Tesseract: Tensorised Actors for Multi-Agent Reinforcement Learning

Anuj Mahajan [1] **Mikayel Samvelyan** [2] **Lei Mao** [3] **Viktor Makoviychuk** [3] **Animesh Garg** [3] **Jean Kossaifi** [3]
Shimon Whiteson [1] **Yuke Zhu** [3] **Animashree Anandkumar** [3]

Abstract

Reinforcement Learning in large action spaces is a challenging problem. Cooperative multi-agent reinforcement learning (MARL) exacerbates matters by imposing various constraints on communication and observability. In this work, we consider the fundamental hurdle affecting both value-based and policy-gradient approaches: an exponential blowup of the action space with the number of agents. For value-based methods, it poses challenges in accurately representing the optimal value function. For policy gradient methods, it makes training the critic difficult and exacerbates the problem of the *lagging* critic. We show that from a learning theory perspective, both problems can be addressed by accurately representing the associated action-value function with a low-complexity hypothesis class. This requires accurately modelling the agent interactions in a sample efficient way. To this end, we propose a novel tensorised formulation of the Bellman equation. This gives rise to our method TESSERACT, which views the Q-function as a tensor whose modes correspond to the action spaces of different agents. Algorithms derived from TESSERACT decompose the Q-tensor across agents and utilise low-rank tensor approximations to model agent interactions relevant to the task. We provide PAC analysis for TESSERACT-based algorithms and highlight their relevance to the class of rich observation MDPs. Empirical results in different domains confirm TESSERACT's gains in sample efficiency predicted by the theory.

1. Introduction

Many real-world problems, such as swarm robotics and autonomous vehicles, can be formulated as multi-agent reinforcement learning (MARL) (Buşoniu et al., 2010) problems. MARL introduces several new challenges that do not arise in single-agent reinforcement learning (RL), including exponential growth of the action space in the number of agents. This affects multiple aspects of learning, such as credit assignment (Foerster et al., 2018), gradient variance (Lowe et al., 2017) and exploration (Mahajan et al., 2019). In addition, while the agents can typically be trained in a centralised manner, practical constraints on observability and communication after deployment imply that decision making must be decentralised, yielding the extensively studied setting of centralised training with decentralised execution (CTDE).

Recent work in CTDE-MARL can be broadly classified into value-based methods and actor-critic methods. Value-based methods (Sunehag et al., 2017; Rashid et al., 2018; Son et al., 2019; Wang et al., 2020a; Yao et al., 2019) typically enforce decentralisability by modelling the joint action Q-value such that the argmax over the joint action space can be tractably computed by local maximisation of per-agent utilities. However, constraining the representation of the Q-function can interfere with exploration, yielding provably suboptimal solutions (Mahajan et al., 2019). Actor-critic methods (Lowe et al., 2017; Foerster et al., 2018; Wei et al., 2018) typically use a centralised critic to estimate the gradient for a set of decentralised policies. In principle, actor-critic methods can satisfy CTDE without incurring suboptimality, but in practice their performance is limited by the accuracy of the critic, which is hard to learn given exponentially growing action spaces. This can exacerbate the problem of the *lagging* critic (Konda & Tsitsiklis, 2002). Moreover, unlike the single-agent setting, this problem cannot be fixed by increasing the critic's learning rate and number of training iterations. Similar to these approaches, an exponential blowup in the action space also makes it difficult to choose the appropriate class of models which strike the correct balance between expressibility and learnability for the given task.

In this work, we present new theoretical results that show how the aforementioned approaches can be improved such that they accurately represent the joint action-value function whilst keeping the complexity of the underlying hypothesis class low. This translates to accurate, sample efficient modelling of long-term agent interactions.

In particular, we propose TESSERACT (derived from "Ten-

[1]University of Oxford [2]University College London [3]NVIDIA. Correspondence to: Anuj Mahajan <anuj.mahajan@cs.ox.ac.uk>.

Proceedings of the 38^{th} International Conference on Machine Learning, PMLR 139, 2021.

sorised Actors"), a new framework that leverages tensors for MARL. Tensors are high dimensional analogues of matrices that offer rich insights into representing and transforming data. The main idea of TESSERACT is to view the output of a joint Q-function as a tensor whose modes correspond to the actions of the different agents. We thus formulate the Tensorised Bellman equation, which offers a novel perspective on the underlying structure of a multi-agent problem. In addition, it enables the derivation of algorithms that decompose the Q-tensor across agents and utilise low rank approximations to model relevant agent interactions.

Many real-world tasks (e.g., robot navigation) involve high dimensional observations but can be completely described by a low dimensional feature vector (e.g., a 2D map suffices for navigation). For value-based TESSERACT methods, maintaining a tensor approximation with rank matching the intrinsic task dimensionality[1] helps learn a compact approximation of the true Q-function (alternatively MDP-dynamics for model based methods). In this way, we can avoid the suboptimality of the learnt policy while remaining sample efficient. Similarly, for actor-critic methods, TESSERACT reduces the critic's learning complexity while retaining its accuracy, thereby mitigating the lagging critic problem. Thus, TESSERACT offers a natural spectrum for trading off accuracy with computational/sample complexity.

To gain insight into how tensor decomposition helps improve sample efficiency for MARL, we provide theoretical results for model-based TESSERACT algorithms and show that the underlying joint transition and reward functions can be efficiently recovered under a PAC framework (in samples polynomial in accuracy and confidence parameters). We also introduce a tensor-based framework for CTDE-MARL that opens new possibilities for developing efficient classes of algorithms. Finally, we explore the relevance of our framework to rich observation MDPs.

Our main contributions are:

1. A novel tensorised form of the Bellman equation;
2. TESSERACT, a method to factorise the action-value function based on tensor decomposition, which can be used for any factored action space;
3. PAC analysis and error bounds for model based TESSERACT that show an exponential gain in sample efficiency of $O(|U|^{n/2})$; and
4. Empirical results illustrating the advantage of TESSERACT over other methods and detailed techniques for making tensor decomposition work for deep MARL.

[1]We define intrinsic task dimensionality (ITD) as the minimum number of dimensions required to describe an environment

2. Background

Cooperative MARL settings In the most general setting, a fully cooperative multi-agent task can be modelled as a multi-agent partially observable MDP (M-POMDP) (Messias et al., 2011). An M-POMDP is formally defined as a tuple $G = \langle S, U, P, r, Z, O, n, \gamma \rangle$. S is the state space of the environment. At each time step t, every agent $i \in \mathcal{A} \equiv \{1, ..., n\}$ chooses an action $u^i \in U$ which forms the joint action $\mathbf{u} \in \mathbf{U} \equiv U^n$. $P(s'|s, \mathbf{u}) : S \times \mathbf{U} \times S \to [0, 1]$ is the state transition function. $r(s, \mathbf{u}) : S \times \mathbf{U} \to [0, 1]$ is the reward function shared by all agents and $\gamma \in [0, 1)$ is the discount factor. An M-POMDP is *partially observable* (Kaelbling et al., 1998): each agent does not have access to the full state and instead samples observations $z \in Z$ according to observation distribution $O(s) : S \to \mathcal{P}(Z)$. The action-observation history for an agent i is $\tau^i \in T \equiv (Z \times U)^*$. We use u^{-i} to denote the action of all the agents other than i and similarly for the policies π^{-i}. Settings where the agents cannot exchange their action-observation histories with others and must condition their policy solely on local trajectories, $\pi^i(u^i|\tau^i) : T \times U \to [0, 1]$, are referred to as a decentralised partially observable MDP (Dec-POMDP) (Oliehoek & Amato, 2016). When the observations have additional structure, namely the joint observation space is partitioned w.r.t. S, i.e., $\forall s_1, s_2 \in S \wedge z \in Z, P(z|s_1) > 0 \wedge s_1 \neq s_2 \implies P(z|s_2) = 0$, we classify the problem as a multi-agent richly observed MDP (M-ROMDP) (Azizzadenesheli et al., 2016). For both M-POMDP and M-ROMDP, we assume $|Z| >> |S|$, thus for this work, we assume a setting with no information loss due to observation but instead, redundancy across different observation dimensions. Such is the case for many real world tasks like 2D robot navigation using observation data from different sensors. Finally, when the observation function is a bijective map $O : S \to Z$, we refer to the scenario as a multi-agent MDP (MMDP) (Boutilier, 1996), which can simply be denoted by the tuple : $\langle S, U, P, r, n, \gamma \rangle$. Fig. 1 gives the relation between different scenarios for the cooperative setting. For ease of exposition, we present our theoretical results for the MMDP case, though they can easily be extended to other cases by incurring additional sample complexity.

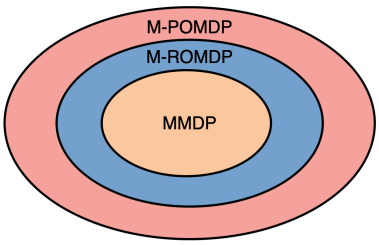


Figure 1. Different settings in MARL

The joint *action-value function* given a policy π is defined as: $Q^\pi(s_t, \mathbf{u}_t) = \mathbb{E}_{s_{t+1:\infty}, \mathbf{u}_{t+1:\infty}} \left[\sum_{k=0}^{\infty} \gamma^k r_{t+k} | s_t, \mathbf{u}_t\right]$. The

goal is to find the optimal policy π^* corresponding to the optimal action value function Q^*. For the special learning scenario called Centralised Training with Decentralised Execution (CTDE), the learning algorithm has access to the action-observation histories of all agents and the full state during training phase. However, each agent can only condition on its own local action-observation history τ^i during the decentralised execution phase.

Reinforcement Learning Methods Both value-based and actor-critic methods for reinforcement learning (RL) rely on an estimator for the action-value function Q^π given a target policy π. Q^π satisfies the (scalar)-Bellman expectation equation: $Q^\pi(s,\mathbf{u}) = r(s,\mathbf{u}) + \gamma \mathbb{E}_{s',\mathbf{u}'}[Q^\pi(s',\mathbf{u}')]$, which can equivalently be written in vectorised form as:

$$Q^\pi = R + \gamma P^\pi Q^\pi, \tag{1}$$

where R is the mean reward vector of size S, P^π is the transition matrix. The operation on RHS $\mathcal{T}^\pi(\cdot) \triangleq R + \gamma P^\pi(\cdot)$ is the Bellman expectation operator for the policy π. In Section 3 we generalise Eq. (1) to a novel tensor form suitable for high-dimensional and multi-agent settings. For large state-action spaces function approximation is used to estimate Q^π. A parametrised approximation Q^ϕ is usually trained using the bootstrapped target objective derived using the samples from π by minimising the mean squared temporal difference error: $\mathbb{E}_\pi[(r(s,\mathbf{u}) + \gamma Q^\phi(s',\mathbf{u}') - Q^\phi(s,\mathbf{u}))^2]$. Value based methods use the Q^π estimate to derive a behaviour policy which is iteratively improved using the policy improvement theorem (Sutton & Barto, 2011). Actor-critic methods seek to maximise the mean expected payoff of a policy π_θ given by $\mathcal{J}_\theta = \int_S \rho^\pi(s) \int_\mathbf{U} \pi_\theta(\mathbf{u}|\mathbf{s}) Q^\pi(s,\mathbf{u}) d\mathbf{u} ds$ using gradient ascent on a suitable class of stochastic policies parametrised by θ, where $\rho^\pi(s)$ is the stationary distribution over the states. Updating the policy parameters in the direction of the gradient leads to policy improvement. The gradient of the above objective is $\nabla \mathcal{J}_\theta = \int_S \rho^\pi(s) \int_\mathbf{U} \nabla \pi_\theta(\mathbf{u}|\mathbf{s}) Q^\pi(s,\mathbf{u}) d\mathbf{u} ds$ (Sutton et al., 2000). An approximate action-value function based critic Q^ϕ is used when estimating the gradient as we do not have access to the true Q-function. Since the critic is learnt using finite number of samples, it may deviate from the true Q-function, potentially causing incorrect policy updates; this is called the *lagging critic* problem. The problem is exacerbated in multi-agent setting where state-action spaces are very large.

Tensor Decomposition Tensors are high dimensional analogues of matrices and tensor methods generalize matrix algebraic operations to higher orders. Tensor decomposition, in particular, generalizes the concept of low-rank matrix factorization (Kolda & Bader, 2009; Janzamin et al., 2020). In the rest of this paper, we use $\hat{\cdot}$ to denote tensors. Formally, an order n tensor $\hat{T}$ has n index sets $I_j, \forall j \in \{1..n\}$ and has elements $T(e), \forall e \in \times_\mathcal{I} I_j$ taking values in a given set $\mathcal{S}$, where $\times$ is the set cross product and we denote the set of index sets by $\mathcal{I}$. Each dimension $\{1..n\}$ is also called a mode. An elegant way of representing tensors and associated operations is via tensor diagrams as shown in Fig. 2. Tensor contraction generalizes the concept of matrix with matrix multiplication. For any two tensors $\hat{T}^1$ and $\hat{T}^2$ with $\mathcal{I}_\cap = \mathcal{I}^1 \cap \mathcal{I}^2$ we define the contraction operation as $\hat{T} = \hat{T}^1 \odot \hat{T}^2$ with $\hat{T}(e_1, e_2) = \sum_{e \in \times_{\mathcal{I}_\cap} I_j} \hat{T}^1(e_1, e) \cdot \hat{T}^2(e_2, e), e_i \in \times_{\mathcal{I}^i \setminus \mathcal{I}_\cap} I_j$. The contraction operation is associative and can be extended to an arbitrary number of tensors. Using this building block, we can define tensor decompositions, which factorizes a (low-rank) tensor in a compact form. This can be done with various decompositions (Kolda & Bader, 2009), such as Tucker, Tensor-Train (also known as Matrix-Product-State), or CP (for Canonical-Polyadic). In this paper, we focus on the latter, which we briefly introduce here. Just as a matrix can be factored as a sum of rank-1 matrices (each being an outer product of vectors), a tensor can be factored as a sum of rank-1 tensors, the latter being an outer product of vectors. The number of vectors in the outer product is equal to the rank of the tensor, and the number of terms in the sum is called the *rank of the decomposition* (sometimes also called CP-rank). Formally, a tensor $\hat{T}$ can be factored using a (rank–k) CP decomposition into a sum of k vector outer products (denoted by $\otimes$), as,

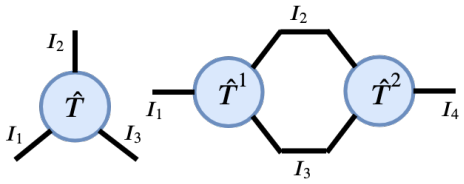


Figure 2. Left: Tensor diagram for an order 3 tensor $\hat{T}$. Right: Contraction between $\hat{T}^1$,$\hat{T}^2$ on common index sets I_2, I_3.

$$\hat{T} = \sum_{r=1}^{k} w_r \otimes^n u_r^i, i \in \{1..n\}, ||u_r^i||_2 = 1. \tag{2}$$

3. Methodology

3.1. Tensorised Bellman equation

In this section, we provide the basic framework for Tesseract. We focus here on the discrete action space. The extension for continuous actions is similar and is deferred to Appendix B.2 for clarity of exposition.

Proposition 1. *Any real-valued function f of n arguments $(x_1..x_n)$ each taking values in a finite set $x_i \in \mathcal{D}_i$ can be represented as a tensor $\hat{f}$ with modes corresponding to the domain sets $\mathcal{D}_i$ and entries $\hat{f}(x_1..x_n) = f(x_1..x_n)$.*

Given a multi-agent problem $G = \langle S, U, P, r, Z, O, n, \gamma \rangle$, let $\mathcal{Q} \triangleq \{Q : S \times U^n \to \mathbb{R}\}$ be the set of real-valued functions on the state-action space. We are interested in the *curried* (Barendregt, 1984) form $Q : S \to U^n \to \mathbb{R}, Q \in \mathcal{Q}$ so that $Q(s)$ is an order n tensor (We use functions and tensors interchangeably where it is clear from context). Algorithms in Tesseract operate directly on the curried form and preserve the structure implicit in the output tensor. (Currying in the context of tensors implies fixing the value of some index. Thus, Tesseract-based methods keep action indices free and fix only state-dependent indices.)

We are now ready to present the tensorised form of the Bellman equation shown in Eq. (1). Fig. 3 gives the equation where $\hat{I}$ is the identity tensor of size $|S| \times |S| \times |S|$. The dependence of the action-value tensor $\hat{Q}^\pi$ and the policy tensor $\hat{U}^\pi$ on the policy is denoted by superscripts π. The novel **Tensorised Bellman equation** provides a theoretically justified foundation for the approximation of the joint Q-function, and the subsequent analysis (Theorems 1-3) for learning using this approximation.

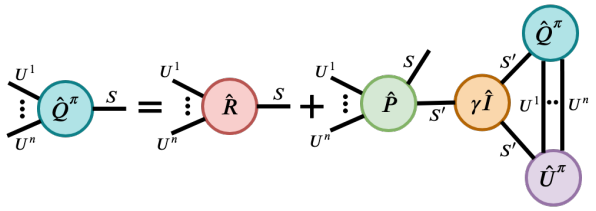


Figure 3. **Tensorised Bellman Equation** for n agents. There is an edge for each agent $i \in \mathcal{A}$ in the corresponding nodes $\hat{Q}^\pi, \hat{U}^\pi, \hat{R}, \hat{P}$ with the index set U^i.

3.2. TESSERACT Algorithms

For any $k \in \mathbb{N}$ let $\mathcal{Q}_k \triangleq \{Q : Q \in \mathcal{Q} \wedge rank(Q(\cdot, s)) \leq k, \forall s \in S\}$. Given any policy π we are interested in projecting Q^π to $\mathcal{Q}_k$ using the projection operator $\Pi_k(\cdot) = \arg\min_{Q \in \mathcal{Q}_k} || \cdot - Q||_{\pi,F}$. where $||X||_{\pi,F} \triangleq \mathbb{E}_{s \sim \rho^\pi(s)}[||X(s)||_F]$ is the weighted Frobenius norm w.r.t. policy visitation over states. Thus a simple planning based algorithm for rank k TESSERACT would involve starting with an arbitrary Q_0 and successively applying the Bellman operator $\mathcal{T}^\pi$ and the projection operator Π_k so that $Q_{t+1} = \Pi_k \mathcal{T}^\pi Q_t$.

As we show in Theorem 1, constraining the underlying tensors for dynamics and rewards $(\hat{P}, \hat{R})$ is sufficient to bound the CP-rank of $\hat{Q}$. From this insight, we propose a model-based RL version for TESSERACT in Algorithm 1. The algorithm proceeds by estimating the underlying MDP dynamics using the sampled trajectories obtained by executing the behaviour policy $\pi = (\pi^i)_1^n$ (factorisable across agents) satisfying Theorem 2. Specifically, we use a rank k approximate CP-Decomposition to calculate the model dynamics R, P as we show in Section 4. Next π is evaluated using the estimated dynamics, which is followed by policy improvement, Algorithm 1 gives the pseudocode for the model-based setting. The termination and policy improvement decisions in Algorithm 1 admit a wide range of choices used in practice in the RL community. Example choices for internal iterations which broadly fall under approximate policy iteration include: 1) Fixing the number of applications of Bellman operator 2) Using norm of difference between consecutive Q estimates etc., similarly for policy improvement several options can be used like ϵ-greedy (for Q derived policy), policy gradients (parametrized policy) (Sutton & Barto, 2011)

Algorithm 1 Model-based Tesseract

```
Initialise rank k, π = (π^i)_1^n and Q̂: Theorem 2
Initialise model parameters P̂, R̂
Learning rate ← α, D ← {}
for each episodic iteration i do
    Do episode rollout τ_i = {(s_t, u_t, r_t, s_{t+1})_0^L} using π
    D ← D ∪ {τ_i}
    Update P̂, R̂ using CP-Decomposition on moments from D (Theorem 2)
    for each internal iteration j do
        Q̂ ← T^π Q̂
    end for
    Improve π using Q̂
end for
Return π, Q̂
```

For large state spaces where storage and planning using model parameters is computationally difficult (they are $\mathcal{O}(kn|U||S|^2)$ in number), we propose a model-free approach using a deep network where the rank constraint on the Q-function is directly embedded into the network architecture. Fig. 4 gives the general network architecture for this approach and Algorithm 2 the associated pseudo-code. Each agent in Fig. 4 has a policy network parameterized by θ which is used to take actions in a decentralised manner. The observations of the individual agents along with the actions are fed through representation function g_ϕ whose output is a set of k unit vectors of dimensionality $|U|$ corresponding to each rank. The output $g_{\phi,r}(s^i)$ corresponding to each agent i for factor r can be seen as an action-wise contribution to the joint utility from the agent corresponding to that factor. The joint utility here is a product over individual agent utilities. For partially observable settings, an additional RNN layer can be used to summarise agent trajectories. The joint action-value estimate of the tensor

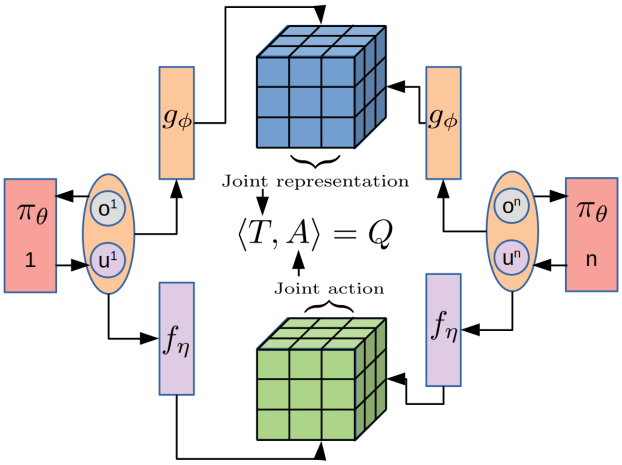


Figure 4. Tesseract architecture

$\hat{Q}(s)$ by the centralized critic is:

$$\hat{Q}(s) \approx T = \sum_{r=1}^{k} w_r \otimes^n g_{\phi,r}(s^i), i \in \{1..n\}, \quad (3)$$

where the weights w_r are learnable parameters exclusive to the centralized learner. In the case of value based methods where the policy is implicitly derived from utilities, the policy parameters θ are merged with ϕ. The network architecture is agnostic to the type of the action space (discrete/continuous) and the action-value corresponding to a particular joint-action $(u^1..u^n)$ is the inner product $\langle T, A \rangle$ where $A = \otimes^n u^i$ (This reduces to indexing using joint action in Eq. (3) for discrete spaces). More representational capacity can be added to the network by creating an abstract representation for actions using f_η, which can be any arbitrary monotonic function (parametrised by η) of vector output of size $m \geq |U|$ and preserves relative order of utilities across actions; this ensures that the optimal policy is learnt as long as it belongs to the hypothesis space. In this case $A = \otimes^n f_\eta(u^i)$ and the agents also carry a copy of f_η during the execution phase. Furthermore, the inner product $\langle T, A \rangle$ can be computed efficiently using the property

$$\langle T, A \rangle = \sum_{r=1}^{k} w_r \prod_{1}^{n} \langle f_\eta(u^i) g_{\phi,r}(s^i) \rangle, i \in \{1..n\}$$

which is $O(nkm)$ whereas a naive approach involving computation of the tensors first would be $O(km^n)$. Training the Tesseract-based Q-network involves minimising the squared TD loss (Sutton & Barto, 2011):

$$\mathcal{L}_{TD}(\phi, \eta) = \mathbb{E}_\pi[(Q(\mathbf{u}_t, s_t; \phi, \eta) - [r(\mathbf{u}_t, s_t) + \gamma Q(\mathbf{u}_{t+1}, s_{t+1}; \phi^-, \eta^-)])^2],$$

where ϕ^-, η^- are target parameters. Policy updates involve gradient ascent w.r.t. to the policy parameters θ on the objective $\mathcal{J}_\theta = \int_S \rho^\pi(s) \int_\mathbf{U} \pi_\theta(\mathbf{u}|\mathbf{s}) Q^\pi(s, \mathbf{u}) d\mathbf{u} ds$. More sophisticated targets can be used to reduce the policy gradient variance (Greensmith et al., 2004; Zhao et al., 2016) and propagate rewards efficiently (Sutton, 1988). Note that Algorithm 2 does not require the individual-global maximisation principle (Son et al., 2019) typically assumed by value-based MARL methods in the CTDE setting, as it is an actor-critic method. In general, any form of function approximation and compatible model-free approach can be interleaved with Tesseract by appropriate use of the projection function Π_k.

Algorithm 2 Model-free Tesseract

Initialise rank k, parameter vectors θ, ϕ, η
Learning rate $\leftarrow \alpha$,$\mathcal{D} \leftarrow \{\}$
for each episodic iteration i **do**
  Do episode rollout $\tau_i = \{(s_t, \mathbf{u}_t, r_t, s_{t+1})_0^L\}$ using π_θ
  $\mathcal{D} \leftarrow \mathcal{D} \cup \{\tau_i\}$
  Sample batch $\mathcal{B} \subseteq \mathcal{D}$.
  Compute empirical estimates for $\mathcal{L}_{TD}, \mathcal{J}_\theta$
  $\phi \leftarrow \phi - \alpha \nabla_\phi \mathcal{L}_{TD}$ (Rank k projection step)
  $\eta \leftarrow \eta - \alpha \nabla_\eta \mathcal{L}_{TD}$ (Action representation update)
  $\theta \leftarrow \theta + \alpha \nabla_\theta \mathcal{J}_\theta$ (Policy update)
end for
Return $\pi, \hat{Q}$

3.3. Why Tesseract?

As discussed in Section 1, $Q(s)$ is an object of prime interest in MARL. Value based methods (Sunehag et al., 2017; Rashid et al., 2018; Yao et al., 2019) that directly approximate the optimal action values Q^* place constraints on $Q(s)$ such that it is a monotonic combination of agent utilities. In terms of Tesseract this directly translates to finding the best projection constraining $Q(s)$ to be rank one (Appendix B.1). Similarly, the following result demonstrates containment of action-value functions representable by FQL(Chen et al., 2018) which uses a learnt inner product to model pairwise agent interactions (**proof and additional results in Appendix B.1**):.

Proposition 2. *The set of joint Q-functions representable by FQL is a subset of that representable by* TESSERACT.

MAVEN (Mahajan et al., 2019) illustrates how rank 1 projections can lead to insufficient exploration and provides a method to avoid suboptimality by using mutual information (MI) to learn a diverse set of rank 1 projections that correspond to different joint behaviours. In Tesseract, this can simply be achieved by finding the best approximation constraining $Q(s)$ to be rank k. Moreover, the CP-decomposition problem, being a product form (Eq. (2)), is well posed, whereas in MAVEN, the problem form is

$\hat{T} = \sum_{r=1}^{k} w_r \oplus^n u_r^i, i \in \{1..n\}, ||u_r^i||_2 = 1$, which requires careful balancing of different factors $\{1..k\}$ using MI as otherwise all factors collapse to the same estimate. The above improvements are equally important for the critic in actor-critic frameworks. Note that TESSERACT is complete in the sense that every possible joint Q-function is representable by it given sufficient approximation rank. This follows as every possible Q-tensor can be expressed as linear combination of one-hot tensors (which form a basis for the set).

Many real world problems have high-dimensional observation spaces that are encapsulated in an underlying low dimensional latent space that governs the transition and reward dynamics (Azizzadenesheli et al., 2016). For example, in the case of robot navigation, the observation is high dimensional visual and sensory input but solving the underlying problem requires only knowing the 2D position. Standard RL algorithms that do not address modelling the latent structure in such problems typically incur poor performance and intractability. In Section 4 we show how Tesseract can be leveraged for such scenarios. Finally, projection to a low rank offers a natural way of regularising the approximate Q-functions and makes them easier to learn, which is important for making value function approximation amenable to multi-agent settings. Specifically for the case of actor-critic methods, this provides a natural way to make the critic learn more quickly. Additional discussion about using Tesseract for continuous action spaces can be found in Appendix B.2.

4. Analysis

In this section we provide a PAC analysis of model-based Tesseract (Algorithm 1). We focus on the MMDP setting (Section 2) for the simplicity of notation and exposition; guidelines for other settings are provided in Appendix A.

The objective of the analysis is twofold: Firstly it provides concrete quantification of the sample efficiency gained by model-based policy evaluation. Secondly, it provides insights into how Tesseract can similarly reduce sample complexity for model-free methods. **Proofs for the results stated can be found in Appendix A**. We begin with the assumptions used for the analysis:

Assumption 1. *For the given MMDP $G = \langle S, U, P, r, n, \gamma \rangle$, the reward tensor $\hat{R}(s), \forall s \in S$ has bounded rank $k_1 \in \mathbb{N}$.*

Intuitively, a small k_1 in Assumption 1 implies that the reward is dependent only on a small number of intrinsic factors characterising the actions.

Assumption 2. *For the given MMDP $G = \langle S, U, P, r, n, \gamma \rangle$, the transition tensor $\hat{P}(s, s'), \forall s, s' \in S$ has bounded rank $k_2 \in \mathbb{N}$.*

Intuitively a small k_2 in Assumption 2 implies that only a small number of intrinsic factors characterising the actions lead to meaningful change in the joint state. Assumption 1-2 always hold for a finite MMDP as CP-rank is upper bounded by $\Pi_{j=1}^{n} |U_j|$, where U_j are the action sets.

Assumption 3. *The underlying MMDP is ergodic for any policy π so that there is a stationary distribution ρ^π.*

Next, we define coherence parameters, which are quantities of interest for our theoretical results: for reward decomposition $\hat{R}(s) = \sum_r w_{r,s} \otimes^n v_{r,i,s}$, let $\mu_s = \sqrt{n} \max_{i,r,j} |v_{r,i,s}(j)|$, $w_s^{\max} = \max_{i,r} w_{r,s}$, $w_s^{\min} = \min_{i,r} w_{r,s}$. Similarly define the corresponding quantities for $\mu_{s,s'}, w_{s,s'}^{\max}, w_{s,s'}^{\min}$ for transition tensors $\hat{P}(s, s')$. A low coherence implies that the tensor's mass is evenly spread and helps bound the possibility of never seeing an entry with very high mass (large absolute value of an entry).

Theorem 1. *For a finite MMDP the action-value tensor satisfies $rank(\hat{Q}^\pi(s)) \leq k_1 + k_2|S|, \forall s \in S, \forall \pi$.*

Proof. We first unroll the Tensor Bellman equation in Fig. 3. The first term $\hat{R}$ has bounded rank k_1 by Assumption 1. Next, each contraction term on the RHS is a linear combination of $\{\hat{P}(s, s')\}_{s' \in S}$ each of which has bounded rank k_2 (Assumption 2). The result follows from the sub-additivity of CP-rank. □

Theorem 1 implies that for approximations with enough factors, policy evaluation converges:

Corollary 1.1. *For all $k \geq k_1 + k_2|S|$, the procedure $Q_{t+1} \leftarrow \Pi_k \mathcal{T}^\pi Q_t$ converges to Q^π for all Q_0, π.*

Corollary 1.1 is especially useful for the case of M-POMDP and M-ROMDP with $|Z| >> |S|$, i.e., where the intrinsic state space dimensionality is small in comparison to the dimensionality of the observations (like robot navigation Section 3.3). In these cases the Tensorised Bellman equation Fig. 3 can be augmented by padding the transition tensor $\hat{P}$ with the observation matrix and the lower bound in Corollary 1.1 can be improved using the intrinsic state dimensionality.

We next give a PAC result on the number of samples required to infer the reward and state transition dynamics for finite MDPs with high probability using sufficient approximate rank $k \geq k_1, k_2$:

Theorem 2 (Model based estimation of $\hat{R}, \hat{P}$ error bounds)**.** *Given any $\epsilon > 0, 1 > \delta > 0$, for a policy π with the policy*

tensor satisfying $\pi(\mathbf{u}|s) \geq \Delta$*, where*

$$\Delta = \max_s \frac{C_1 \mu_s^6 k^5 (w_s^{max})^4 \log(|U|)^4 \log(3k||R(s)||_F/\epsilon)}{|U|^{n/2}(w_s^{min})^4}$$

and C_1 *is a problem dependent positive constant. There exists* N_0 *which is* $O(|U|^{\frac{n}{2}})$ *and polynomial in* $\frac{1}{\delta}, \frac{1}{\epsilon}, k$ *and relevant spectral properties of the underlying MDP dynamics such that for samples* $\geq N_0$*, we can compute the estimates* $\bar{R}(s), \bar{P}(s,s')$ *such that w.p.* $\geq 1-\delta$*,* $||\bar{R}(s) - \hat{R}(s)||_F \leq \epsilon, ||\bar{P}(s,s') - \hat{P}(s,s')||_F \leq \epsilon, \forall s, s' \in S$*.*

Theorem 2 gives the relation between the order of the number of samples required to estimate dynamics and the tolerance for approximation. Theorem 2 states that aside from allowing efficient PAC learning of the reward and transition dynamics of the multi-agent MDP, Algorithm 1 requires only $O(|U|^{\frac{n}{2}})$ to do so, which is a vanishing fraction of $|U|^n$, the total number of joint actions in any given state. This also hints at why a tensor based approximation of the Q-function helps with sample efficiency. Methods that do not use the tensor structure typically use $O(|U|^n)$ samples. The bound is also useful for off-policy scenarios, where only the behaviour policy needs to satisfy the bound. Given the result in Theorem 2, it is natural to ask what is the error associated with computing the action-values of a policy using the estimated transition and reward dynamics. We address this in our next result, but first we present a lemma bounding the total variation distance between the estimated and true transition distributions:

Lemma 1. *For transition tensor estimates satisfying* $||\bar{P}(s,s') - \hat{P}(s,s')||_F \leq \epsilon$*, we have for any given state-action pair* (s,a)*, the distribution over the next states follows:* $TV(P'(\cdot|s,a), P(\cdot|s,a)) \leq \frac{1}{2}(|1-f| + f|S|\epsilon)$ *where* $\frac{1}{1+\epsilon|S|} \leq f \leq \frac{1}{1-\epsilon|S|}$*, where* TV *is the total variation distance. Similarly for any policy* π*,* $TV(\bar{P}_\pi(\cdot|s), P_\pi(\cdot|s)), TV(\bar{P}_\pi(s',a'|s), P_\pi(s',a'|s)) \leq \frac{1}{2}(|1-f| + f|S|\epsilon)$

We now bound the error of model-based evaluation using approximate dynamics in Theorem 3. The first component on the RHS of the upper bound comes from the tensor analysis of the transition dynamics, whereas the second component can be attributed to error propagation for the rewards.

Theorem 3 (Error bound on policy evaluation)**.** *Given a behaviour policy* π_b *satisfying the conditions in Theorem 2 and executed for steps* $\geq N_0$*, for any policy* π *the model based policy evaluation* $Q^\pi_{\bar{P},\bar{R}}$ *satisfies:*

$$|Q^\pi_{P,R}(s,a) - Q^\pi_{\bar{P},\bar{R}}(s,a)| \leq (|1-f| + f|S|\epsilon)\frac{\gamma}{2(1-\gamma)^2} + \frac{\epsilon}{1-\gamma}, \forall (s,a) \in S \times U^n$$

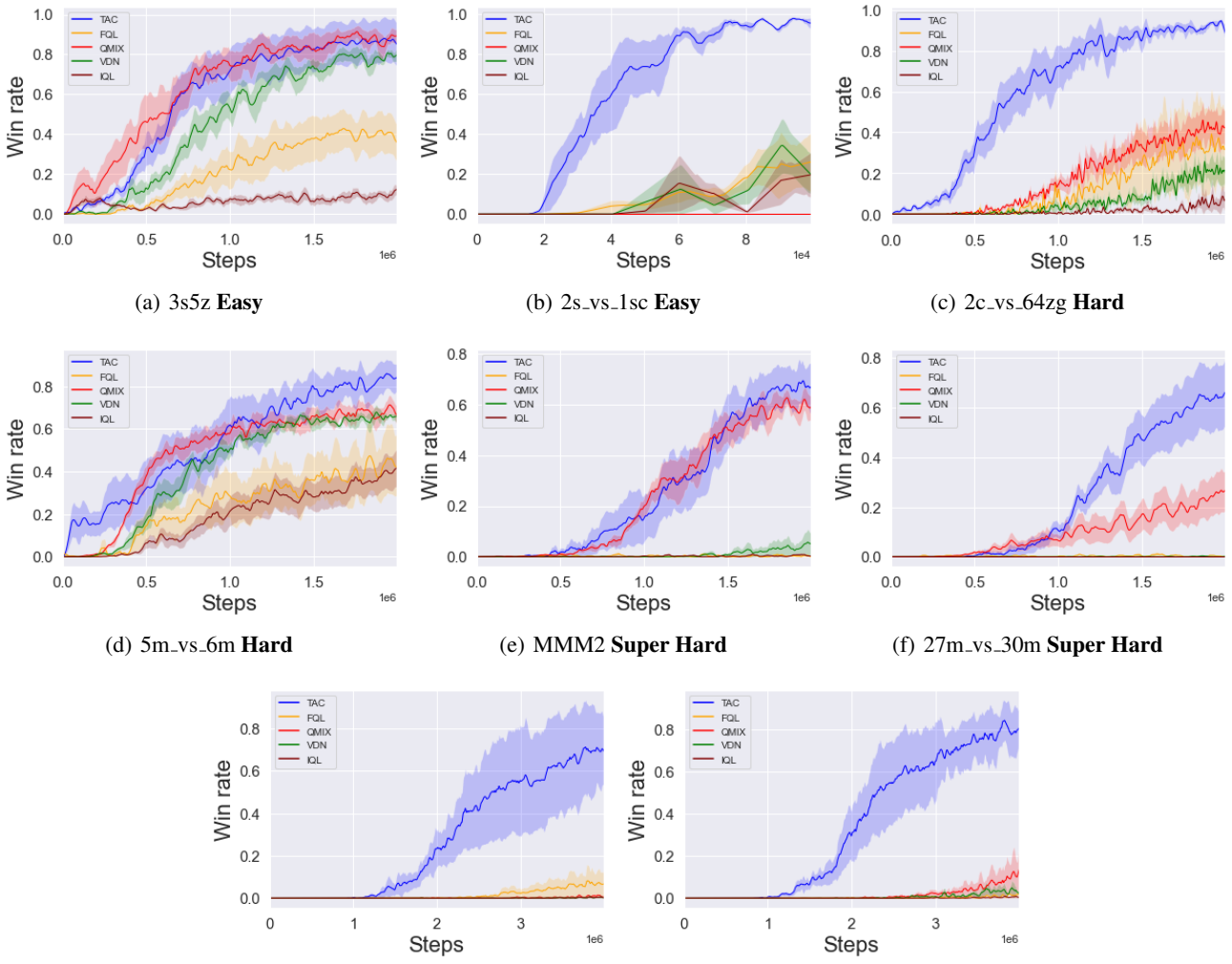
where f *is as defined in Lemma 1.*

Additional theoretical discussion can be found in Appendix B.3

5. Experiments

In this section we present the empirical results on the StarCraft domain. Experiments for a more didactic domain of Tensor games can be found in Appendix C.3. We use the model-free version of TESSERACT (Algorithm 2) for all the experiments.

StarCraft II We consider a challenging set of cooperative scenarios from the StarCraft Multi-Agent Challenge (SMAC) (Samvelyan et al., 2019). Scenarios in SMAC have been classified as **Easy, Hard and Super-hard** according to the performance of exiting algorithms on them. We compare TESSERACT (TAC in plots) to, QMIX (Rashid et al., 2018), VDN (Sunehag et al., 2017), FQL (Chen et al., 2018), and IQL (Tan, 1993). VDN and QMIX use monotonic approximations for learning the Q-function. FQL uses a pairwise factorized model to capture effects of agent interactions in joint Q-function, this is done by learning an inner product space for summarising agent trajectories. IQL ignores the multi-agentness of the problem and learns an independent per agent policy for the resulting non-stationary problem. Fig. 5 gives the win rate of the different algorithms averaged across five random runs. Fig. 5(c) features 2c_vs_64zg, a hard scenario that contains two allied agents but 64 enemy units (the largest in the SMAC domain) making the action space of the agents much larger than in the other scenarios. TESSERACT gains a huge lead over all the other algorithms in just one million steps. For the asymmetric scenario of 5m_vs_6m Fig. 5(d), TESSERACT, QMIX, and VDN learn effective policies, similar behavior occurs in the heterogeneous scenarios of 3s5z Fig. 5(a) and MMM2Fig. 5(e) with the exception of VDN for the latter. In 2s_vs_1sc in Fig. 5(b), which requires a 'kiting' strategy to defeat the spine crawler, TESSERACT learns an optimal policy in just 100k steps. In the **super-hard** scenario of 27m_vs_30m Fig. 5(f) having largest ally team of 27 marines, TESSERACT again shows improved sample efficiency; this map also shows TESSERACT's ability to scale with the number of agents. Finally in the **super-hard** scenarios of 6 hydralisks vs 8 zealots Fig. 5(g) and Corridor Fig. 5(h) which require careful exploration, TESSERACT is the only algorithm which is able to find a good policy. We observe that IQL doesn't perform well on any of the maps as it doesn't model agent interactions/non-stationarity explicitly. FQL loses performance possibly because modelling just pairwise interactions with a single dot product might not be expressive enough for joint-Q. Finally, VDN and QMIX are unable to perform well on many of the challenging scenarios possibly due to the monotonic approximation affecting the exploration adversely (Mahajan et al., 2019). Additional plots and exper-



(a) 3s5z **Easy** (b) 2s_vs_1sc **Easy** (c) 2c_vs_64zg **Hard**

(d) 5m_vs_6m **Hard** (e) MMM2 **Super Hard** (f) 27m_vs_30m **Super Hard**

(g) 6h_vs_8z **Super Hard** (h) Corridor **Super Hard**

Figure 5. Performance of different algorithms on different SMAC scenarios: TAC, QMIX, VDN, FQL, IQL.

iment details can be found in Appendix C.1 with **comparison with other baselines in Appendix C.1.1** including QPLEX(Wang et al., 2020a), QTRAN(Son et al., 2019), HQL(Matignon et al., 2007), COMA(Foerster et al., 2018) . We detail the techniques used for stabilising the learning of tensor decomposed critic in Appendix C.2.

6. Related Work

Previous methods for modelling multi-agent interactions include those that use coordination graph methods for learning a factored joint action-value estimation (Guestrin et al., 2002a;b; Bargiacchi et al., 2018), however typically require knowledge of the underlying coordination graph. To handle the exponentially growing complexity of the joint action-value functions with the number of agents, a series of value-based methods have explored different forms of value function factorisation. VDN (Sunehag et al., 2017) and QMIX (Rashid et al., 2018) use monotonic approximation with latter using a mixing network conditioned on global state. QTRAN (Son et al., 2019) avoids the weight constraints imposed by QMIX by formulating multi-agent learning as an optimisation problem with linear constraints and relaxing it with L2 penalties. MAVEN (Mahajan et al., 2019) learns a diverse ensemble of monotonic approximations by conditioning agent Q-functions on a latent space which helps overcome the detrimental effects of QMIX's monotonicity constraint on exploration. Similarly, Uneven (Gupta et al., 2020) uses universal successor features for efficient exploration in the joint action space. Qatten (Yang et al., 2020) makes use of a multi-head attention mechanism to decompose Q_{tot} into a linear combination of per-agent terms. RODE (Wang et al., 2020b) learns an action effect based role decomposition for sample efficient learning.

Policy gradient methods, on the other hand, often utilise the actor-critic framework to cope with decentralisation. MADDPG (Lowe et al., 2017) trains a centralised critic for each agent. COMA (Foerster et al., 2018) employs a centralised critic and a counterfactual advantage function. These actor-critic methods, however, suffer from poor sample efficiency compared to value-based methods and often converge to sub-optimal local minima. While sample efficiency has been an important goal for single agent reinforcement learning methods (Mahajan & Tulabandhula, 2017a;b; Kakade, 2003; Lattimore et al., 2013), in this work we shed light on attaining sample efficiency for cooperative multi-agent systems using low rank tensor approximation.

Tensor methods have been used in machine learning, in the context of learning latent variable models (Anandkumar et al., 2014), signal processing (Sidiropoulos et al., 2017), deep learning and computer vision (Panagakis et al., 2021). They provide powerful analytical tools that have been used for various applications, including the theoretical analysis of deep neural networks (Cohen et al., 2016). Model compression using tensors (Cheng et al., 2017) has recently gained momentum owing to the large sizes of deep neural nets. Using tensor decomposition within deep networks, it is possible to both compress and speed them up (Cichocki et al., 2017; Kossaifi et al., 2019). They allow generalization to higher orders (Kossaifi et al., 2020) and have also been used for multi-task learning and domain adaptation (Bulat et al., 2020). In contrast to prior work on value function factorisation, TESSERACT provides a natural spectrum for approximation of action-values based on the rank of approximation and provides theoretical guarantees derived from tensor analysis. Multi-view methods utilising tensor decomposition have previously been used in the context of partially observable single-agent RL (Azizzadenesheli et al., 2016; Azizzadenesheli, 2019). There the goal is to efficiently infer the underlying MDP parameters for planning under rich observation settings (Krishnamurthy et al., 2016). Similarly (Bromuri, 2012) use four dimensional factorization to generalise across Q-tables whereas here we use them for modelling interactions across multiple agents.

7. Conclusions & Future Work

We introduced TESSERACT, a novel framework utilising the insight that the joint action value function for MARL can be seen as a tensor. TESSERACT provides a means for developing new sample efficient algorithms and obtain essential guarantees about convergence and recovery of the underlying dynamics. We further showed novel PAC bounds for learning under the framework using model-based algorithms. We also provided a model-free approach to implicitly induce low rank tensor approximation for better sample efficiency and showed that it outperforms current state of art methods. There are several interesting open questions to address in future work, such as convergence and error analysis for rank insufficient approximation, and analysis of the learning framework under different types of tensor decompositions like Tucker and tensor-train (Kolda & Bader, 2009) and augmentations such as tensor dropout (Kolbeinsson et al., 2021).

Acknowledgements

AM is funded by the J.P. Morgan A.I. fellowship. Part of this work was done during AM's internship at NVIDIA. This project has received funding from the European Research Council under the European Union's Horizon 2020 research and innovation programme (grant agreement number 637713). The experiments were made possible by generous equipment grant from NVIDIA.

References

Anandkumar, A., Hsu, D., and Kakade, S. M. A method of moments for mixture models and hidden markov models. In *Conference on Learning Theory*, pp. 33–1, 2012.

Anandkumar, A., Ge, R., Hsu, D., Kakade, S. M., and Telgarsky, M. Tensor decompositions for learning latent variable models. *Journal of Machine Learning Research*, 15:2773–2832, 2014.

Azizzadenesheli, K. *Reinforcement Learning in Structured and Partially Observable Environments*. PhD thesis, UC Irvine, 2019.

Azizzadenesheli, K., Lazaric, A., and Anandkumar, A. Reinforcement learning in rich-observation mdps using spectral methods. *arXiv preprint arXiv:1611.03907*, 2016.

Barendregt, H. P. Introduction to lambda calculus. 1984.

Bargiacchi, E., Verstraeten, T., Roijers, D., Nowé, A., and Hasselt, H. Learning to coordinate with coordination graphs in repeated single-stage multi-agent decision problems. In *International conference on machine learning*, pp. 482–490, 2018.

Boutilier, C. Planning, learning and coordination in multiagent decision processes. In *Proceedings of the 6th Conference on Theoretical Aspects of Rationality and Knowledge*, TARK '96, pp. 195–210. Morgan Kaufmann Publishers Inc., 1996.

Bromuri, S. A tensor factorization approach to generalization in multi-agent reinforcement learning. In *2012 IEEE/WIC/ACM International Conferences on Web Intelligence and Intelligent Agent Technology*, volume 2, pp. 274–281. IEEE, 2012.

Bulat, A., Kossaifi, J., Tzimiropoulos, G., and Pantic, M. Incremental multi-domain learning with network latent tensor factorization. 2020.

Buşoniu, L., Babuška, R., and De Schutter, B. Multi-agent reinforcement learning: An overview. In *Innovations in multi-agent systems and applications-1*, pp. 183–221. Springer, 2010.

Chen, Y., Zhou, M., Wen, Y., Yang, Y., Su, Y., Zhang, W., Zhang, D., Wang, J., and Liu, H. Factorized q-learning for large-scale multi-agent systems. *arXiv preprint arXiv:1809.03738*, 2018.

Cheng, Y., Wang, D., Zhou, P., and Zhang, T. A survey of model compression and acceleration for deep neural networks. *arXiv preprint arXiv:1710.09282*, 2017.

Cichocki, A., Phan, A.-H., Zhao, Q., Lee, N., Oseledets, I., Sugiyama, M., Mandic, D. P., et al. Tensor networks for dimensionality reduction and large-scale optimization: Part 2 applications and future perspectives. *Foundations and Trends® in Machine Learning*, 9(6):431–673, 2017.

Cohen, N., Sharir, O., and Shashua, A. On the expressive power of deep learning: A tensor analysis. In *Conference on Learning Theory*, pp. 698–728, 2016.

Foerster, J. N., Farquhar, G., Afouras, T., Nardelli, N., and Whiteson, S. Counterfactual multi-agent policy gradients. In *Thirty-Second AAAI Conference on Artificial Intelligence*, 2018.

Greensmith, E., Bartlett, P. L., and Baxter, J. Variance reduction techniques for gradient estimates in reinforcement learning. *Journal of Machine Learning Research*, 5(9), 2004.

Guestrin, C., Lagoudakis, M., and Parr, R. Coordinated reinforcement learning. In *ICML*, volume 2, pp. 227–234. Citeseer, 2002a.

Guestrin, C., Venkataraman, S., and Koller, D. Context-specific multiagent coordination and planning with factored mdps. In *AAAI/IAAI*, pp. 253–259, 2002b.

Gupta, T., Mahajan, A., Peng, B., Böhmer, W., and Whiteson, S. Uneven: Universal value exploration for multi-agent reinforcement learning. *arXiv preprint arXiv:2010.02974*, 2020.

Hillar, C. J. and Lim, L.-H. Most tensor problems are np-hard. *Journal of the ACM (JACM)*, 60(6):1–39, 2013.

Jain, P. and Oh, S. Provable tensor factorization with missing data. In *Advances in Neural Information Processing Systems*, pp. 1431–1439, 2014.

Janzamin, M., Ge, R., Kossaifi, J., and Anandkumar, A. Spectral learning on matrices and tensors. *arXiv preprint arXiv:2004.07984*, 2020.

Kaelbling, L. P., Littman, M. L., and Cassandra, A. R. Planning and acting in partially observable stochastic domains. *Artificial intelligence*, 101(1-2):99–134, 1998.

Kakade, S. M. *On the sample complexity of reinforcement learning*. PhD thesis, UCL (University College London), 2003.

Kearns, M. J., Vazirani, U. V., and Vazirani, U. *An introduction to computational learning theory*. MIT press, 1994.

Kingma, D. P. and Ba, J. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.

Kolbeinsson, A., Kossaifi, J., Panagakis, Y., Bulat, A., Anandkumar, A., Tzoulaki, I., and Matthews, P. M. Tensor dropout for robust learning. *IEEE Journal of Selected Topics in Signal Processing*, 15(3):630–640, 2021.

Kolda, T. G. and Bader, B. W. Tensor decompositions and applications. *SIAM review*, 51(3):455–500, 2009.

Konda, V. and Tsitsiklis, J. N. *Actor-Critic Algorithms*. PhD thesis, USA, 2002. AAI0804543.

Kossaifi, J., Bulat, A., Tzimiropoulos, G., and Pantic, M. T-net: Parametrizing fully convolutional nets with a single high-order tensor. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2019.

Kossaifi, J., Toisoul, A., Bulat, A., Panagakis, Y., Hospedales, T., and Pantic, M. Factorized higher-order cnns with an application to spatio-temporal emotion estimation. In *IEEE CVPR*, 2020.

Krishnamurthy, A., Agarwal, A., and Langford, J. Pac reinforcement learning with rich observations. In *Advances in Neural Information Processing Systems*, pp. 1840–1848, 2016.

Lattimore, T., Hutter, M., and Sunehag, P. The sample-complexity of general reinforcement learning. In *International Conference on Machine Learning*, pp. 28–36. PMLR, 2013.

Lowe, R., Wu, Y., Tamar, A., Harb, J., Abbeel, O. P., and Mordatch, I. Multi-agent actor-critic for mixed cooperative-competitive environments. In *Advances in Neural Information Processing Systems*, pp. 6379–6390, 2017.

Mahajan, A. and Tulabandhula, T. Symmetry detection and exploitation for function approximation in deep rl. In *Proceedings of the 16th Conference on Autonomous Agents and MultiAgent Systems*, pp. 1619–1621, 2017a.

Mahajan, A. and Tulabandhula, T. Symmetry learning for function approximation in reinforcement learning. *arXiv preprint arXiv:1706.02999*, 2017b.

Mahajan, A., Rashid, T., Samvelyan, M., and Whiteson, S. Maven: Multi-agent variational exploration. In *Advances in Neural Information Processing Systems*, pp. 7611–7622, 2019.

Matignon, L., Laurent, G. J., and Le Fort-Piat, N. Hysteretic q-learning: an algorithm for decentralized reinforcement learning in cooperative multi-agent teams. In *2007 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pp. 64–69. IEEE, 2007.

Messias, J. a. V., Spaan, M. T. J., and Lima, P. U. Efficient offline communication policies for factored multiagent pomdps. In *Proceedings of the 24th International Conference on Neural Information Processing Systems*, pp. 1917–1925. Curran Associates Inc., 2011.

Oliehoek, F. A. and Amato, C. *A Concise Introduction to Decentralized POMDPs*. SpringerBriefs in Intelligent Systems. Springer, 2016.

Panagakis, Y., Kossaifi, J., Chrysos, G. G., Oldfield, J., Nicolaou, M. A., Anandkumar, A., and Zafeiriou, S. Tensor methods in computer vision and deep learning. *Proceedings of the IEEE*, 109(5):863–890, 2021.

Rashid, T., Samvelyan, M., de Witt, C. S., Farquhar, G., Foerster, J., and Whiteson, S. QMIX: Monotonic Value Function Factorisation for Deep Multi-Agent Reinforcement Learning. In *Proceedings of the 35th International Conference on Machine Learning*, pp. 4295–4304, 2018.

Samvelyan, M., Rashid, T., de Witt, C. S., Farquhar, G., Nardelli, N., Rudner, T. G., Hung, C.-M., Torr, P. H., Foerster, J., and Whiteson, S. The StarCraft Multi-Agent Challenge. In *Proceedings of the 18th International Conference on Autonomous Agents and MultiAgent Systems*, 2019.

Sidiropoulos, N. D., De Lathauwer, L., Fu, X., Huang, K., Papalexakis, E. E., and Faloutsos, C. Tensor decomposition for signal processing and machine learning. *IEEE Transactions on Signal Processing*, 65(13):3551–3582, 2017.

Son, K., Kim, D., Kang, W. J., Hostallero, D. E., and Yi, Y. Qtran: Learning to factorize with transformation for cooperative multi-agent reinforcement learning. *arXiv preprint arXiv:1905.05408*, 2019.

Sunehag, P., Lever, G., Gruslys, A., Czarnecki, W. M., Zambaldi, V., Jaderberg, M., Lanctot, M., Sonnerat, N., Leibo, J. Z., Tuyls, K., and Graepel, T. Value-Decomposition Networks For Cooperative Multi-Agent Learning Based On Team Reward. In *Proceedings of the 17th International Conference on Autonomous Agents and Multiagent Systems*, 2017.

Sutton, R. S. Learning to predict by the methods of temporal differences. *Machine learning*, 3(1):9–44, 1988.

Sutton, R. S. and Barto, A. G. Reinforcement learning: An introduction. 2011.

Sutton, R. S., McAllester, D. A., Singh, S. P., and Mansour, Y. Policy gradient methods for reinforcement learning with function approximation. In *Advances in neural information processing systems*, pp. 1057–1063, 2000.

Tan, M. Multi-agent reinforcement learning: Independent vs. cooperative agents. In *Proceedings of the Tenth International Conference on Machine Learning*, pp. 330–337, 1993.

Wang, J., Ren, Z., Liu, T., Yu, Y., and Zhang, C. Qplex: Duplex dueling multi-agent q-learning. *arXiv preprint arXiv:2008.01062*, 2020a.

Wang, T., Gupta, T., Mahajan, A., Peng, B., Whiteson, S., and Zhang, C. Rode: Learning roles to decompose multi-agent tasks. *arXiv preprint arXiv:2010.01523*, 2020b.

Wei, E., Wicke, D., Freelan, D., and Luke, S. Multiagent soft q-learning. In *2018 AAAI Spring Symposium Series*, 2018.

Yang, Y., Hao, J., Liao, B. L., Shao, K., Chen, G., Liu, W., and Tang, H. Qatten: A general framework for cooperative multiagent reinforcement learning. *ArXiv*, abs/2002.03939, 2020.

Yao, X., Wen, C., Wang, Y., and Tan, X. Smix: Enhancing centralized value functions for cooperative multi-agent reinforcement learning. *arXiv preprint arXiv:1911.04094*, 2019.

Zhao, T., Niu, G., Xie, N., Yang, J., and Sugiyama, M. Regularized policy gradients: direct variance reduction in policy gradient estimation. In *Asian Conference on Machine Learning*, pp. 333–348. PMLR, 2016.