\tau_2) - P_{M_K}(\tau_2)| \tag{22}$$
$$= R_{\max}P(escape) \tag{23}$$

Because non-escapes result in exactly the same trajectories with the same dynamics, so the probability of escaping to the other states is the same in both M and M_K. ■

Corollary 4 *Suppose π_1 is the optimal policy for M_k and π_2 is the optimal policy for M. Suppose $U^{*}_{M_K}(T) \geq U^{*}_{M}(T)$ i.e. M_K is optimistic. Then $U^{\pi_1}_{M_K} \geq U^{\pi_2}_{M} - TR_{\max}P(escape)$.*

Proof

$$U^{\pi_1}_{M} \geq U^{\pi_1}_{M_K} - R_{\max}P(escape) \tag{24}$$
$$\geq U^{\pi_2}_{M} - R_{\max}P(escape) \tag{25}$$

■

B.4 LearnAndSelect

We now prove several results about the LearnAndSelect (Algorithm 2) subprocedure.

First, note that throughout the whole subprocedure, we are accumulating samples (s, a, s', r) which we use to learn a dynamics model. There exists multiple algorithms which can take samples and learn a dynamics model for a factored MDP, and in general they use the samples to estimate probabilities $P(x_i|P, a)$, $P(x_i|Q, a)$, and $P(x_i|P \cup Q, a)$ where P, Q are potential parent sets for feature x_i, and use these estimates to narrow down which parent sets could be correct. In particular, one can use the Adaptive k-Meteorologists Algorithm (Diuk et al., 2009).

The optimistic policy (line 6) is then computed by picking a parent set out of the remaining possible parent sets for every feature such that results in largest optimal value.

B.4.1 DETERMINING m

In this section we determine a sufficient value for m, the number of samples that LearnAndSelect tries to obtain from all feature-value targets of size $2K$.

Lemma 5 *Let F be a subset of features with in-degree K. Suppose we have $m = O\left(|A|K\frac{d}{\epsilon_1^2}\log(d|F|/\delta_1)\right)$ samples from every feature-value target of size $2K$ of F. Then we can construct a dynamics model for F with an L_1 accuracy of ϵ_1 with probability $1 - \delta_1$.*

Proof

The Adaptive k-Meteorologists Algorithm (Diuk et al., 2009) is a sample efficient online learning algorithm for learning the dynamics of a factored MDP. Given any (nonstationary) exploration policy and a known in-degree, it will learn a near-optimal dynamics model making only a finite number of sub-optimal mistakes (prediction accuracy), with high probability.

By Hoeffdings, we need $O(\frac{d}{\epsilon_1^2}\log(d/\delta_1))$ samples to learn each set of values of a parent set to an L_1 accuracy of ϵ_1 (we apply Hoeffdings d times, learning the probability of each outcome with Hoeffdings) with probability $1 - \delta_1$. Since there are $|A|$ actions, the number of samples we need is

$$O\left(|A|\frac{d}{\epsilon_1^2}\log(d/\delta_1)\right) \tag{26}$$

By Adaptive k-Meteorologist, we need $O\left(\frac{1}{\epsilon_1^2}\log\frac{k}{\delta_1}\right)$ samples from $P(x_i|P \cup Q, a)$, i.e. the pair of parent sets P, Q, in order to eliminate whichever P or Q is more incorrect (k is the number of possible feature-value vectors of parents). Therefore if we have that many visits for all possible pairs $P \cup Q$, i.e. all feature-value pairs of size $2K$, then all incorrect parent sets will be eliminated, and we are left with parent sets whose predictions are all ϵ_1 accurate (and we can just pick one arbitrarily).

There are $\binom{|F|}{K}$ possible parent sets in total. Each parent set has d^K possible value instantiations, thus $k = O\left((d|F|)^K\right)$. Then a sufficient value for m is $m = O\left(|A|K\frac{d}{\epsilon_1^2}\log(d|F|/\delta_1)\right)$ where we do a union bound for the error probability with $\delta_1 = \frac{1}{\delta\binom{|D|}{K}}$ and bound $\binom{|F|}{K} \leq O(|F|^K)$.

■

B.4.2 DIRECTED EXPLORATION

In this section, we present several results on the directed exploration of LearnAndSelect.

Lemma 6 *(Exploration Episode Lemma) The following holds w.p.* $1 - \delta_1$. *At the end of each iteration of the while loop (line 4 – line 17) in the LearnAndSelect algorithm (Algorithm 2), one of two things will happen: either the target* g *or another feature-value vector that has not been visited* m *times will be visited, or some potential parent set will be eliminated as a possible parent for some feature.*

Proof

The idea behind the directed exploration is the Explore or Exploit lemma (Lemma 3) and the diameter assumption. The diameter assumption allows the algorithm to potentially reach g with high probability. The Explore or Exploit lemma allows the algorithm to either reach g or end up gathering new data, or fail. If it fails, then we take advantage of the Superset assumption to eliminate an incorrect parent set.

First, we compute how large H needs to be in order for a good policy to reach g with high probability. By the diameter assumption, there exists a policy expected to reach g within D steps, thereby obtaining a reward of 1 from the artificially defined reward function R_{tmp}. By the Markov Inequality, the probability of reaching the goal within $2D$ steps is at least $\frac{1}{2}$. Thus the optimal average value within $2D$ steps is at least $\frac{1}{4D}$. If we used an ϵ_2-optimal policy, it would have an expected value of at least $\frac{1}{4D} - \epsilon_2$. Let τ be trajectories of length $2D$. Then $\frac{1}{4D} - \epsilon_2 = \frac{1}{2D}\sum_\tau Pr(escape)Pr(\tau|escape)TotalReward(\tau)$. The probability that the ϵ_2-optimal policy reached the goal (escapes) can be lower bounded by the worst case scenario: every escape trajectory has every step giving a reward. That means the probability of reaching the goal (escape) is at least $\frac{1}{4D} - \epsilon_2$. Then probability of failing to reach the goal is at most $1 - \frac{1+2D\epsilon_2}{4D}$. Then by repeating this 2D-step trial N times, the error probability is upper bounded by $\left(1 - \frac{1+2D\epsilon_2}{4D}\right)^N$. Then that means if we want to have a failure probability of δ_1, we would need to repeat this $2D$-step sub-episode $\frac{\log(\delta_1)}{\log(1-\frac{1+2D\epsilon_2}{4D})}$ times. We can simplify the denominator $\log(1 - \frac{1+2D\epsilon_2}{4D})$ by the upper bound $\log(1 - \frac{1}{4D})$. Note that the $\log$ function is concave, so we can upper bound it with its first order approximation around $\log(1)$ i.e. by $O(-\frac{1}{D})$. Simplifying the whole fraction becomes $O(D\log(1/\delta_1))$. Thus we end up with

$$O(D^2 \log(1/\delta_1)) \tag{27}$$

as the number of steps we need before reaching the goal with high probability. Thus this is a lower bound for H and we also know in this case the while loop will terminate early after these many steps.

Now we consider the case when our optimistic policy is bad i.e. the probability of escaping is at least ϵ_2. Then either we get lucky and visit some other feature-value target that has not yet been visited m times, or we get stuck. If we get stuck, then we need the data in these H steps for the Superset test (line 13).

For the Superset test, we apply the Superset assumption (Assumption 2). We know that we got stuck so the data that we have is where the escape probability is high, thus meeting the superset assumption requirements of visiting distinguishing states (states where our model is incorrect).

Now we count how much data we need from getting stuck to perform the Superset Test. We need to gather new data for the prediction of supersets of size $2K$. By Assumption 2 we will need $O(d^{2K}m)$ steps. However since we only have an escape probability of ϵ_2, we need to add repeats to escape with high probability, just like we did earlier.

This means we need an additional factor of $O(\frac{R_{\max}}{\epsilon_2}\log(1/\delta_1))$. So we need

$$H > O\left(\frac{d^{2K}mR_{\max}}{\epsilon_2}\log(1/\delta_1)\right)$$

. Combining this with earlier means a sufficient H is

$$H = O\left(\frac{D^2d^{2K}mR_{\max}}{\epsilon_2}\log(1/\delta_1)\right) \tag{28}$$

By Lemma 5, we have an ϵ_1-accurate dynamics model, so together with the Simulation Lemma (Lemma 2) that means $\epsilon_1 = O\left(\frac{\epsilon_2^2}{\max(D,T_\epsilon)^4R_{\max}^2n^2|A|^2d^{2K}}\right)$ where n is the total number of features. So the updated value for m in terms of ϵ_2 is

$$m = O\left(\frac{KdR_{\max}^4n^4|A|^5d^{4K}\max(D,T_\epsilon)^8}{\epsilon_2^4}\log(dn\max(D,T_\epsilon)R_{\max}|A|/\delta_2)\right) \tag{29}$$

■

Lemma 7 *(LearnAndSelect Lemma) The following holds w.p. $1-\delta_1$. After LearnAndSelect (Algorithm 2) is finished, all targets g will either have been visited m times, or one of its features will have been eliminated. If $K \geq J$ where J is the in-degree of the necessary features, all necessary features will remain in F. This will take $O\left(\frac{D^2K(dn)^{3K}m^2R_{\max}}{\epsilon_2}\log(dn/\delta_1)\right)$ steps.*

Proof

Now we will count how many steps LearnAndSelect (Algorithm 2) will take. From Lemma 6, every while loop iteration (line 4 – line 17) contributes to one of two cases: visiting a feature-value vector that has not yet been visited m times, or the Superset Test.

Since there are at most $O((dn)^K)$ superset tests (each test eliminates at least one parent set), and we know how much data each superset test requires (equation 28), we combine those to get a total of

$$O((dn)^KH) \tag{30}$$

$$= O\left(\frac{D^2(dn)^{2K+1}mR_{\max}}{\epsilon_2}\log(1/\delta_1)\right) \tag{31}$$

steps towards superset tests.

Our targets are subsets of features and values of size $2K$, thus there are $O((dn)^{3K})$ targets. Each target needs to be visited m times, thus $O((dn)^{3K}mH)$ total steps are needed. Then the number of steps this all takes is

$$O((dn)^K H + (dn)^{3K} mH) \tag{32}$$

$$= O\left(\frac{D^2 K (dn)^{3K} m^2 R_{\max}}{\epsilon_2} \log(dn/\delta_1)\right) \tag{33}$$

where this includes a union bound over all $O((dn)^{3K})$ targets and $O((dn)^K)$

Thus eventually the while loop will have visited all possible targets from the remaining set of features F. If $K \geq J$, where J is the in-degree of the necessary features, then the necessary features must remain in F because they will never be eliminated through the superset test, since $K \geq J$.

■

B.5 Main Theorem

In this section, we give the main theorem and its proof.

Theorem 8 *Given ϵ, δ. Let $T_{\epsilon,D} = \max(D, T_\epsilon)$. Let J be the in-degree of the necessary features. Let n be the total number of features. Then the following is true of Algorithm 1 with probability $1 - \delta$*

1. *The total number of steps taken up to $K = J$ is*

$$O\left(\frac{J^2 D^2 (dn)^{11J+10} |A|^{10} R_{\max}^{10} T_{\epsilon,D}^{16}}{\epsilon^{10}} \log^2(dn T_{\epsilon,D} R_{\max} |A|/\delta)\right)$$

2. *For all $K \geq J$ i.e. at least as large as the in-degree of the necessary features, the average per-step reward is ϵ-optimal i.e. $|U - U^*| \leq \epsilon$*

Proof

From Lemma 7, LearnAndSelect will take $O\left(\frac{D^2 K (dn)^{3K} m^2 R_{\max}}{\epsilon_2} \log(dn/\delta_1)\right)$ steps. Once $K \geq J$, the necessary features will remain in the returned F, and by Lemma 5, the returned dynamics of the necessary features will be ϵ_1 accurate. Due to the value of m (eqn 29), the optimal policy computed from the learned dynamics will be ϵ_2-optimal, since we can ignore the dynamics of any unnecessary features in F by Lemma 1.

Thus after LearnAndSelect, Algorithm 1 will execute an ϵ_2-optimal policy for M steps. Counting all of the steps of LearnAndSelect as mistakes and setting as B, the average error per-step for any single $K \geq J$ is

$$\frac{R_{\max} B + \epsilon_2 M}{B + M} \tag{34}$$

So to bound this by ϵ, we need $\epsilon_2 < \epsilon$, so we'll use $\epsilon_2 = \epsilon/2$. Also we need to make M large enough. Then

$$\frac{2R_{\max}B + \epsilon M}{2B + 2M} \leq \epsilon \tag{35}$$

$$\iff 2B(\frac{R_{\max}}{\epsilon} - 1) \leq M \tag{36}$$

$$\impliedby M \geq \frac{2BR_{\max}}{\epsilon} \tag{37}$$

$$\impliedby M = O\left(\frac{BR_{\max}}{\epsilon}\right) \tag{38}$$

where $B = O\left(\frac{D^2K(dn)^{3K}m^2R_{\max}}{\epsilon_2}\log(dn/\delta_1)\right)$. So for each K, Algorithm 1 runs for $M + B$ steps, which is

$$O\left(\frac{D^2K(dn)^{3K}m^2R_{\max}^2}{\epsilon^2}\log(dn/\delta_1)\right) \tag{39}$$

Then we count all $K < J$ as mistakes, which is

$$O\left(\frac{J^2D^2(dn)^{3K}m^2R_{\max}^2}{\epsilon^2}\log(dn/\delta_1)\right) \tag{40}$$

Finally plugging in m (equation 29) results in the final sample complexity of

$$O\left(\frac{J^2D^2(dn)^{11J+10}|A|^{10}R_{\max}^{10}T_{\epsilon,D}^{16}}{\epsilon^{10}}\log^2(dnT_{\epsilon,D}R_{\max}|A|/\delta)\right) \tag{41}$$

where we use a union bound to bound the error for each K with $\delta_1 = \delta/n$. ■