Energy-based Surprise Minimization for Multi-Agent Value Factorization

Karush Suri[1], **Xiao Qi Shi**[2], **Konstantinos Plataniotis**[1] **and Yuri Lawryshyn**[1]
[1]University of Toronto [2]RBC Capital Markets
karush.suri@mail.utoronto.ca

Abstract

Multi-Agent Reinforcement Learning (MARL) has demonstrated significant success in training decentralised policies in a centralised manner by making use of value factorization methods. However, addressing surprise across spurious states and approximation bias remain open problems for multi-agent settings. We introduce the Energy-based MIXer (EMIX), an algorithm which minimizes surprise utilizing the energy across agents. Our contributions are threefold; (1) EMIX introduces a novel surprise minimization technique across multiple agents in the case of multi-agent partially-observable settings. (2) EMIX highlights the first practical use of energy functions in MARL (to our knowledge) with theoretical guarantees and experiment validations of the energy operator. Lastly, (3) EMIX presents a novel technique for addressing overestimation bias across agents in MARL. When evaluated on a range of challenging StarCraft II micromanagement scenarios, EMIX demonstrates consistent state-of-the-art performance for multi-agent surprise minimization. Moreover, our ablation study highlights the necessity of the energy-based scheme and the need for elimination of overestimation bias in MARL. Our implementation of EMIX, videos of agents and blog are available in the supplementary material.

1 Introduction

Reinforcement Learning (RL) has seen tremendous growth in applications such as arcade games [1], board games [2, 3], robot control tasks [4, 5] and lately, real-time games [6]. The rise of RL has led to an increasing interest in the study of multi-agent systems [7, 8], commonly known as Multi-Agent Reinforcement Learning (MARL). In the case of partially observable settings, MARL enables the learning of policies with centralised training and decentralised control [9]. This has proven to be useful for exploiting value-based methods which are often found to be sample-inefficient [10, 11]. Value Factorization [12, 13] is a common technique which enables the joint value function to be represented as a combination of individual value functions. In the case of Value Decomposition Network (VDN) [12], a linear additive factorization is carried out whereas QMIX [13] generalizes the factorization to a non-linear combination, hence improving the expressive power of centralised action-value functions. Furthermore, monotonicity constraints in QMIX enable scalability in the number of agents. On the other hand, factorization across multiple value functions leads to the aggregation of approximation biases [14, 15] originating from overoptimistic estimations in action values [16, 17] which remain an open problem in the case of multi-agent settings. Moreover, value factorization methods are conditioned on states and do not account for spurious changes in partially-observed observations, commonly referred to as surprise [18].

Surprise minimization [19] is a recent phenomenon observed in the case of single-agent RL methods which deals with environments consisting of spurious states. In the case of model-based RL [20], surprise minimization is used as an effective planning tool in the agent's model [19] whereas in the case of model-free RL, surprise minimization is witnessed as an intrinsic motivation [18, 21] or generalization problem [22]. On the other hand, MARL does not account for

surprise across agents as a result of which agents remain unaware of drastic changes in the environment [23]. Thus, surprise minimization in the case of multi-agent settings requires attention from a critical standpoint.

We introduce the Energy-based MIXer (EMIX), an algorithm based on QMIX which minimizes surprise utilizing the energy across agents. Our contributions are threefold; (1) EMIX introduces a novel surprise minimization technique across multiple agents in the case of multi-agent partially-observable settings. (2) EMIX highlights the first practical use of energy functions in MARL (to our knowledge) with theoretical guarantees and experiment validations of the energy operator. Lastly, (3) EMIX presents a novel technique for addressing overestimation bias across agents in MARL which, unlike previous single-agent methods [17], do not rely on a computationally-expensive family of action value functions. When evaluated on a range of challenging StarCraft II scenarios [24], EMIX demonstrates state-of-the-art performance for multi-agent surprise minimization by significantly improving the consistent performance of QMIX. Moroever, our ablation study highlights the necessity of our energy-based scheme and the need for elimination of overestimation bias in MARL.

2 The Value Factorization Problem

2.1 Preliminaries

We review the cooperative MARL setup. The problem is modeled as a Partially Observable Markov Decision Process (POMDP) [25] defined by the tuple $(\mathcal{S}, \mathcal{A}, r, N, P, Z, O, \gamma)$ where the state space $\mathcal{S}$ and action space $\mathcal{A}$ are discrete, $r : \mathcal{S} \times \mathcal{A} \rightarrow [r_{min}, r_{max}]$ presents the reward observed by agents $a \in N$ where N is the set of all agents, $P : \mathcal{S} \times \mathcal{S} \times \mathcal{A} \rightarrow [0, \infty)$ presents the unknown transition model consisting of the transition probability to the next state $s' \in \mathcal{S}$ given the current state $s \in \mathcal{S}$ and joint action $u \in \mathcal{A}$ at time step t and γ is the discount factor. We consider a partially observable setting in which each agent n draws individual observations $z \in Z$ according to the observation function $O(s, u) : \mathcal{S} \times \mathcal{A} \rightarrow Z$. We consider a joint policy $\pi_\theta(u|s)$ as a function of model parameters θ. Standard RL defines the agent's objective to maximize the expected discounted reward $\mathbb{E}_{\pi_\theta}[\sum_{t=0}^{T} \gamma^t r(s_t, u_t)]$ as a function of the parameters θ. The action-value function for an agent is represented as $Q(u, s; \theta) = \mathbb{E}_{\pi_\theta}[\sum_{t=1}^{T} \gamma^t r(s, u) | s = s_t, u = u_t]$ which is the expected sum of payoffs obtained in state s upon performing action u by following the policy π_θ. We denote the optimal policy π^*_θ such that $Q(u, s; \theta^*) \geq Q(u, s; \theta) \forall s \in S, u \in A$. In the case of multiple agents, the joint optimal policy can be expressed as the Nash Equilibrium [26] of the Stochastic Markov Game as $\pi^* = (\pi^{1,*}, \pi^{2,*}, ...\pi^{N,*})$ such that $Q(u^a, s; \theta^*) \geq Q(u^a, s; \theta) \forall s \in S, u \in A, a \in N$. Q-Learning is an off-policy, model-free algorithm suitable for continuous and episodic tasks. The algorithm uses semi-gradient descent to minimize the Temporal Difference (TD) error: $\mathbb{L}(\theta) = \mathop{\mathbb{E}}_{b \sim R}[(y - Q(u, s; \theta))^2]$ where $y = r + \gamma \max_{u' \in A} Q(u', s'; \theta^-)$ is the TD target consisting of θ^- as the target parameters and b is the batch sampled from memory R.

2.2 Surprise Minimization

Despite the recent success of value-based methods [27, 28] RL agents suffer from spurious state spaces and encounter sudden changes in trajectories. These anomalous transitions between consecutive states are termed as surprise [18]. Quantitatively, surprise can be inferred as a measure of deviation [19, 22] among states encountered by the agent during its interaction with the environment. While exploring [29, 30] the environment, agents tend to have higher deviation among states which is gradually reduced by gaining a significant understanding of state-action transitions. Agents can then start selecting optimal actions which is essential for maximizing reward. These actions often lead the agent to spurious experiences which the agent may not have encountered. In the case of model-based RL, agents can leverage spurious experiences [19] and plan effectively for future steps. On the other hand, in the case of model-free RL, surprise results in sample-inefficient learning [18]. This can be tackled by making use of rigorous exploration strategies [31, 32]. However, such techniques do not necessarily scale to high-dimensional tasks and often require extrinsic feature engineering [33] and meta models [34]. A suitable way to tackle high-dimensional dynamics is by utilizing surprise as a penalty on the reward [22]. This leads to improved generalization. However, such solutions do not show evidence for multiple agents consisting of individual partial observations [35].

2.3 Overestimation Bias

Recent advances [16] in value-based methods have addressed overestimation bias (also known as approximation error) which stems from the value estimates approximated by the function approximator. Such methods make use of dual target functions [36] which improve stability in the Bellman updates. This has led to a significant

improvement in single-agent off-policy RL methods [37]. However, MARL value-based methods continue to suffer from overestimation bias [38, 39]. Figure 1 highlights the overestimation bias originating from the overoptimistic estimations of the target value estimator. Plots present the variation of absolute TD error during learning for state-of-the-art MARL methods, namely Independent Q-Learning [10], Counterfactual Multi-Agent Policy Gradients (COMA) [11], VDN [12] and QMIX [13]. Significant rise in error values of value factorization methods such as QMIX and VDN presents the aggregation of errors from individual Q-value functions. Thus, overestimation bias in MARL value factorization requires attention from a critical standpoint.

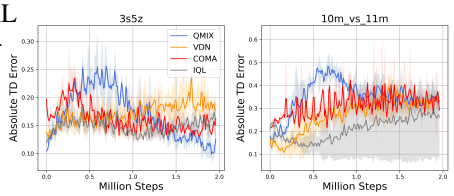

Figure 1: Absolute TD error for state-of-the-art MARL methods in StarCraft II micromanagement scenarios. Rise in error values depict the overoptimistic approximations estimated by the target value estimator.

Various MARL methods [40] make use of a dual architecture approach which increases the stability in value factorization. However, these methods are only applicable to small set of micromanagement tasks and do not generalize to scenarios consisting of a larger number of opponents and environments with different dynamics. Another suitable approach observed in literature is the usage of weighted bellman updates in double Q-learning [41]. The Weighted Double Deep Q-Network (WDDQN) provides stability and sample efficiency for fully-observable MDPs. In the case of cooperative POMDPS, Weighted-QMIX (WQMIX) [42] yields a more sophisticated weighting scheme which aids in the retrieval of optimal policy [43]. Although suitable for value factorization in challenging micromanagement tasks, the method needs to be carefully hand-engineered and, in the case of multiple weighting schemes, does not include a basis for selection. A more practical approach in the case of single-agent methods is the use of a family of Q-functions [17] wherein each estimator is optimized individually. Such a framework provides a generalized method for training agents with greedy policies and minimum approximation error. Although successful in single-agent settings, generalized Q-function methods do not scale well in the number of agents [43] since each agent requires a family of Q-functions which needs to be updated concurrently. Thus, addressing overestimation bias from value factorization in cooperative multi-agent frameworks requires a scalable and sample-efficient perspective.

2.4 Energy-based Models

Energy-Based Models (EBMs) [44, 45] have been successfully applied in the field of machine learning [46] and probabilistic inference [47]. A typical EBM $\mathcal{E}$ formulates the equilibrium probabilities [48] $P(v,h) = \frac{\exp\left(-\mathcal{E}(v,h)\right)}{\sum_{\hat{v},\hat{h}}[\exp\left(-\mathcal{E}(\hat{v},\hat{h})\right)]}$ via a Boltzmann distribution [49] where v and h are the values of the visible and hidden variables and $\hat{v}$ and $\hat{h}$ are all the possible configurations of the visible and hidden variables respectively. The probability distribution over all the visible variables can be obtained by summing over all possible configurations of the hidden variables. This is mathematically expressed in Equation 1.

$$P(v) = \frac{\sum_h[\exp\left(-\mathcal{E}(v,h)\right)]}{\sum_{\hat{v},\hat{h}}[\exp\left(-\mathcal{E}(\hat{v},\hat{h})\right)]} \tag{1}$$

Here, $\mathcal{E}(v,h)$ is called the equilibrium free energy which is the minimum of the variational free energy and $\sum_{\hat{v},\hat{h}}[\exp\left(-\mathcal{E}(\hat{v},\hat{h})\right)]$ is the partition function. EBMs have been successfully implemented in single-agent RL methods [50, 51]. These typically make use of Boltzmann distributions to approximate policies [49]. Such a formulation results in the minimization of free energy within the agent. While policy approximation depicts promise in the case of unknown dynamics, inference methods [52] play a key role in optimizing goal-oriented behavior. A second type of usage of EBMs follows the maximization of entropy [53]. The maximum entropy framework [37] highlighted in Soft Q-Learning (SQL) [51] allows the agent to obey a policy which maximizes its reward and entropy concurrently. Maximization of agent's entropy results in diverse and adaptive behaviors [54] which may be difficult to accomplish using standard exploration techniques [29, 30]. Moreover, the maximum entropy framework is equivalent to approximate inference in the case of policy gradient methods [55]. Such a connection between likelihood ratio gradient techniques and energy-based formulations leads to diverse and robust policies [56] and their hierarchical extensions [57] which preserve the lower levels of hierarchies.

In the case of MARL, EBMs have witnessed limited applicability as a result of the increasing number of agents and complexity within each agent [58]. While the probabilistic framework is readily transferable to opponent-aware multi-agent

systems [59], cooperative settings consisting of coordination between agents require a firm formulation of energy which is scalable in the number of agents [60] and accounts for environments consisting of spurious states [61].

3 Energy-based Surprise Minimization

In this section we introduce the novel surprise minimizing EMIX agent. The motivation behind EMIX stems from spurious states and overestimation bias among agents in the case of partially-observed settings. EMIX aims to address these challenges by making use of an energy-based surprise value function in conjunction with dual target function approximators.

3.1 The Surprise Minimization Objective

Firstly, we formulate the energy-based objective consisting of surprise as a function of states s, joint actions u and deviation σ within states for each agent a. We call this function as the surprise value function $V^a_{surp}(s,u,\sigma)$ which serves as a mapping from agent and environment dynamics to surprise. We then define an energy operator presented in Equation 2 which sums the free energy across all agents.

$$\mathcal{T}V^a_{surp}(s,u,\sigma) = \log \sum_{a=1}^{N} \exp\left(V^a_{surp}(s,u,\sigma)\right) \tag{2}$$

We make use of the Mellowmax operator [62] as our energy operator. The energy operator is similar to the SQL energy formulation [51] where the energy across different actions is evaluated. In our case, inference is carried out across all agents with actions as prior variables. However, in the special case of using an EBM as a Q-function, the EMIX objective reduces to the SQL objective (details in the supplementary material).

Our choice of the energy operator is based on its unique mathematical properties which result in better convergence. Of these properties, the most useful result is that the energy operator forms a contraction on the surprise value function indicating a guaranteed minimization of surprise within agents. This is formally stated in Theorem 1. Proof of Theorem 1 can be found in the supplementary material.

Theorem 1. *Given a surprise value function* $V^a_{surp}(s,u,\sigma) \forall a \in N$, *the energy operator* $\mathcal{T}V^a_{surp}(s,u,\sigma) = \log \sum_{a=1}^{N} \exp\left(V^a_{surp}(s,u,\sigma)\right)$ *forms a contraction on* $V^a_{surp}(s,u,\sigma)$.

The energy-based surprise minimization objective can then be formulated by simply adding the approximated energy-based surprise to the initial Bellman objective as expressed below.

$$L(\theta) = \mathop{\mathbb{E}}_{b\sim R}[\frac{1}{2}(y - (Q(u,s;\theta) + \beta \log \sum_{a=1}^{N} \exp\left(V^a_{surp}(s,u,\sigma)\right)))^2]$$

where $y = r + \gamma \max_{u'} Q(u', s'; \theta^-) + \beta \log \sum_{a=1}^{N} \exp\left(V^a_{surp}(s', u', \sigma')\right)$. This yields the following,

$$= \mathop{\mathbb{E}}_{b\sim R}[\frac{1}{2}(r + \gamma \max_{u'} Q(u', s'; \theta^-) + \beta \log \frac{\sum_{a=1}^{N} \exp\left(V^a_{surp}(s', u', \sigma')\right)}{\sum_{a=1}^{N} \exp\left(V^a_{surp}(s,u,\sigma)\right)} -$$

$$L(\theta) = \mathop{\mathbb{E}}_{b\sim R}[\frac{1}{2}(r + \gamma \max_{u'} Q(u', s'; \theta^-) + \beta E - Q(u,s;\theta))^2] \tag{3}$$

Here, E is defined as the surprise ratio with β as a temperature parameter and σ' as the deviation among next states in the batch. The surprise value function is approximated by a universal function approximator (in our case a neural network) with its parameters as ϕ. $V_a(s', u', \sigma')$ is expressed as the negative free energy and $\sum_{a=1}^{N} \exp\left(V_a(s,u,\sigma)\right)$ the partition function. Alternatively, $V_a(s,u,\sigma)$ can be formulated as the negative free energy with $\sum_{a=1}^{N} \exp\left(V_a(s', u', \sigma')\right)$ as the partition function. The objective incorporates the minimization of surprise across all agents as minimizing the energy in spurious states. Such a formulation of surprise acts as intrinsic motivation and at the same time provides robustness to multi-agent behavior. Furthermore, the energy formulation in the form of energy ratio E is a suitable one as it guarantees convergence to minimum surprise at optimal policy π^*. This is formally expressed in Theorem 2 with its corresponding proof in the supplementary material.

Theorem 2. *Upon agent's convergence to an optimal policy* π^*, *total energy of* π^*, *expressed by* E^* *will reach a thermal equilibrium consisting of minimum surprise among consecutive states* s *and* s'.

The objective can be modified to tackle approximation error in the target Q-values. We introduce a total of m target approximators making $\{Q_1(u', s'; \theta^-), Q_2(u', s'; \theta^-) ..., Q_m(u', s'; \theta^-)\}$ as the set of target approximators. However, unlike generalized Q-learning [17], we do not instantiate another Q-function but simply keep a copy of θ and select the target estimates with minimum values during optimization. This allows the objective to address overestimation

bias in a scalable manner without using multiple Q-functions. The final EMIX objective is mathematically expressed in Equation 4.

$$L(\theta) = \mathop{\mathbb{E}}_{b\sim R}[\frac{1}{2}(r + \gamma \max_{u^{'}} \min_{i} Q_i(u^{'}, s^{'}; \theta^{-}) + \beta E - Q(u, s; \theta))^2] \tag{4}$$

Here, i depicts each of the m target estimators with $\min_i Q_i(u^{'}, s^{'}; \theta^{-})$ indicating the estimate with minimum error.

3.2 Energy-based MIXer (EMIX)

Algorithm 1 Energy-based MIXer (EMIX)

Initialize ϕ, θ, $\theta_1^{-}...,\theta_m^{-}$, agent and hypernetwork parameters.
Initialize learning rate α, temperature β and replay buffer $\mathcal{R}$.
for environment step **do**
  $u \leftarrow (u_1, u_2..., u_N)$
  $\mathcal{R} \leftarrow \mathcal{R} \cup \{(s, u, r, s^{'})\}$
  if $|\mathcal{R}| >$ batch-size **then**
    for random batch **do**
      $Q_{tot}^{\theta} \leftarrow$ *Mixer-Network*$(Q_1, Q_2..., Q_N, s)$
      $Q_i^{\theta^{-}} \leftarrow$ *Target-Mixer*$_i(Q_1, Q_2..., Q_N, s^{'}), \forall i = 1, 2.., m$
      Calculate σ and $\sigma^{'}$ using s and $s^{'}$
      $V_{surp}^{a}(s, u, \sigma) \leftarrow$ *Surprise-Mixer*(s, u, σ)
      $V_{surp}^{a}(s^{'}, u^{'}, \sigma^{'}) \leftarrow$ *Target-Surprise-Mixer*$(s^{'}, u^{'}, \sigma^{'})$
      $E \leftarrow \log \frac{\sum_{a=1}^{N} \exp(V_{surp}^{a}(s^{'}, u^{'}, \sigma^{'}))}{\sum_{a=1}^{N} \exp(V_{surp}^{a}(s, u, \sigma))}$
      Calculate $L(\theta)$ using E in Equation 4
      $\theta \leftarrow \theta - \alpha \nabla_{\theta} L(\theta)$
    end for
  end if
  if update-interval steps have passed **then**
    $\theta_i^{-} \leftarrow \theta, \forall i = 1, 2.., m$
  end if
end for

Algorithm 1 presents the EMIX algorithm. We initialize surprise value function parameters ϕ, mixer parameters θ, target parameters θ_i^{-} for $i = 1, 2..., m$ and lastly the agent and hypernetwork parameters of QMIX. A learning rate α, temperature β and replay buffer $\mathcal{R}$ are instantiated. During environment interactions, agents in state s perform joint action u, observe reward r and transition to next-states $s^{'}$. These experiences are collected in $\mathcal{R}$ as $(s, u, r, s^{'})$ tuples. In order to make the agents explore the environment, an ϵ-greedy schedule is used similar to the original QMIX [13] implementation. During the update steps, a random batch of $batch - size$ is sampled from $\mathcal{R}$. The total Q-value Q_{tot}^{θ} is computed by the mixer network with its inputs as the Q-values of all the agents conditioned on s via the hypernetworks. Similarly, the target mixers approximate $Q_i^{\theta^{-}}$ conditioned on $s^{'}$. In order to evaluate surprise within agents, we compute the standard deviations σ and $\sigma^{'}$ across all observations z and $z^{'}$ for each agent using s and $s^{'}$ respectively. The surprise value function called the Surprise-Mixer estimates the surprise $V_{surp}^{a}(s, u, \sigma)$ conditioned on s, u and σ. The same computation is repeated using the Target-Surprise-Mixer for estimating surprise $V_{surp}^{a}(s^{'}, u^{'}, \sigma^{'})$ within next-states in the batch. Application of the energy operator along the non-singleton agent dimension for $V_{surp}^{a}(s, u, \sigma)$ and $V_{surp}^{a}(s^{'}, u^{'}, \sigma^{'})$ yields the energy ratio E which is used in Equation 4 to evaluate $L(\theta)$. We then use batch gradient descent to update parameters of the mixer θ. Target parameters θ_i^{-} are updated every $update - interval$ steps. We now take a closer look at the surprise-mixer approximating the surprise value function. In order to condition surprise on states, joint actions and the deviation among states, we construct an expressive architecture motivated by provable exploration in RL [63]. The original architecture constructs a state abstraction model for a classification setting. It maps the transitions consisting of states s, actions u and next-states $s^{'}$ to the conditional probability $p(y|s, a, s^{'})$ depicting whether the transition belongs to the same data distribution y or not. Such models have proven to be efficient in the case of provable exploration [63] as it allows the agent to learn an exploration policy for every value of abstract state related to the latent space. We borrow from this technique of provable exploration and extend it to the surprise minimization setting.

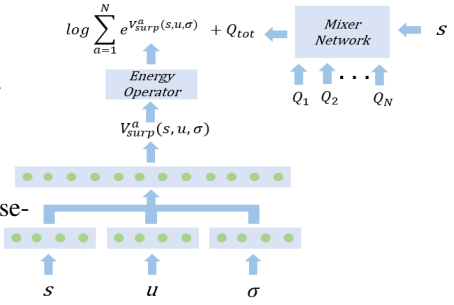

Figure 2: Surprise-Mixer architecture for estimation of the surprise value function.

Figure 2 presents the expressive architecture of surprise-mixer network utilized for surprise value function approximation and minimization. In contrast to the original state abstraction model [63], the surprise-mixer maps transitions consisting of states s, joint actions u and deviations σ to a surprise value $V^a_{surp}(s, u, \sigma)$ for all agents a. Hierarchical layers of the network aid in the extraction of latent space representations followed by the estimation of $V^a_{surp}(s, u, \sigma)$. The architecture allows the agent to learn a robust and surprise-agnostic policy for every value of abstract state related to the latent space. Moreover, the latent space accommodates every value of surprise across agents as a result of state deviations induced in the intermediate representations. Surprise value estimates $V^a_{surp}(s, u, \sigma)$ are evaluated by the energy operator with the resulting expression becoming a part of the Bellman objective in Equation 4 comprising of the total Q-values Q_{tot} estimated by the mixer network.

4 Experiments

Our experiments aim to evaluate the performance, consistency, sample-efficiency and effectiveness of the various components of our method. Specifically, we aim to answer the following questions- (1) How does our method compare to current state-of-the-art MARL methods in terms of performance, consistency and sample efficiency?, (2) How much does each component of the method contribute to its performance? and (3) Does the algorithm validate the theoretical claims corresponding to its components?

4.1 Energy-based Surprise Minimization

We assess the performance and sample-efficiency of EMIX on multi-agent StarCraft II micromanagement scenarios [24] as these consist of a larger number of agents with different action spaces which motivates a greater deal of coordination. Additionally, micromanagement scenarios in StarCraft II consist of multiple opponents which introduce a greater degree of surprise within consecutive states. We compare our method to current state-of-the-art methods, namely QMIX [13], VDN [12], COMA [11] and IQL [10]. In order to compare our surprise-based scheme against pre-existing surprise minimization mechanisms, we compare EMIX additionally to a model-free implementation of SMiRL [19] in QMIX. All methods were implemented using the PyMARL framework [24]. The SMiRL component was additionally incorporated as per the update rule provided in [22]. We use the generalized version of SMiRL as it demonstrates reduced variance across batches. We term this implementation as SMiRL-QMIX for our comparisons. Details related to the implementation of EMIX are presented in the supplementary material.

Table 1 presents the comparison of success rate percentages between EMIX and state-of-the-art MARL algorithms on the StarCraft II micromanagement scenarios. Along with the success rates, we also measure the deviation of performance across the 5 random seeds considered during experiments. (complete results in supplementary material). We evaluate the performance of agents on a total of 12 scenarios. Corresponding to each scenario, algorithms demonstrating higher success rate values in comparison to other methods have their entries highlighted. Out of the 12 scenarios considered, EMIX presents higher success rates on 9 of these scenarios depicting the suitability of the proposed approach. EMIX presents significant performance gains in cases of *so_many_baneling* and *5m_vs _6m* which consist of a large number of opponents and a greater difficulty level respectively. When compared to QMIX, EMIX depicts improved success rates on all of the 12 scenarios. For instance, in scenarios such as *3s_vs_5z*, *8m_vs_9m* and *5m_vs_6m* QMIX presents sub-optimal performance whereas EMIX utilizes a comparatively improved joint policy and yields better convergence in a sample-efficient manner. Moreover, on comparing EMIX with SMiRL-QMIX, we note that EMIX demonstrates a higher average success rate. This highlights the suitability of the energy-based scheme in the case of a larger number of agents and complex environment dynamics for surprise minimization.

4.2 Ablation Study

We now present the ablation study for the various components of EMIX. Our experiments aim to determine the effectiveness of the energy-based surprise minimization method and the multiple target Q-function scheme. Additionally, we also aim to determine the extent up to which our proposed framework is viable in the standard QMIX objective.

EMIX Objective: To weigh the effectiveness of the multiple target Q-function scheme we remove the energy-based surprise minimization from EMIX and replace it with the prior QMIX objective. For simplicity, we make use of two target Q-functions. We call this implementation of QMIX combined with the dual target function scheme as *TwinQMIX*. We can now add the energy-based surprise minimization scheme in the TwinQMIX objective to retrieve the EMIX objective. Thus, we can compare between QMIX, TwinQMIX and EMIX to assess the contribu-

Scenarios	EMIX	SMiRL-QMIX	QMIX	VDN	COMA	IQL
2s_vs_1sc	90.33 ± 0.72	88.41 ± 1.31	89.19 ± 3.23	91.42 ± 1.23	**96.90 ± 0.54**	86.07 ± 0.98
2s3z	**95.40±0.45**	94.93±0.32	95.30±1.28	92.03±2.08	43.33±2.70	55.74±6.84
3m	**94.90±0.39**	93.94±0.22	93.43±0.20	94.58±0.58	84.75±7.93	94.79±0.50
3s_vs_3z	**99.58±0.07**	97.63±1.08	99.43±0.20	97.90±0.58	0.21±0.54	92.32±2.83
3s_vs_4z	**97.22±0.73**	0.24±0.11	96.01±3.93	94.29±2.13	0.00±0.00	59.75±12.22
3s_vs_5z	52.91±11.80	0.00±0.00	43.44±7.09	**68.51±5.60**	0.00±0.00	18.14±2.34
3s5z	**88.88±1.07**	88.53±1.03	88.49±2.32	63.58±3.99	0.25±0.11	7.05±3.52
8m	**94.47±1.38**	89.96±1.42	94.30±2.90	90.26±1.12	92.82±0.53	83.53±1.62
8m_vs_9m	**71.03±2.69**	69.90±1.94	68.28±2.30	58.81±4.68	4.17±0.58	28.48±22.38
10m_vs_11m	75.35±2.30	**77.85±2.02**	70.36±2.87	71.81±6.50	4.55±0.73	32.27±25.68
so_many_baneling	**95.87±0.16**	93.61±0.94	93.35±0.78	92.26±1.06	91.65±2.26	74.97±6.52
5m_vs_6m	**37.07±2.42**	33.27±2.79	34.42±2.63	35.63±3.32	0.52±0.13	14.78±2.72

Table 1: Comparison of success rate percentages between EMIX and state-of-the-art MARL methods for StarCraft II micromanagement scenarios. Results are averaged over 5 random seeds with each session consisting of 2 million environment interactions. EMIX significantly improves the performance of the QMIX agent on a total of 9 out of 12 scenarios. EMIX demonstrates state-of-the-art performance for surprise minimization on all 12 scenarios in comparison to the SMiRL scheme. In addition, EMIX presents less deviation between its random seeds indicating consistency in collaboration across agents.

tions of each of the proposed methods. **??** (top) presents the comparison of average success rates for QMIX, TwinQMIX and EMIX on six different scenarios. Agents were evaluated for a total of 2 million timesteps with the lines in the plot indicating average success rates and the shaded area as the deviation across 5 random seeds. In comparison to QMIX, TwinQMIX adds stability to the original objective by reducing the overoptimistic estimates in the initial QMIX objective. On comparing TwinQMIX to EMIX we note that the energy-based surprise minimization scheme provides significant performance improvement in the modified QMIX objective. This is demonstrated in the *5m_vs_6m* scenario wherein the EMIX implementation improves the performance of TwinQMIX in comparison to QMIX by utilizing a surprise-robust policy. In the case of *so_many_baneling* scenario which consists of a large number of opponents (27 banelings), EMIX tackles surprise effectively by preventing a significant drop in performance which is observed in cases of QMIX and TwinQMIX.

Temperature Parameter: The importance of β can be validated by assessing its usage in surprise minimization. However, it is difficult to evaluate surprise minimization directly as surprise value function estimates $V^a_{surp}(s, u, \sigma)$ vary from state-to-state across different agents and thus, they present high variance during agent's learning. This, in turn poses hindrance to gain an intuitive understanding of the surprise distribution. We instead observe the variation of E as it is a collection of surprise-based sample estimates across the batch. Additionally, E consists of prior samples $V^a_{surp}(s, u, \sigma)$ for $V^a_{surp}(s^{'}, u^{'}, \sigma^{'})$ which makes inference across different agents tractable. **??** (bottom) presents the variation of Energy ratio E with the temperature parameter β during learning. We compare two stable variations of E at $\beta = 0.001$ and $\beta = 0.01$. The objective minimizes E over the course of learning and attains thermal equilibrium with minimum energy. Intuitively, equilibrium corresponds to convergence to optimal policy π^* which validates the claim in Theorem 2. With $\beta = 0.01$, EMIX presents improved convergence and surprise minimization for 5 out of the 6 considered scenarios, hence validating the suitable choice of β. On the other hand, a lower value of $\beta = 0.001$ does little to minimize surprise across agents. In the case of high β values, EMIX demonstrates unstable behavior as a result of increasing overestimation error. Thus, a suitable value of β is critical for optimal performance and surprise-robust behavior.

5 Conclusion

In this paper, we introduced the Energy-based MIXer (EMIX), a multi-agent value factorization algorithm based on QMIX which minimizes surprise utilizing the energy across agents. The EMIX objective satisfies theoretical guarantees of total energy and surprise minimization with experimental results validating these claims. Additionally, EMIX presents a novel technique for addressing overestimation bias across agents in MARL based on multiple target value approximators. EMIX demonstrates state-of-the-art performance and sample-efficiency on 9 out of total 12 StarCraft II micromanagement scenarios. Our ablations carried out on the proposed energy-based scheme, multiple target approximators and temperature parameter highlight the suitability and significance of each of the proposed contributions. While EMIX serves as the first practical example

(to our knowledge) of energy-based models in cooperative MARL, we aim to extend the energy framework to opponent-aware and hierarchical MARL. We leave this as our future work.

References

[1] Volodymyr Mnih, Koray Kavukcuoglu, David Silver, Alex Graves, Ioannis Antonoglou, Daan Wierstra, and Martin A. Riedmiller. Playing atari with deep reinforcement learning. *CoRR*, abs/1312.5602, 2013.

[2] David Silver, Aja Huang, Chris J. Maddison, Arthur Guez, Laurent Sifre, George van den Driessche, Julian Schrittwieser, Ioannis Antonoglou, Veda Panneershelvam, Marc Lanctot, Sander Dieleman, Dominik Grewe, John Nham, Nal Kalchbrenner, Ilya Sutskever, Timothy Lillicrap, Madeleine Leach, Koray Kavukcuoglu, Thore Graepel, and Demis Hassabis. Mastering the game of Go with deep neural networks and tree search. *Nature*, 529(7587):484–489, January 2016.

[3] Julian Schrittwieser, Ioannis Antonoglou, Thomas Hubert, Karen Simonyan, Laurent Sifre, Simon Schmitt, Arthur Guez, Edward Lockhart, Demis Hassabis, Thore Graepel, Timothy Lillicrap, and David Silver. Mastering atari, go, chess and shogi by planning with a learned model, 2019.

[4] Timothy P. Lillicrap, Jonathan J. Hunt, Alexander Pritzel, Nicolas Manfred Otto Heess, Tom Erez, Yuval Tassa, David Silver, and Daan Wierstra. Continuous control with deep reinforcement learning. *CoRR*, abs/1509.02971, 2015.

[5] John Schulman, Filip Wolski, Prafulla Dhariwal, Alec Radford, and Oleg Klimov. Proximal policy optimization algorithms. *CoRR*, abs/1707.06347, 2017.

[6] Oriol Vinyals, Timo Ewalds, Sergey Bartunov, Petko Georgiev, Alexander Sasha Vezhnevets, Michelle Yeo, Alireza Makhzani, Heinrich Küttler, John Agapiou, Julian Schrittwieser, John Quan, Stephen Gaffney, Stig Petersen, Karen Simonyan, Tom Schaul, Hado van Hasselt, David Silver, Timothy Lillicrap, Kevin Calderone, Paul Keet, Anthony Brunasso, David Lawrence, Anders Ekermo, Jacob Repp, and Rodney Tsing. Starcraft ii: A new challenge for reinforcement learning, 2017.

[7] Ryan Lowe, Yi Wu, Aviv Tamar, Jean Harb, Pieter Abbeel, and Igor Mordatch. Multi-agent actor-critic for mixed cooperative-competitive environments, 2017.

[8] Oriol Vinyals, Igor Babuschkin, Wojciech Czarnecki, Michaël Mathieu, Andrew Dudzik, Junyoung Chung, David Choi, Richard Powell, Timo Ewalds, Petko Georgiev, Junhyuk Oh, Dan Horgan, Manuel Kroiss, Ivo Danihelka, Aja Huang, Laurent Sifre, Trevor Cai, John Agapiou, Max Jaderberg, and David Silver. Grandmaster level in starcraft ii using multi-agent reinforcement learning. *Nature*, 575, 11 2019.

[9] Landon Kraemer and Bikramjit Banerjee. Multi-agent reinforcement learning as a rehearsal for decentralized planning. *Neurocomputing*, 190, 02 2016.

[10] Ming Tan. Multi-agent reinforcement learning: Independent vs. cooperative agents. In *In Proceedings of the Tenth International Conference on Machine Learning*, 1993.

[11] Jakob Foerster, Gregory Farquhar, Triantafyllos Afouras, Nantas Nardelli, and Shimon Whiteson. Counterfactual multi-agent policy gradients, 2017.

[12] Peter Sunehag, Guy Lever, Audrunas Gruslys, Wojciech Marian Czarnecki, Vinicius Zambaldi, Max Jaderberg, Marc Lanctot, Nicolas Sonnerat, Joel Z. Leibo, Karl Tuyls, and Thore Graepel. 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*, AAMAS '18, page 2085–2087, 2018.

[13] Tabish Rashid, Mikayel Samvelyan, Christian Schroeder de Witt, Gregory Farquhar, Jakob Foerster, and Shimon Whiteson. Qmix: Monotonic value function factorisation for deep multi-agent reinforcement learning. In *ICML 2018: Proceedings of the Thirty-Fifth International Conference on Machine Learning*, 2018.

[14] Hado V. Hasselt. Double q-learning. In *Advances in Neural Information Processing Systems 23*. 2010.

[15] Hado van Hasselt, Arthur Guez, and David Silver. Deep reinforcement learning with double q-learning. In *Proceedings of the Thirtieth AAAI Conference on Artificial Intelligence*, 2016.

[16] Scott Fujimoto, Herke van Hoof, and David Meger. Addressing function approximation error in actor-critic methods, 2018.

[17] Qingfeng Lan, Yangchen Pan, Alona Fyshe, and Martha White. Maxmin q-learning: Controlling the estimation bias of q-learning. In *International Conference on Learning Representations*, 2020.

[18] Joshua Achiam and Shankar Sastry. Surprise-based intrinsic motivation for deep reinforcement learning, 2017.

[19] Glen Berseth, Daniel Geng, Coline Devin, Dinesh Jayaraman, Chelsea Finn, and Sergey Levine. Smirl: Surprise minimizing rl in entropic environments. 2019.

[20] Lukasz Kaiser, Mohammad Babaeizadeh, Piotr Milos, Blazej Osinski, Roy H Campbell, Konrad Czechowski, Dumitru Erhan, Chelsea Finn, Piotr Kozakowski, Sergey Levine, Afroz Mohiuddin, Ryan Sepassi, George Tucker, and Henryk Michalewski. Model-based reinforcement learning for atari, 2019.

[21] Luis Macedo, Rainer Reisezein, and Amilcar Cardoso. Modeling forms of surprise in artificial agents: empirical and theoretical study of surprise functions. In *Proceedings of the Annual Meeting of the Cognitive Science Society*, volume 26, 2004.

[22] Jerry Zikun Chen. Reinforcement learning generalization with surprise minimization, 2020.

[23] Luis Macedo and Amilcar Cardoso. The role of surprise, curiosity and hunger on exploration of unknown environments populated with entities. In *2005 portuguese conference on artificial intelligence*, 2005.

[24] Mikayel Samvelyan, Tabish Rashid, Christian Schroeder de Witt, Gregory Farquhar, Nantas Nardelli, Tim G. J. Rudner, Chia-Man Hung, Philip H. S. Torr, Jakob Foerster, and Shimon Whiteson. The starcraft multi-agent challenge, 2019.

[25] Richard S. Sutton and Andrew G. Barto. *Reinforcement Learning: An Introduction*. 2018.

[26] John F. Nash. Equilibrium points in n-person games. *Proceedings of the National Academy of Sciences*, 36(1), 1950.

[27] Volodymyr Mnih, Adria Puigdomenech Badia, Mehdi Mirza, Alex Graves, Timothy Lillicrap, Tim Harley, David Silver, and Koray Kavukcuoglu. Asynchronous methods for deep reinforcement learning. In *International conference on machine learning*, 2016.

[28] Matteo Hessel, Joseph Modayil, Hado Van Hasselt, Tom Schaul, Georg Ostrovski, Will Dabney, Dan Horgan, Bilal Piot, Mohammad Azar, and David Silver. Rainbow: Combining improvements in deep reinforcement learning. *arXiv preprint arXiv:1710.02298*, 2017.

[29] Yuri Burda, Harri Edwards, Deepak Pathak, Amos Storkey, Trevor Darrell, and Alexei A. Efros. Large-scale study of curiosity-driven learning. In *ICLR*, 2019.

[30] Sebastian B Thrun. Efficient exploration in reinforcement learning. 1992.

[31] Bradly C Stadie, Sergey Levine, and Pieter Abbeel. Incentivizing exploration in reinforcement learning with deep predictive models. *arXiv preprint arXiv:1507.00814*, 2015.

[32] Lisa Lee, Benjamin Eysenbach, Emilio Parisotto, Eric Xing, Sergey Levine, and Ruslan Salakhutdinov. Efficient exploration via state marginal matching. *arXiv preprint arXiv:1906.05274*, 2019.

[33] Tejas D Kulkarni, Karthik Narasimhan, Ardavan Saeedi, and Josh Tenenbaum. Hierarchical deep reinforcement learning: Integrating temporal abstraction and intrinsic motivation. In *Advances in neural information processing systems*, 2016.

[34] Abhishek Gupta, Russell Mendonca, YuXuan Liu, Pieter Abbeel, and Sergey Levine. Meta-reinforcement learning of structured exploration strategies. In *Advances in Neural Information Processing Systems 31*. 2018.

[35] Wei Ren, Randal W Beard, and Ella M Atkins. A survey of consensus problems in multi-agent coordination. In *Proceedings of the 2005, American Control Conference, 2005.*, 2005.

[36] Ziyu Wang, Tom Schaul, Matteo Hessel, Hado Hasselt, Marc Lanctot, and Nando Freitas. Dueling network architectures for deep reinforcement learning. In *International conference on machine learning*, 2016.

[37] Tuomas Haarnoja, Aurick Zhou, Pieter Abbeel, and Sergey Levine. Soft actor-critic: Off-policy maximum entropy deep reinforcement learning with a stochastic actor. *arXiv preprint arXiv:1801.01290*, 2018.

[38] Johannes Ackermann, Volker Gabler, Takayuki Osa, and Masashi Sugiyama. Reducing overestimation bias in multi-agent domains using double centralized critics. *arXiv preprint arXiv:1910.01465*, 2019.

[39] Xueguang Lyu and Christopher Amato. Likelihood quantile networks for coordinating multi-agent reinforcement learning. In *Proceedings of the 19th International Conference on Autonomous Agents and MultiAgent Systems*, 2020.

[40] Zipeng Fu, Qingqing Zhao, and Weinan Zhang. Reducing overestimation in value mixing for cooperative deep multi-agent reinforcement learning. *ICAART*, 2020.

[41] Yan Zheng, Zhaopeng Meng, Jianye Hao, and Zongzhang Zhang. Weighted double deep multiagent reinforcement learning in stochastic cooperative environments. In *Pacific Rim international conference on artificial intelligence*, 2018.

[42] Tabish Rashid, Gregory Farquhar, Bei Peng, and Shimon Whiteson. Weighted qmix: Expanding monotonic value function factorisation, 2020.

[43] Thanh Thi Nguyen, Ngoc Duy Nguyen, and Saeid Nahavandi. Deep reinforcement learning for multiagent systems: A review of challenges, solutions, and applications. *IEEE transactions on cybernetics*, 2020.

[44] Yann LeCun, Sumit Chopra, Raia Hadsell, M Ranzato, and F Huang. A tutorial on energy-based learning. *Predicting structured data*, 1, 2006.

[45] Yann LeCun, Sumit Chopra, M Ranzato, and F-J Huang. Energy-based models in document recognition and computer vision. In *Ninth International Conference on Document Analysis and Recognition (ICDAR 2007)*, volume 1, 2007.

[46] Yee Whye Teh, Max Welling, Simon Osindero, and Geoffrey E Hinton. Energy-based models for sparse overcomplete representations. *Journal of Machine Learning Research*, 4, 2003.

[47] David J. C. MacKay. *Information Theory, Inference & Learning Algorithms*. Cambridge University Press, 2002.

[48] Brian Sallans and Geoffrey E Hinton. Reinforcement learning with factored states and actions. *Journal of Machine Learning Research*, 5, 2004.

[49] Sergey Levine and Pieter Abbeel. Learning neural network policies with guided policy search under unknown dynamics. In *Advances in Neural Information Processing Systems*, 2014.

[50] Brendan O'Donoghue, Remi Munos, Koray Kavukcuoglu, and Volodymyr Mnih. Combining policy gradient and q-learning. *arXiv preprint arXiv:1611.01626*, 2016.

[51] Tuomas Haarnoja, Haoran Tang, Pieter Abbeel, and Sergey Levine. Reinforcement learning with deep energy-based policies. *arXiv preprint arXiv:1702.08165*, 2017.

[52] Marc Toussaint. Robot trajectory optimization using approximate inference. In *Proceedings of the 26th annual international conference on machine learning*, 2009.

[53] Brian D Ziebart, Andrew L Maas, J Andrew Bagnell, and Anind K Dey. Maximum entropy inverse reinforcement learning. In *AAAI*, 2008.

[54] Brian D Ziebart. Modeling purposeful adaptive behavior with the principle of maximum causal entropy. 2010.

[55] John Schulman, Xi Chen, and Pieter Abbeel. Equivalence between policy gradients and soft q-learning. *arXiv preprint arXiv:1704.06440*, 2017.

[56] Tuomas Haarnoja. *Acquiring Diverse Robot Skills via Maximum Entropy Deep Reinforcement Learning*. PhD thesis, UC Berkeley, 2018.

[57] Tuomas Haarnoja, Kristian Hartikainen, Pieter Abbeel, and Sergey Levine. Latent space policies for hierarchical reinforcement learning. *arXiv preprint arXiv:1804.02808*, 2018.

[58] Lucian Buşoniu, Robert Babuška, and Bart De Schutter. Multi-agent reinforcement learning: An overview. In *Innovations in multi-agent systems and applications-1*. 2010.

[59] Ying Wen, Yaodong Yang, Rui Luo, Jun Wang, and Wei Pan. Probabilistic recursive reasoning for multi-agent reinforcement learning. *arXiv preprint arXiv:1901.09207*, 2019.

[60] Jordi Grau-Moya, Felix Leibfried, and Haitham Bou-Ammar. Balancing two-player stochastic games with soft q-learning. *arXiv preprint arXiv:1802.03216*, 2018.

[61] Ermo Wei, Drew Wicke, David Freelan, and Sean Luke. Multiagent soft q-learning. *arXiv preprint arXiv:1804.09817*, 2018.

[62] Kavosh Asadi and Michael L Littman. An alternative softmax operator for reinforcement learning. In *International Conference on Machine Learning*, 2017.

[63] Dipendra Misra, Mikael Henaff, Akshay Krishnamurthy, and John Langford. Kinematic state abstraction and provably efficient rich-observation reinforcement learning. *arXiv preprint arXiv:1911.05815*, 2019.

2