Self-Supervised Mixture-of-Experts by Uncertainty Estimation

Zhuobin Zheng,[1,2] **Chun Yuan,**[2] **Xinrui Zhu,**[1,2] **Zhihui Lin,**[1,2]
Yangyang Cheng,[1,2] **Cheng Shi,**[1,2] **Jiahui Ye**[1,3]
[1]Department of Computer Science and Technologies, Tsinghua University, Beijing, China
[2]Graduate School at Shenzhen, Tsinghua University, Shenzhen, China
[3]Tsinghua-Berkeley Shenzhen Institue, Tsinghua University, Shenzhen, China
{zhengzb16, zhuxr17, lin-zh14, cheng-yy13, shic17, yejh16}@mails.tsinghua.edu.cn, yuanc@sz.tsinghua.edu.cn

Abstract

Learning related tasks in various domains and transferring exploited knowledge to new situations is a significant challenge in Reinforcement Learning (RL). However, most RL algorithms are data inefficient and fail to generalize in complex environments, limiting their adaptability and applicability in multi-task scenarios. In this paper, we propose Self-Supervised Mixture-of-Experts (SUM), an effective algorithm driven by predictive uncertainty estimation for multi-task RL. SUM utilizes a multi-head agent with shared parameters as experts to learn a series of related tasks simultaneously by Deep Deterministic Policy Gradient (DDPG). Each expert is extended by predictive uncertainty estimation on known and unknown states to enhance the Q-value evaluation capacity against overfitting and the overall generalization ability. These enable the agent to capture and diffuse the common knowledge across different tasks improving sample efficiency in each task and the effectiveness of expert scheduling across multiple tasks. Instead of task-specific design as common MoEs, a self-supervised gating network is adopted to determine a potential expert to handle each interaction from unseen environments and calibrated completely by the uncertainty feedback from the experts without explicit supervision. To alleviate the imbalanced expert utilization as the crux of MoE, optimization is accomplished via decayed-masked experience replay, which encourages both diversification and specialization of experts during different periods. We demonstrate that our approach learns faster and achieves better performance by efficient transfer and robust generalization, outperforming several related methods on extended OpenAI Gym's MuJoCo multi-task environments.

1 Introduction

Reinforcement Learning (RL) (Sutton and Barto 1998) trains an agent to solve sequential decision-making problems through trial and error interactions with the environment. With the significant advance of deep neural networks as effective function approximators, Deep RL has been scaled and demonstrated success in various complex domains like Go game (Silver et al. 2017), video games (Mnih et al. 2015) and robotic control tasks (Gu et al. 2017). However, most typical RL methods are sample inefficient requiring substantial experiences and large computational cost before obtaining acceptable behaviors, especially when solving multiple related problems.

One effective direction for improving data efficiency is multi-task learning (Caruana 1997), which shares similar characteristics and transfers reusable representations across multiple tasks. Intuitively, by applying multi-task strategies, algorithms can accelerate the convergence and improve the performance of each task w.r.t. single-task learning, while requiring less training data overall. This further renders the agent applicable to complex real-world environments with robust generalization in different situations (Finn et al. 2017). However, in practice when RL combines with multi-task techniques such as shared structure, the learning process tends to be unstable due to the different complexity and reward schemes between tasks (Teh et al. 2017). We ascribe this negative effect to the interference caused by noised gradients from different domains, which misleads the training of vanilla multi-task approaches.

Another issue is deep RL agent "robustly" suffers from overfitting (Zhang et al. 2018), which negatively affects the performance of transfer. As deep RL algorithms are increasingly employed in critical problems such as autonomous control, healthcare, finance, and safety (Amodei et al. 2016), it is crucial to understand the generalization and adaptation abilities of trained agents before real-world deployment. Besides, the crux of multi-task RL is not only exploiting reusable representations but furthermore transferring them across domains effectively. Apart from techniques for mitigating overfitting, quantifying the uncertainty of specific tasks brings a more intuitive understanding of generalization capacity in deep models. In this case, multi-task RL agents with well-calibrated predictive uncertainty estimation can avert overconfident incorrect predictions (Lakshminarayanan, Pritzel, and Blundell 2017) and accomplish multiple tasks by robust generalization and efficient transfer.

In this paper, we propose Self-Supervised Mixture-of-Experts (SUM), an effective approach for multi-task RL aiming to improve both data efficiency and generalization capacity in multiple tasks by effective knowledge sharing and expert scheduling. Based on multi-head DDPG (Zheng et al. 2018), SUM extends the model to Mixture-of-Experts (MoE) architecture consisting of an agent and a self-supervised gating network, which is suitable for the multi-task environment. The agent is constructed by experts including multiple actor heads for exploration and multiple critic heads for evaluation. To counteract the adverse effects of overfitting, experts are enhanced by uncertainty estimation on known and unknown states to improve the Q-value evaluation and generalize robustly across multiple tasks. When interacting with tasks from an unknown distribution, a gating network determines a potential expert according to all uncertainty estimates of the state from experts. It leverages these uncertainty feedback as self-supervision to optimize without explicit supervision. Furthermore, we exploit decayed-masked experience replay to improve both early diversification and late specialization of experts in different stages. We present empirical experiments to analyze our algorithm dealing with a series of continuous control tasks on extended MuJoCo environments (Henderson et al. 2017).

The contributions of this work include:

- We introduce uncertainty estimation to the critic of DDPG for enhanced Q-value evaluation, which alleviates overfitting in an individual task and improves robust generalization ability across multiple tasks.
- We extend multi-head DDPG as MoE architecture with a gating network self-supervised completely by uncertainty estimates from experts without extra supervision, which improves data efficiency and performance substantially by effective knowledge sharing and expert scheduling.
- To tackle the imbalanced expert utilization, we exploit decayed-masked experience replay to motivate the experts to concentrate on different objectives during training.
- We evaluate our approach via extensive experiments from different perspectives, including uncertainty enhancement, generalization ability, and multi-task performance.

2 Related Work

In recent years, most research has focused on developing multi-task RL algorithms by transfer learning paradigm (Lazaric 2012). A primary series of approaches are based on policy distillation, which usually constructed as a student-teacher architecture, including (Rusu et al. 2015; Parisotto, Ba, and Salakhutdinov 2015; Teh et al. 2017). Some attempts focus on effective representation reuse in different perspectives, such as shared abstractions of state-action space (Borsa, Graepel, and Shawe-Taylor 2016), progressive network (Rusu et al. 2016), etc.

Uncertainty estimation has been increasingly prevalent as an effective method to reduce the risk of overfitting (Zhang et al. 2018) and understand the agent's generalization ability. Furthermore, with the advance of reliable uncertainty estimation, algorithms can tackle difficult specific issues in RL, including balancing between exploration and exploitation (Dearden, Friedman, and Russell 1998), avoiding collisions in unknown control tasks (Kahn et al. 2017), etc. However, for uncertainty in multi-task RL, there are relatively few methods available based on the Bayesian approach (Lazaric and Ghavamzadeh 2010; Wilson et al. 2007). Unlike these work, we exploit an auxiliary extension of the neural network (Nix and Weigend 1994) to quantify the predictive uncertainty in the critic of DDPG.

Mixture-of-experts (MoE) (Jacobs et al. 1991; Jordan and Jacobs 1994) is an effective ensemble learning approach that uses a gating function to specialize models to alleviate overfitting and improve the performance of complex tasks (Shazeer et al. 2017). This paper presents a novel framework based on MoE architecture tackling multi-task RL by two components, one based on the multi-head DDPG (Zheng et al. 2018) as experts and one using a self-supervised gating function for expert scheduling.

Self-supervised learning enables learning without explicit supervision and exploits unlabeled data to provide intrinsic representations as self-supervision. It has been prevalently pursued in computer vision (Donahue, Krähenbühl, and Darrell 2017; Doersch and Zisserman 2017) and RL (Shelhamer et al. 2016; Pathak et al. 2017). In this work, we first investigate the effectiveness of optimizing a gating function via uncertainty estimation in a completely self-supervised manner, instead of common supervised approaches.

3 Background

3.1 Preliminary and Notation

In the reinforcement learning setup, tasks are modeled as a *Markov decision process* (MDP) where an agent in a state s_t interacts with an environment E by applying an action a_t and receives a reward r_t together with a new state s_{t+1} at each discrete time step t. MDP consists of a state space $\mathcal{S}$, an action space $\mathcal{A}$, a state transition function $\mathcal{P} : \mathcal{S} \times \mathcal{A} \mapsto \mathcal{S}$, a reward function $\mathcal{R} : \mathcal{S} \times \mathcal{A} \mapsto \mathbb{R}$ and a discount factor $\gamma \in (0, 1]$, which can be formalized as $\langle \mathcal{S}, \mathcal{A}, \mathcal{P}, \mathcal{R}, \gamma \rangle$. We define the overall return from the state s_t as the discounted cumulated reward $R_t = r_t + \sum_{i=t+1}^{T} \gamma^{i-t} \mathcal{R}(s_i, a_i)$. The agent aims to find an optimal policy $\pi : \mathcal{S} \mapsto \mathcal{A}$, which can be stochastic or deterministic, to maximize the expected return from the initial state, $R^{\pi}(s_0) = \mathbb{E}_{r_{i\geq 0}, s_{i\geq 0} \sim E, a_{i\geq 0} \sim \pi}[R_0]$. Another key concept is the action value, also called Q-value, which is an estimate of the expected discounted return for selecting action a_t in state s_t and following policy π:

$$Q^{\pi}(s_t, a_t) = \mathbb{E}_{r_{i\geqslant t}, s_{i>t} \sim E, a_{i>t} \sim \pi}[R_t | s_t, a_t]. \quad (1)$$

3.2 Deep Deterministic Policy Gradient (DDPG)

This work builds upon DDPG (Lillicrap et al. 2016), a model-free off-policy algorithm for high-dimensional continuous action domains. DDPG utilizes an actor-critic architecture including a Q-value function (the critic Q) optimized by temporal difference (TD) in *policy evaluation* step,

$$\mathcal{L}(\theta^Q) = \frac{1}{n} \sum_i (r_i + \gamma Q'(s_{i+1}, \mu'(s_{i+1} | \theta^{\mu'}) | \theta^{Q'}) - Q(s_i, a_i | \theta^Q))^2, \quad (2)$$

and a policy function (the actor μ) updated by policy gradient (Silver et al. 2014) in *policy improvement* step,

$$\nabla_{\theta^{\mu}} \leftarrow \frac{1}{n} \sum_i \nabla_a Q(s_i, a | \theta^Q)|_{a=\mu(s_i|\theta^{\mu})} \nabla_{\theta^{\mu}} \mu(s_i | \theta^{\mu}), \quad (3)$$

where Q', μ' are target networks that slowly track Q, μ respectively in each update step,

$$\theta^{Q'} \leftarrow (1 - \tau)\theta^{Q'} + \tau \theta^Q, \theta^{\mu'} \leftarrow (1 - \tau)\theta^{\mu'} + \tau \theta^{\mu}, \quad (4)$$

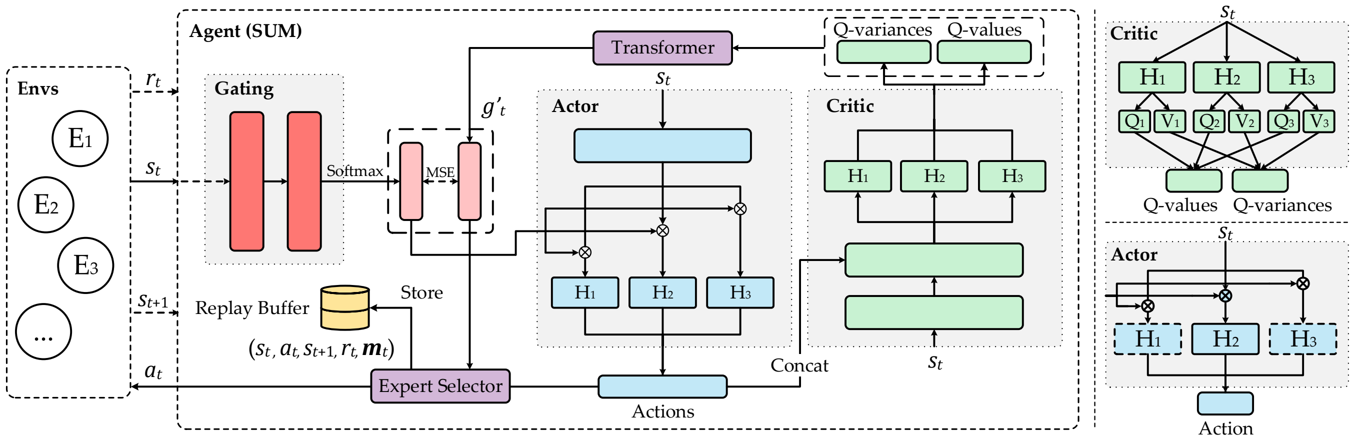

Figure 1: Left: Structure of SUM. When the agent receives a state from an unknown environment, the gating network (red) outputs a scheduling vector expecting a specific expert (actor head) with the highest activation to interact. Given the same state, each actor head (blue) produces an action together with respective Q-value and Q-variance from the paired critic head (green). Based on this information, a transformer (purple) generates a "ground truth" scheduling vector to select an expert to interact. Besides, experiences with decayed masks are stored for training both the experts and the gating function. Right: Mechanism of the multi-head critic evaluating during training and the multi-head actor activated by the gating network during testing.

with $\tau \in (0, 1]$. Benefiting from experience replay and target networks, DDPG is trainable on off-policy data with stability, which significantly brings sample efficiency.

4 Self-Supervised Mixture-of-Experts

In this work, we develop Self-Supervised Mixture-of-Experts (SUM) for multi-task RL (see Figure 1). For clarity of explanation, we first describe Uncertainty-Enhanced Multi-head DDPG as basic experts of MoE. Following that, we show how Self-Supervised Gating Network trains and works for expert specialization and scheduling.

4.1 Uncertainty-Enhanced Multi-head DDPG

Though vanilla DDPG (Lillicrap et al. 2016) shows impressive performance in continuous control tasks, it is susceptible to the randomness of complex environments, which may lead to data inefficiency and poor generalization. To counteract the adverse effects of the above problems, we introduce predictive uncertainty estimation into DDPG to capture the uncertainty of states from known and unknown tasks. Specifically, following (Nix and Weigend 1994) which is demonstrated effective in quantifying uncertainty, we extend the critic network to generate two values in the final layer, corresponding to the predicted mean Q-value $Q(s_i, a_i)$ and Q-variance $\sigma^2(s_i, a_i)$ (shown in Figure 1). By treating the observed Q-value as a sample from a Gaussian distribution with the predicted mean and variance, the critic is optimized by minimizing the negative log-likelihood (NLL) criterion,

$$\mathcal{L}(\theta^Q) = \frac{1}{n}\sum_i \left(\frac{\log \sigma^2(s_i, a_i)}{2} + \frac{(y_i - Q(s_i, a_i))^2}{2\sigma^2(s_i, a_i)}\right), \quad (5)$$

where y_i is the target Q-value and n is the mini-batch size.

With this enhancement, the critic is supervised towards producing not only accurate but reliable Q-value with well-calibrated uncertainty estimation reduction, which is robust to task shift. Besides, uncertainty-enhanced DDPG can detect and avert overfitting in advance by techniques like early stopping. In multi-task scenarios, utilizing only Q-value from the critic trained on a specific task is not sufficient and reliable for estimating Q-value in different or unseen tasks. However, a critic with auxiliary uncertainty estimation can improve understanding of the generalization and adaptation abilities in unknown domains, which further benefits multi-task performance. Notably, this is a general extension for estimating the uncertainty in the environment requiring few modifications to value-based RL algorithms.

Towards the goal of multi-task learning, we contribute predictive uncertainty estimation to SOUP (Zheng et al. 2018), an algorithm that combines multi-head bootstrapped DDPG with self-adaptive confidence to alleviate spotty Q-value in single-task RL. However, this rough state-based confidence strategy with the scarcity of crucial relevant information, such as actions selected by respective actors and intermediate representations in the critics, hampers its capacity of accurate evaluation, especially in multi-task domains with different complexity and reward schemes. Instead of crude external uncertainty approximator in SOUP, we directly extend each critic of multi-head DDPG with an auxiliary output unit as Q-variance. Since this unit is accessible to exploit the entire crucial information, it yields higher potentiality towards well-calibrated uncertainty estimation. Note that, different from SOUP selecting an action simply with the maximum Q-value, our approach takes not only Q-value but mainly Q-variance, into consideration for potential action selection more effectively and reasonably.

Robust Generalization Well-calibrated predictive uncertainty estimation assists the agent in robustly averting overconfident incorrect predictions. Utilizing the multi-head architecture simulates Deep Ensembles technique (Lakshminarayanan, Pritzel, and Blundell 2017), averaging predictions over multiple models, which significantly improves uncertainty quality and robustness by variance reduction. In this case, critics can produce more accurate Q-value estimates by "ensemble" improving the generalization ability against overfitting. Furthermore, when applying to unknown tasks, highly specialized experts with relatively lower uncertainty shows reliably higher confidence to tackle specific states, which benefits effective expert scheduling enhancing the overall generalization capacity across multiple tasks.

4.2 Self-Supervised Gating Network

The core of MoE is an effective gating function which performs expert specialization in training and expert scheduling in testing. We propose a self-supervised gating network calibrated completely by uncertainty estimates from experts in an end-to-end manner without explicit supervision.

Self-Supervised Training When the agent of MoE with K heads $\{Q, \mu\}_{1,\dots,K}$ interacts with the environment given s_t, the gating network G, parameterized by θ^G, generates a gating value as a scheduling vector $G(s_t)$, which denotes different degrees of expected preference for experts to interact in the current state s_t. Note that each actor head in multi-head DDPG is considered as an expert. Following that, each actor head produces an action candidate a_t^k while at the same time its paired critic head produces respective Q-value Q_k and Q-variance σ_k^2, which represents how uncertain the expert is about its evaluation of the action a_t^k. Based on the uncertainty estimates $\boldsymbol{\sigma}^2$, we construct the "ground truth" softmax gating $g'(s_t)$ (Shazeer et al. 2017) as self-supervision:

$$g'(s_t) = Softmax(H(s_t)). \tag{6}$$

Note that $H(s_t)$ is a one-hot vector only when an expert specializes and "masters" exploiting much more rewards in s_t with discriminatively highest Q-value (i.e., twice of others) and reliably lowest Q-variance concurrently, otherwise

$$H(s_t)_k = \begin{cases} +1, & \sigma_k^2 \in KeepTopX(\boldsymbol{\sigma}^2, x), \\ 0, & otherwise, \end{cases} \tag{7}$$

where $KeepTopX(\boldsymbol{v}, x)$ keeps only top x values in $\boldsymbol{v}$ (we set $x = 2$). In this case, $H(s_t)$ encourages the gating network to alleviate overfitting by activating experts with relatively high uncertainty which own more potential to explore and exploit more rewards in the state s_t compared with the lower ones, which may be stuck in local optima. In practice, $H(s_t)$ usually results in a one-hot vector in most cases (85%$\sim$). The action a_t is chosen to interact with s_t, receiving a new state s_{t+1} and a reward r_t, according to

$$a_t = \{a_t^k | k = \arg\max_k \{g'(s_t)\}\}. \tag{8}$$

A new transition $(s_t, a_t, s_{t+1}, r_t, \boldsymbol{m}_t)$ is stored in replay buffer for experience play, where $\boldsymbol{m}_t = \{m_t^k | m_t^k = g_k'(s_t)\}$ represents the probabilities of this transition trained by the experts. During training, a mini-batch of n samples are collected to optimize both experts as vanilla DDPG and the gating function by Mean Squared Error (MSE) as follows:

$$\mathcal{L}(\theta^G) = \frac{1}{n}\sum_i (g'(s_i) - G(s_i))^2, g'(s_i) = \boldsymbol{m}_i. \tag{9}$$

Algorithm 1 Self-Supervised Mixture-of-Experts

```
Input: number of heads $K$, environments $N$. batch size $n$
Initialize: Initialize critic and actor networks with $K$ heads $\{\theta_k^Q, \theta_k^\mu\}_{k=1}^K$ and copy weights to target networks $\{\theta_k^{Q'}, \theta_k^{\mu'}\}_{k=1}^K$. Initialize the gating network $\theta^G$, the replay buffer $R$, and multi-task environments $\{\mathrm{E}\}_{i=1}^N$
for episode $e = 1, E_{max}$ do
    Initialize a Gaussian noise $\mathcal{N}$ for action exploration
    Get state $s_0$ from randomly selected environment $\mathrm{E}_i$
    for step $t = 1, T_{max}$ do
        Generate a scheduling vector $G(s_t)$
        $K$ actor heads output actions
        $K$ critic heads output $Q$-values and $Q$-variances
        Generate targeted scheduling vector $g'(s_t)$ by (6)
        Select action $a_t$ according to (8) and apply $\mathcal{N}$
        Execute $a_t$ then observe reward $r_t$ and state $s_{t+1}$
        Assign mask $\boldsymbol{m}_t = g'(s_t)$
        Store transition $(s_t, a_t, s_{t+1}, r_t, \boldsymbol{m}_t)$ in $R$
        Randomly select $k$-th actor and critic heads
        Randomly sample a batch of $n$ transitions
        Update paired heads $\mu^k, Q^k$ according to (3) (5)
        Update the $k$-th target networks according to (4)
        Update the gating network according to (9)
        Decay $x_{DM}^t$ according to (10)
    end for
end for
```

A more detailed procedure is in Figure 1 and Algorithm 1.

Why Uncertainty Estimation as Self-Supervision Different from common supervised based MoE demanding large task-specific datasets with rich annotations (i.e., classification), utilizing self-supervision exploited from raw data is significant to accomplish the supervision-starved tasks without human intervention. Well-calibrated uncertainty estimation is demonstrated effective towards understanding the generalization capacity when tasks shift and simple to implement (Lakshminarayanan, Pritzel, and Blundell 2017). It is intuitively suitable for MoE in multi-task RL domain, where Q-value evaluation may be inaccurate and sufficient for decision-making across MDPs. In this situation, uncertainty estimation as a reliable criterion encourages expert specialization and scheduling appropriately driven by their "confidence" in a completely self-supervised manner.

Decayed Mask Experience Replay (DMER) The crux of MoE is balancing expert utilization in both diversification and specialization (Shazeer et al. 2017). We notice that the gating network sometimes tends to converge to a situation where it only produces large weights for the same few experts, which hampers the overall multi-task performance. We exploit decayed mask experience replay designed for training multiple RL experts diversely. Concretely, we replace x in Eq. (7) with x_{DM}^t, which is initialized as the number of heads K and decays throughout training:

$$x_{DM}^{t+1} = \lceil x_{DM}^t * \lambda^n \rceil, \lambda^n = \prod_{i=1}^{t} \lambda, \tag{10}$$

where λ is the decay rate. In the early period, experts are trained in the entire replay buffer for acquiring common behaviors to tackle basic tasks, which leads to diversification and alleviates the risk of imbalanced capacity and even single-expert domination. As x_{DM}^{t} decreases, experts with fundamental skills are provided different experiences according to the diverse masks generated from their uncertainty estimates and specialized in different directions.

Data Efficiency Bootstrap technique is demonstrated effective for deep exploration (Osband et al. 2016). DMER applies a masking mechanism similar as bootstrap, which induces diversity for efficient exploration, though in varying degrees during different stages from early diversification to late specialization. This breaks the limitation of common task-specific design where experts are only trained on assigned tasks with risk of overfitting. Besides, under the architecture including multiple experts with shared parameters, common knowledge across related tasks can be diffused and transferred quickly, which significantly improves data efficiency and the overall multi-task performance.

5 Experiments

We evaluate our approach on continuous control environment MuJoCo (Todorov, Erez, and Tassa 2012) and its multi-task extension (Henderson et al. 2017) (see Figure 2). In most environments, a specific robot is rewarded by moving forward as fast as possible. We demonstrate the effectiveness and analyze the performance of SUM by experiments:

1. We compare the impacts benefiting from uncertainty enhancement in the single-task environment.
2. We further examine the generalization ability in tackling multiple related tasks in difficult situations.
3. We evaluate data efficiency and expert utilization of our model on learning multiple tasks simultaneously.

In all cases, we use fully-connected network (see Figure 1), where hidden layer and head layer sizes are denoted by (N, $\boldsymbol{M}$). Unless otherwise stated, we adopt the network structure

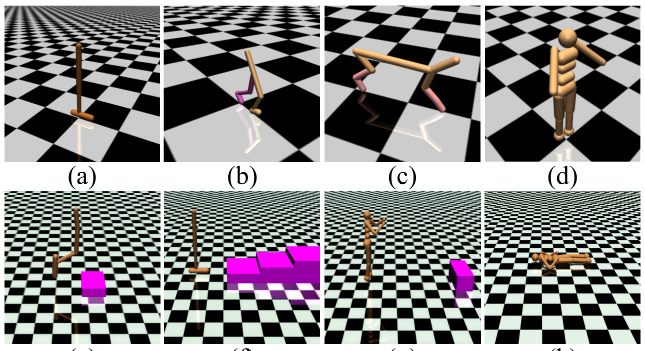

Figure 2: Illustrations of basic (top) and extended (bottom) locomotion tasks on MuJoCo: (a) Hopper, (b) Walker2D, (c) HalfCheetah, (d) Humanoid, (e) HopperWall, (f) HopperStairs, (g) HumanoidWall and (h) HumanoidStandup.

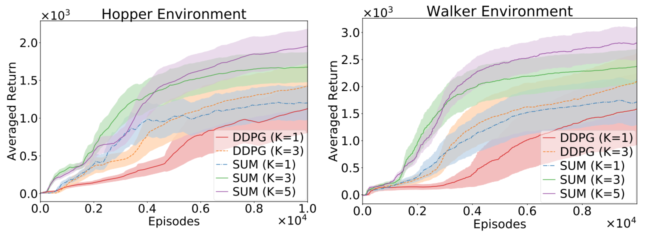

Figure 3: Performance of models with a different number of heads. The shaded area denotes the mean ± the standard deviation. SUM ($K = 3, 5$) with uncertainty enhancement outperforms other models in both reward and learning speed by reliably accurate Q-value evaluation.

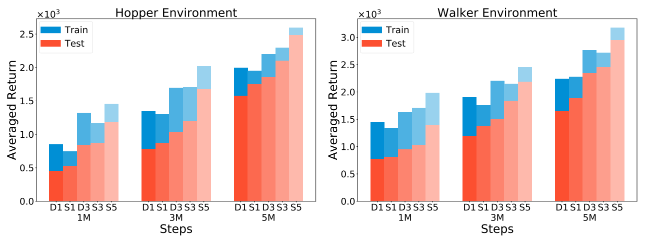

Figure 4: Generalization in different steps. DK and SK denote DDPG and SUM with K heads. The bar plots indicate SUM achieve robust generalization with less gap between the test and training performances, which demonstrates uncertainty enhancement effective for alleviating overfitting.

and common hyperparameters same as (Zheng et al. 2018): (256, 256, **128**) for the critic and (256, **128**) for the actor with Leaky ReLU activation. The gating network is (256, 128) with a softmax layer and updated by a learning rate $1e^{-4}$. These networks are trained by Adam (Kingma and Ba 2015) with a batch size $n = 1024$. Besides, we fix the decay rate for DMER $\lambda = 0.9997$. Figure 3, 5, 6 depict the averaged return by lines and standard deviation (std) return by shaded areas over 10k episodes, while Table 1, 2 tabulate the mean and std of the cumulative reward across 20 sample rollouts.

5.1 Uncertainty Enhancement

Accurate Q-value Evaluation in Individual Tasks Figure 3 shows that in both environments, equipped with the same head $K = 1$, SUM achieves slightly better performance faster than vanilla DDPG since SUM is optimized by NLL, which captures uncertainty, instead of MSE. Namely, training by the extra objective for uncertainty motivates the critic to be optimized in the direction of producing not only accurate but reliable Q-value with well-calibrated uncertainty estimation reduction. The auxiliary supervision further stabilizes the training with better performance and less oscillations. This active impact is magnified obviously with the increasing number of expert heads by comparison between multi-head DDPG and SUM with heads $K = 3$. In this case, shared architecture mimics "ensemble combination", which further achieves high-quality uncertainty estimation and reduces variance for accurate Q-value.

HalfCheetah	TRPO - After Training	TRPO - Fully Trained		**After Training**	**Fully Trained**		**Together**
SmallFoot	898.51 ± 363.85	2003.46 ± 933.59	55%	1441 ± 597	3460 ± 460	**58%**	$\mathbf{3748 \pm 478}$
BigFoot	1997.73 ± 101.36	2211.92 ± 65.81	10%	1532 ± 580	2392 ± 421	**36%**	$\mathbf{2754 \pm 389}$
SmallLeg	1494.03 ± 310.11	2327.16 ± 702.69	36%	2243 ± 516	$\mathbf{4233 \pm 322}$	**47%**	4175 ± 357
BigLeg	2101.74 ± 95.98	2269.78 ± 95.57	7%	2234 ± 415	3147 ± 347	**29%**	$\mathbf{3264 \pm 303}$
SmallThigh	1672.22 ± 110.11	2555.16 ± 96.80	35%	2041 ± 538	3346 ± 382	**39%**	$\mathbf{4143 \pm 291}$
BigThigh	2345.88 ± 381.33	2424.95 ± 94.19	3%	1805 ± 586	$\mathbf{2465 \pm 475}$	**27%**	2218 ± 514
SmallTorso	1845.20 ± 86.03	2294.72 ± 109.20	20%	1816 ± 425	$\mathbf{2870 \pm 245}$	**37%**	2548 ± 387
BigTorso	2620.46 ± 297.88	2686.13 ± 97.96	2%	2809 ± 551	3478 ± 423	**19%**	$\mathbf{3737 \pm 375}$

Table 1: Average and standard deviation ($\mu \pm \sigma$) of reward across a set of 20 sample rollouts on modified HalfCheetah tasks. The percentage represents the ratio of average changes between **After Training** and **Fully Trained** (in order). By comparison to TRPO (Henderson et al. 2017), SUM ($K = 3$) achieves an overall improvement of performance attacking catastrophic forgetting effectively. **Together** shows the reward obtained by SUM after training in all tasks simultaneously (without order).

Robust Generalization against Overfitting To demonstrate the effectiveness of uncertainty enhancement for tackling overfitting, we measure the generalization capacity in individual tasks by the gap between the test and training performances. Figure 4 represents the averaged loss evaluated in the same environments with different random seeds after training. Though sometimes multi-head DDPG performs better in training period, it suffers from overfitting during testing with different randomness, since overconfident Q-values mislead the agent into local optima. After training by NLL with uncertainty estimation, SUM generalizes robustly against overfitting tackling both tasks with less loss in performance when tested. Concretely, DDPG generalizes with a decreased performance of factor $\alpha \in [0.32, 0.51]$ while SUM decreases by $\alpha \in [0.05, 0.28]$ throughout training.

5.2 Generalization in Difficult Situations

In this experiment, we focus on the generalization ability of SUM attacking catastrophic forgetting and handling unseen tasks. The environments are a series of HalfCheetah variant tasks with modified body parts, which demands a common and robust behavior adaptive to any variants.

Generalization against Catastrophic Forgetting Due to the susceptibility of RL agents to different reward schemes and catastrophic forgetting, it is difficult to achieve an overall satisfying performance across all tasks simultaneously. We train SUM consecutively on each environment in the order as listed in Table 1. After having trained on a specific task, we evaluate SUM immediately on that environment across 20 sample rollouts, denoted by **After Training** (having seen all the previous environments). We repeat this procedure and finally evaluate the reward across 20 sample rollouts on each environment, denoted by **Fully Trained** (having seen all the environments). We measure the generalization capacity against catastrophic forgetting by the gap between After Training and Fully Trained in all tasks as (Henderson et al. 2017). To ensure comparability, we adopt the same network with hidden layers (100, 50, **25**).

Table 1 shows the difference between TRPO and SUM attacking catastrophic forgetting when learning in order. We ascribe this to the difficulty of tackling task shift by only one single policy with risk of overfitting, which may limit the improvement of performance in multiple tasks. As an auxiliary criterion, uncertainty estimation works by understanding the generalization of different experts for effective expert scheduling. With this enhancement, SUM outperforms TRPO not only in substantially higher reward but robust generalization against catastrophic forgetting. Moreover, it is also feasible for SUM to learn concurrently without order and achieve a satisfying performance, which means SUM can be trained by samples from different tasks with varying reward schemes simultaneously (shown as **Together**). In this situation, SUM can capture each expert's uncertainty estimates of states from various domains. On the one hand, experts are trained with a shared structure for efficient knowledge sharing. On the other hand, the gating network self-supervised by uncertainty estimation can determine the most reliably potential expert to tackle the task.

Env. Train (3M)		Env. Test	
SmallFoot	5628 ± 139	SmallLeg	5298 ± 288
BigFoot	4487 ± 289	BigLeg	5083 ± 134
SmallThigh	5025 ± 137	SmallTorso	5112 ± 207
BigThigh	5644 ± 171	BigTorso	5391 ± 167
SmallFoot	5058 ± 490	BigFoot	4326 ± 333
SmallLeg	5822 ± 367	BigLeg	5741 ± 132
SmallThigh	5346 ± 436	BigThigh	5681 ± 325
SmallTorso	5432 ± 258	BigTorso	5837 ± 360
Wall	4304 ± 521	Wall	4017 ± 501
All Envs	3525 ± 614	Wall	$\mathbf{3955 \pm 451}$

Table 2: Performance of SUM ($K = 3$) tested on unseen tasks. The last row shows results on a different environment, Wall-v0, after having trained on all modified body tasks is competitive with that trained on the individual Wall-v0 task.

Generalization in Unseen Environments Table 2 shows when tested on unseen environments, SUM can robustly generalize based on the common knowledge learned before. In particular, HalfCheetahWall-v0 differently rewards

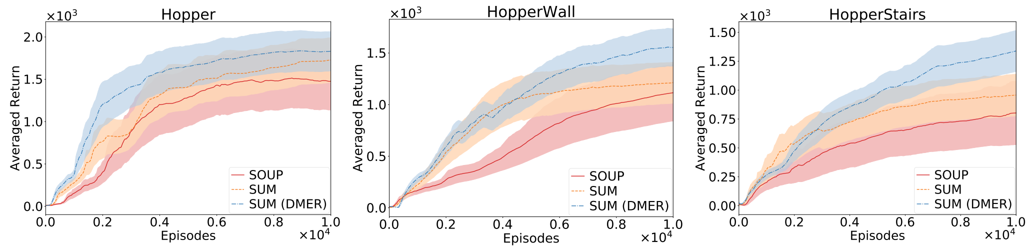

Figure 5: Performance of models with heads $K = 3$ on learning multiple Hopper tasks without order and any prior knowledge (i.e., the distribution of all tasks) simultaneously. SUM with DMER outperforms others in both reward and learning speed.

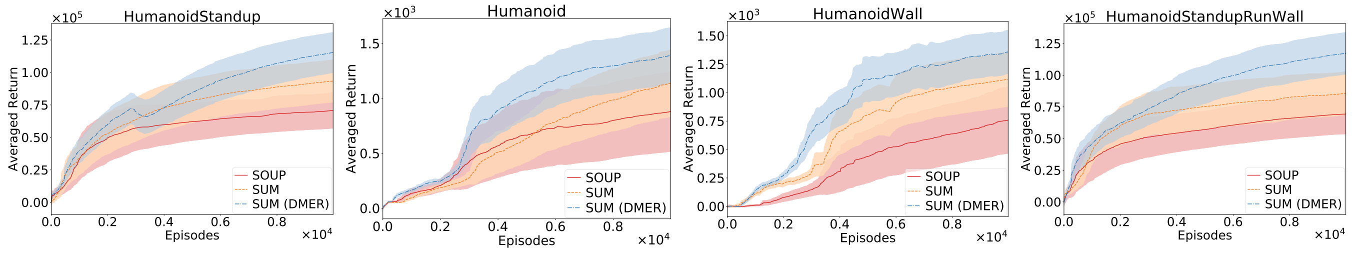

Figure 6: Performance of models with heads $K = 5$ on learning multiple Humanoid tasks without order and any prior knowledge (i.e., the distribution of all tasks) concurrently. SUM with DMER outperforms others in both reward and learning speed.

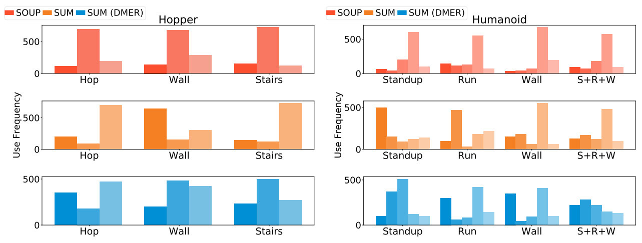

Figure 7: Expert utilization of different models during testing, shown by the use frequencies in 1K steps. DMER motivates relatively evener utilization and better specialization.

an agent to step over a wall, where common multi-task agents tested on this unseen environment are always stuck in front of the wall. However, SUM trained on modified body variant tasks achieves a competitive performance as single-task learning. Utilizing self-supervised expert scheduling, experts with different potential for tackling each state are allocated more appropriately and effectively.

5.3 Multi-task Performance

We conduct two group of multi-task experiments including Hopper that learns to hop on flat ground, walls, and stairs, and Humanoid that learns to stand up, walk, and pass a wall (see Figure 2). We compare SUM with SOUP (Zheng et al. 2018), an approach based on multi-head DDPG with a confidence strategy, to emphasize clearly the strengths of self-supervised expert scheduling and DMER. Figure 5, 6 shows that SUM not only accelerates the training, but achieves more cumulative rewards by data efficiency. Under multi-head MoE architecture, common representations and knowledge are shared and diffused quickly, which boosts the learning with fewer experiences. Furthermore, experts are well-generalized for scheduling, which improves performance.

Expert Utilization and Specialization Note that, StandupRunWall is an extremely challenging task since one of the component environment, HumanoidStandup, provides denser rewards via a different reward scheme making a robot stand up as fast as possible. Though we apply reward scale techniques, SUM and SOUP suffer from imbalanced expert utilization dominated by the expert which is always trained by rewards from HumanoidStandup.

To counteract the adverse effects, SUM solves the problem on the strength of decayed mask experience replay (DMER). In the early period, DMER encourages experts to acquire basic behaviors with equal access to the overall experiences towards exploration in various directions. In this case, different from ensemble evaluation in SOUP with risk of single-head and individual-task domination, no experts can dominate the training in SUM with DMER. While in the late period, samples are masked according to uncertainty estimates from experts, which works similar as bootstrap technique limiting the experts to only adapt to fewer and diverse environments. It further leads to expert specialization for efficient expert scheduling tackling multiple tasks. Figure 7 illustrates that SOUP suffers from single-head domination. Though SUM avoids this issue, imbalanced expert utilization still hampers its performance. Taking full advantage of DMER, SUM can motivate an expert to specialize in one or two tasks averting domination. Figure 7 represents that SUM learns to accomplish each task by activating several heads more frequently and tackle StandupRunWall task by balanced utilization and robust generalization.

6 Conclusion

In this paper, we propose SUM, an effective algorithm for multi-task reinforcement learning, improving both data efficiency in each task and generalization ability across multiple tasks. SUM utilizes multi-head DDPG as experts enhanced by predictive uncertainty estimation, which improves generalization capacity against overfitting. A self-supervised gating network trained by uncertainty feedback from experts is exploited to achieve efficient expert scheduling, which improves data efficiency and performance across multiple tasks. Moreover, to alleviate the imbalanced expert utilization, we adopt decayed mask experience replay to motivate early diversification and late specialization.

To the best of our knowledge, our approach is the first work to investigate the effectiveness of exploiting predictive uncertainty estimation in multi-task reinforcement learning in an end-to-end self-supervised manner. We demonstrate the effectiveness, performance and robust generalization ability of our algorithm on extended MuJoCo multi-task environments, especially in difficult situations.

7 Acknowledgments

This work is supported by the NSFC project (No. U1833101), Shenzhen Foundational Research Project (No. JCYJ20160428182137473) and the Joint Research Center of Tencent & Tsinghua University.

References

Amodei, D.; Olah, C.; Steinhardt, J.; Christiano, P.; Schulman, J.; and Mané, D. 2016. Concrete problems in ai safety. *arXiv:1606.06565*.

Borsa, D.; Graepel, T.; and Shawe-Taylor, J. 2016. Learning shared representations in multi-task reinforcement learning. *arXiv:1603.02041*.

Caruana, R. 1997. Multitask learning. *Machine learning* 28(1).

Dearden, R.; Friedman, N.; and Russell, S. 1998. Bayesian q-learning. In *AAAI/IAAI*.

Doersch, C., and Zisserman, A. 2017. Multi-task self-supervised visual learning. In *ICCV*.

Donahue, J.; Krähenbühl, P.; and Darrell, T. 2017. Adversarial feature learning.

Finn, C.; Yu, T.; Fu, J.; Abbeel, P.; and Levine, S. 2017. Generalizing skills with semi-supervised reinforcement learning. In *ICLR*.

Gu, S.; Holly, E.; Lillicrap, T.; and Levine, S. 2017. Deep reinforcement learning for robotic manipulation with asynchronous off-policy updates. In *ICRA*.

Henderson, P.; Chang, W.-D.; Shkurti, F.; Hansen, J.; Meger, D.; and Dudek, G. 2017. Benchmark environments for multitask learning in continuous domains. *arXiv:1708.04352*.

Jacobs, R. A.; Jordan, M. I.; Nowlan, S. J.; and Hinton, G. E. 1991. Adaptive mixtures of local experts. *Neural computation* 3(1).

Jordan, M. I., and Jacobs, R. A. 1994. Hierarchical mixtures of experts and the em algorithm. *Neural computation* 6(2).

Kahn, G.; Villaflor, A.; Pong, V.; Abbeel, P.; and Levine, S. 2017. Uncertainty-aware reinforcement learning for collision avoidance. *arXiv:1702.01182*.

Kingma, D., and Ba, J. 2015. Adam: A method for stochastic optimization. In *ICLR*.

Lakshminarayanan, B.; Pritzel, A.; and Blundell, C. 2017. Simple and scalable predictive uncertainty estimation using deep ensembles. In *NIPS*.

Lazaric, A., and Ghavamzadeh, M. 2010. Bayesian multi-task reinforcement learning. In *ICML*.

Lazaric, A. 2012. Transfer in reinforcement learning: a framework and a survey.

Lillicrap, T. P.; Hunt, J. J.; Pritzel, A.; Heess, N.; Erez, T.; Tassa, Y.; Silver, D.; and Wierstra, D. 2016. Continuous control with deep reinforcement learning. In *ICLR*.

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

Nix, D. A., and Weigend, A. S. 1994. Estimating the mean and variance of the target probability distribution. In *Neural Networks, 1994. IEEE World Congress on Computational Intelligence., 1994 IEEE International Conference On*, volume 1.

Osband, I.; Blundell, C.; Pritzel, A.; and Van Roy, B. 2016. Deep exploration via bootstrapped dqn. In *NIPS*.

Parisotto, E.; Ba, J. L.; and Salakhutdinov, R. 2015. Actor-mimic: Deep multitask and transfer reinforcement learning. *arXiv:1511.06342*.

Pathak, D.; Agrawal, P.; Efros, A. A.; and Darrell, T. 2017. Curiosity-driven exploration by self-supervised prediction. In *ICML*, volume 2017.

Rusu, A. A.; Colmenarejo, S. G.; Gulcehre, C.; Desjardins, G.; Kirkpatrick, J.; Pascanu, R.; Mnih, V.; Kavukcuoglu, K.; and Hadsell, R. 2015. Policy distillation. *arXiv:1511.06295*.

Rusu, A. A.; Rabinowitz, N. C.; Desjardins, G.; Soyer, H.; Kirkpatrick, J.; Kavukcuoglu, K.; Pascanu, R.; and Hadsell, R. 2016. Progressive neural networks. *arXiv:1606.04671*.

Shazeer, N.; Mirhoseini, A.; Maziarz, K.; Davis, A.; Le, Q.; Hinton, G.; and Dean, J. 2017. Outrageously large neural networks: The sparsely-gated mixture-of-experts layer. In *ICLR*.

Shelhamer, E.; Mahmoudieh, P.; Argus, M.; and Darrell, T. 2016. Loss is its own reward: Self-supervision for reinforcement learning. *arXiv:1612.07307*.

Silver, D.; Lever, G.; Heess, N.; Degris, T.; Wierstra, D.; and Riedmiller, M. 2014. Deterministic policy gradient algorithms. In *ICML*.

Silver, D.; Schrittwieser, J.; Simonyan, K.; Antonoglou, I.; Huang, A.; Guez, A.; Hubert, T.; Baker, L.; Lai, M.; Bolton, A.; et al. 2017. Mastering the game of go without human knowledge. *Nature* 550(7676).

Sutton, R. S., and Barto, A. G. 1998. *Reinforcement learning: An introduction*, volume 1.

Teh, Y.; Bapst, V.; Czarnecki, W. M.; Quan, J.; Kirkpatrick, J.; Hadsell, R.; Heess, N.; and Pascanu, R. 2017. Distral: Robust multitask reinforcement learning. In *NIPS*.

Todorov, E.; Erez, T.; and Tassa, Y. 2012. Mujoco: A physics engine for model-based control. In *IROS*.

Wilson, A.; Fern, A.; Ray, S.; and Tadepalli, P. 2007. Multi-task reinforcement learning: a hierarchical bayesian approach. In *ICML*.

Zhang, C.; Vinyals, O.; Munos, R.; and Bengio, S. 2018. A study on overfitting in deep reinforcement learning. *arXiv:1804.06893*.

Zheng, Z.; Yuan, C.; Lin, Z.; Cheng, Y.; and Wu, H. 2018. Self-adaptive double bootstrapped ddpg. In *IJCAI*.