中国科学技术大学

研究生学位论文开题报告

论文题目 分布式深度强化学习训练优化问题研究

学生姓名 张红杰

学生学号 BA17011022

指导教师 李京 教授

所在院系 计算机科学与技术学院

学科专业 计算机软件与理论

研究方向 分布式系统

填表日期 2019 年 1 月 3 日

中国科学技术大学研究生院培养办公室

二零零四年五月制表

说　明

1. 抓好研究生学位论文开题报告工作是保证学位论文质量的一个重要环节。为加强对研究生培养的过程管理，规范研究生学位论文的开题报告，特印发此表。

2. 研究生一般应在课程学习结束之后的第一个学期内主动与导师协商，完成学位论文的开题报告。

3. 研究生需在学科点内报告，听取意见，进行论文开题论证。

4. 研究生论文开题论证通过后，在本表末签名后将此表交所在学院教学办公室备查。

一．选题依据

1．阐述该选题的研究意义，分析该研究课题国内外研究的概况和发展趋势。

研究背景

（1） 深度强化学习（DRL）

强化学习是机器学习大家族中的一大类算法，使用强化学习能够让机器学会如何在环境中拿到高分，表现出优秀的成绩，而这些成绩背后，是算法不断的在环境中试错，不断探索环境并积累经验，学习经验。近年来，和结合深度学习，使得强化学习有了进一步的运用，比如玩 Atari 游戏[1]、围棋[2]以及即时策略游戏 DOTA 等都取得了超越人类的成绩。强化学习除了应用在模拟游戏中，也开始在自然语言处理[3]、智能驾驶、智能医疗等领域崭露头角。在深度强化学习中，深度的意思是通过多层的神经网络结构和非线性变换，提取出高维输入中的抽象特征，强化学习利用这些抽象特征完成策略的学习。深度强化学习目前是人工智能领域的一个新的研究热点，通过端到端的学习方式实现从原始输入到目标输出的映射。

强化学习中唯一的信息就是奖励。智能体在采取一个动作后会获得环境的奖励，奖励信息告诉智能体动作的好坏。智能体会学习历史经验，调整当前策略，避免负奖励的动作，尽量执行正奖励的动作。智能体将以最大化整个交互序列的累积奖励作为目标，学得最优策略。详细的运行流程如图 1 所示。智能体用神经网络表示，输入是当前环境状态，输出是策略，即当前状态下该采取的动作分布。智能体根据当前环境状态 s_t 预测动作 a_t，环境执行动作 a_t，返回即时奖励 r_{t+1} 以及下一个状态 s_{t+1}。所有的状态、动作以及奖励将作为经验数据更新神经网络参数。

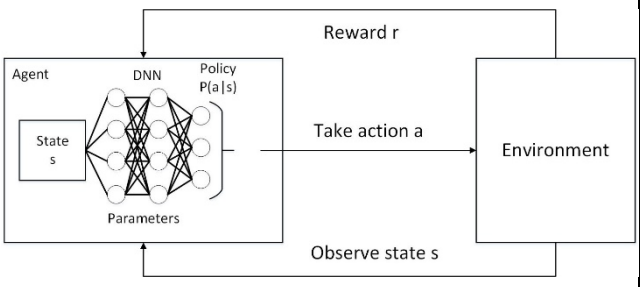

Figure 1 强化学习流程

（2） 并行分布式深度强化学习（DDRL）

训练一个深度神经网络需要大量数据，在单块 GPU 上训练 DQN（Deep Q-learning Network 一种 DRL 算法）完成一个 Atari 游戏需要 12-14 天时间[4]。训练如此慢的原因除了硬件，还有算法上的问题。这里有两个瓶颈，一是采样慢，样本需要神经网络与环境交互产生，而神经网络预测以及环境状态修改都需要消耗时间；二是训练慢，深度神经网络的预测和梯度计算也需要大量时间。为了加速 DRL 的训练，研究人员设计出多种并行分布式的训练框架和算法，将原本需要训练数十天的任务缩短到几小时。并行分布式框架同时解决采样慢和训练慢的问题。为了加速采样，使用多个智能体与环境的交互进程，同时运行上述的 DRL 执行流程，相同时间产生成倍的样本量。为了加速训练，将数据或模型拆分到不同机器上，并行运行。最常见的是采用数据并行的方式（如图 2 所示），将训练数据拆分成多份，每份数据发送给不同的机器，每台机器上根据本地数据计算神经网络梯度值，通过同步[19]或异步[20]的方式将梯度发送给一个参数服务器（Parameter Server），在参数服务器上合并各个计算节点上传的梯度并更新神经网络，再将最新的神经网络下发至各个计算节点继续运行。

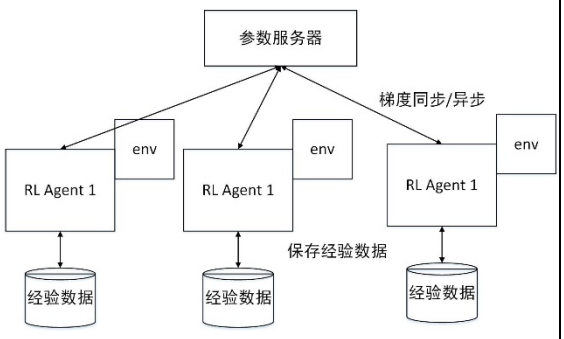

Figure 2 数据并行分布式训练

采取同步还是异步更新需要根据不同物理环境以及神经网络复杂度的具体情况具体分析。在通信代价不构成瓶颈的前提下，同步方式是一种很好的选择。而异步随机梯度下降无需等待慢节点，吞吐量大，但存在梯度陈旧问题，可能会导致整个训练过程不收敛。当通信成为分布式计算的瓶颈时，可以使用异步算法在更短的时间内训练出一个不错的模型。在实践中需要根据自己的应用场景，参照每种算法的优缺点，选择最合适的算法。

研究概况和发展趋势

分布式深度强化学习刚刚起步，仍然有许多问题存在，优化空间还很大，我们的研究将针对分布式训练中的多个问题进行分析和优化。我们在现有研究的基础上，进一步优化分布式训练，主要围绕一下三个研究问题进行分析和优化：

（1） 基于神经网络压缩技术加速 DRL 的训练。在 DRL 训练流程中，采样过程会占用很大比例的时间，现有的并行分布式 DRL 框架和算法都是通过同时运行多个智能体更快的产生样本。目前比较成熟的框架包括 Gorila[5]、DDQL[6]、A3C[7]、GA3C[8]、PAAC[9]、ELF[10]、IMPALA[11]、Ape-X[12]等等。然而，在单个智能体采样过程中，神经网络预测部分占用时间更多，通过神经网络压缩以及加速的技术能有效解决采样慢的问题。在神经网络压缩技术中，最重要的两类技术是剪枝与量化。剪枝[13]技术根据神经网络权值的特点而设计，在神经网络训练结束后包含大量接近 0 的参数，通过删除这些近似 0 的参数并不会对最后的预测准确率有任何影响。量化[12]技术是将神经网络 32 位浮点数的权值直接量化为 16 位或 8 位，减少存储代价的同时能有效加速预测。二值化[15]作为量化技术的极端版本，直接将神经网络权值用-1 和 1 表示。在 CPU 上通过异或指令能快速运算，但预测准确率会有所下降。

（2） 分阶段加速分布式深度强化学习训练问题。强化学习训练和监督学习训练的收敛过程存在明显的差异。强化学习由于前期随机探索，产生的样本质量差，前期训练中策略质量很难提升。而监督学习由于样本是通过人类专家标记好的，所以前期神经网络的准确率会是快速增长。为了避免前期的随机探索，DeepMind 提出使用预训练的模型监督当前神经网络的训练[16]。该方法确实能有效避免前期随机探索，相当于有了任务的先验知识。但该方法不适用于新的领域和任务，因为不存在预训练的模型。训练中期，随着智能体策略提升，经验数据中样本质量也逐渐变好。为了提高样本利用率，DeepMind 提出优先级经验数据池[17]，根据实际得分与估计得分的差异给经验数据池中的样本确定优先级，提高样本利用率，加速训练。但只能针对 off-policy 类的算法，这类算法有经验数据池的概念。On-policy 类算法没有经验数据池，所以无法直接使用。

（3） 基于策略集成的 DDRL 训练框架。在 DRL 训练分阶段中，最后的阶段是神经网络趋于收敛，策略质量不再显著提升。此时采用集成学习，通过集成各个 local model 的策略函数得到更高质量的策略。微软研究院已经探索过集成技术在分布式深度学习训练中的效果。Ensemble-Compression[18]是一种集成训练的实现，但单个 worker 节点运算速度成为瓶颈。

研究意义

并行分布式技术能有效加速深度强化学习的训练，现有并行分布式技术在强化学习领域仍有许多问题需要解决，比如如何加速单个智能体的采样，如何利用 DRL 阶段性特征加速训练，如何利用集成学习技术实现扩展性更强的 DDRL 训练框架。快速的 DRL 训练能加快强化学习新算法的研究和迭代，以及强化学习应用开发。

２．国内外主要参考文献（列出作者、论文名称、期刊名称、出版年月）。

[1] Mnih V, Kavukcuoglu K, Silver D, et al. Playing atari with deep reinforcement learning[J]. arXiv preprint arXiv:1312.5602, 2013.

[2] Silver D, Schrittwieser J, Simonyan K, et al. Mastering the game of Go without human knowledge[J]. Nature, 2017, 550(7676): 354.

[3] Wang W Y, Li J, He X. Deep Reinforcement Learning for NLP[J]. Proceedings of ACL 2018, Tutorial Abstracts, 2018: 19-21.

[4] Mnih V, Kavukcuoglu K, Silver D, et al. Human-level control through deep reinforcement learning[J]. Nature, 2015, 518(7540): 529.

[5] Nair A, Srinivasan P, Blackwell S, et al. Massively parallel methods for deep reinforcement learning[J]. arXiv preprint arXiv:1507.04296, 2015.

[6] Ong H Y, Chavez K, Hong A. Distributed deep Q-learning[J]. arXiv preprint arXiv:1508.04186, 2015.

[7] Mnih V, Badia A P, Mirza M, et al. Asynchronous methods for deep reinforcement learning[C]//International conference on machine learning. 2016: 1928-1937.

[8] Babaeizadeh M, Frosio I, Tyree S, et al. GA3C: GPU-based A3C for deep reinforcement learning[J]. CoRR abs/1611.06256, 2016.

[9] Clemente A V, Castejón H N, Chandra A. Efficient Parallel Methods for Deep Reinforcement Learning[J]. arXiv preprint arXiv:1705.04862, 2017.

[10] Tian Y, Gong Q, Shang W, et al. Elf: An extensive, lightweight and flexible research platform for real-time strategy games[C]//Advances in Neural Information Processing Systems. 2017: 2659-2669.

[11] Espeholt L, Soyer H, Munos R, et al. IMPALA: Scalable distributed Deep-RL with importance weighted actor-learner architectures[J]. arXiv preprint arXiv:1802.01561, 2018.

[12] Horgan D, Quan J, Budden D, et al. Distributed prioritized experience replay[J]. arXiv preprint arXiv:1803.00933, 2018.

[13] Han S, Pool J, Tran J, et al. Learning both weights and connections for efficient neural network[C]//Advances in neural information processing systems. 2015: 1135-1143.

[14] Lin D, Talathi S, Annapureddy S. Fixed point quantization of deep convolutional networks[C]//International Conference on Machine Learning. 2016: 2849-2858.

[15] Rastegari M, Ordonez V, Redmon J, et al. Xnor-net: Imagenet classification using binary convolutional neural networks[C]//European Conference on Computer Vision. Springer, Cham, 2016: 525-542.

[16] Schmitt S, Hudson J J, Zidek A, et al. Kickstarting Deep Reinforcement Learning[J]. arXiv preprint arXiv:1803.03835, 2018.

[17] Schaul T, Quan J, Antonoglou I, et al. Prioritized experience replay[J]. arXiv preprint arXiv:1511.05952, 2015.

[18] Sun S, Chen W, Bian J, et al. Ensemble-compression: A new method for parallel training of deep neural networks[C]//Joint European Conference on Machine Learning and Knowledge Discovery in Databases. Springer, Cham, 2017: 187-202.

[19] Goyal P, Dollár P, Girshick R, et al. Accurate, large minibatch SGD: training imagenet in 1 hour[J]. arXiv preprint arXiv:1706.02677, 2017.

[20] Zheng S, Meng Q, Wang T, et al. Asynchronous stochastic gradient descent with delay compensation[J]. arXiv preprint arXiv:1609.08326, 2016.

[21] Hinton G, Vinyals O, Dean J. Distilling the knowledge in a neural network[J]. arXiv preprint arXiv:1503.02531, 2015.

二．已取得的与论文研究内容相关的成果

已发表或被接收发表的文章目录或其它相关研究成果。

已发表文章：

Zhang H, Shu G, Liao S, et al. Workload-Aware VM Consolidation in Cloud Based on Max-Min Ant System[C]//International Conference on Cloud Computing and Security. Springer, Cham, 2017: 176-188.

相关研究成果：

研究内容一神经网络压缩技术解决采样慢的问题已完成，论文《Accelerating the Deep Reinforcement Learning with Neural Network Compression》投稿至 IJCNN 2019。

三．研究内容和研究方法

主要研究内容及预期成果，拟采用的研究方法、技术路线、实验方案的可行性分析。

研究内容

概述：

本论文拟在现有的并行分布式深度强化学习训练框架与算法的研究基础上，进一步提升和改进。具体的，我们针对如何加速 DRL 采样和加速 DRL 训练提出三个研究问题。如图 3 所示，首先是在加速 DRL 采样方面，我们深入分析智能体采样具体过程，发现许多任务上，神经网络预测过程成为采样的瓶颈，以此提出基于神经网络压缩技术加速 DDRL 训练的框架，在加速采样的同时不影响最终的策略质量。然后我们针对 DDRL 训练过程中的特点，将 DDRL 训练划分为三个阶段，即前期随机探索阶段、中期策略提升阶段以及后期策略稳定阶段。不同的阶段有不同的特点，采用不同的技术进行优化加速，最终实现整个 DDRL 训练的加速。最后是针对 DDRL 后期策略稳定阶段进行拓展，我们设计了基于策略集成的 DDRL 训练框架，旨在结合现有 DDRL 框架以及集成训练，增强训练框架的可扩展性。

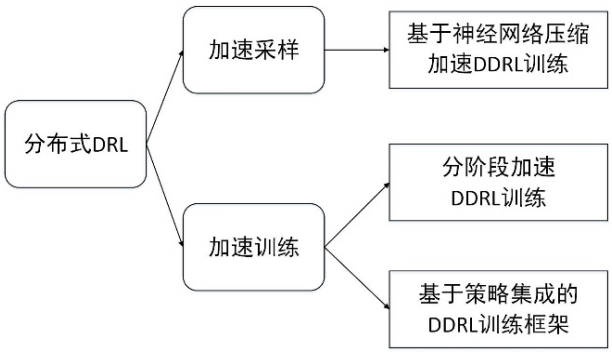

Figure 3 研究内容

研究内容一：基于神经网络压缩加速 DDRL 训练

（1）研究动机

现有的并行分布式深度强化学习训练框架和算法通过同时运行多个智能体与环境的交互进程快速产生样本，但在单个智能体与环境进程里的采样复杂度并没有降低。DRL 的采样过程包含两个步骤，首先是智能体根据当前环境状态 s_t 预测动作 a_t，然后环境执行动作 a_t，返回即时奖励 r_{t+1} 以及下一个状态 s_{t+1}。$(s_t, a_t, r_{t+1}, s_{t+1})$作为经验数据用于更新神经网络参数。那么采样时间等于神经网络预测时间加上状态更新时间。在许多任务上，神经网络预测过程占主要部分，比如围棋自博弈的时候。我们也在一些模拟游戏上做了详细的实验，不同的 CPU/GPU 配置以及神经网络架构下的网络预测、状态改变以及网络训练的时间占比。在并行训练框架 GA3C 上，发现神经网络预测时间占比约 40%，如果采样过程是运行在 CPU 上甚至是移动设备上，其时间占比更大。所以通过神经网络压缩和加速能有效减少采样时间。

神经网络压缩与加速技术目前已经非常成熟，在研究概况与发展现状部分我们简要介绍了重要的剪枝和量化技术。但这些压缩加速技术无法直接应用于 DDRL 训练的场景中。传统的神经网络压缩技术主要是针对已经训练好的网络进行压缩。DDRL 下的神经网络仍然处在训练过程中，目标网络的权值会不断变化。剪枝和量化后的网络参数变化大，每次压缩都需要多次再训练，而且不能保证压缩后的网络与原始网络有相似的输出。此外，压缩后的神经网络输出与原始网络输出不会完全一致，毕竟是两个不同的神经网络。用压缩后的网络去采样会影响整个 DRL 的收敛性。我们不仅要加速采样，同时应该保证算法仍然具有良好的收敛性。我们的研究问题就是如何保证策略质量的情况下利用神经网络压缩加速整个 DDRL 训练过程。

（2）技术路线

典型的深度强化学习训练中的神经网络既用于与环境交互产生样本，同时也用于训练。我们设计了一种基于神经网络压缩的 DDRL 训练框架（如图 4 所示），不同于原始深度强化学习训练，我们的框架下包含了两个不同大小的神经网络，其中小网络（student）预测速度快于大网络（teacher）。小网络与环境交互，快速产生样本。这些样本会被用于训练大网络，我们最终的目标是训练大网络。因为小网络本身是大网络的替代者，所以其输出要和大网络保持近似。当大网络输出有明显改变时，我们将其输出分布迁移至小网络。在深度强化学习中，神经网络的输出根据不同算法有不同的含义。主要分为两类，一类是价值函数（当前状态的价值），一类是策略分布（动作概率分布）。为了让 student 网络学习 teacher 的输出，针对两种类型我们分别采用 MSE（Mean Squared Error）以及 KL（Kullback-Leibler）散度作为小网络的损失函数。其中 MSE 用于价值函数（实数值），KL 散度用于动作概率分布。KL 散度是衡量两个概率分布的差异，值越小说明两个概率分布越接近。特别的，在强化学习中，AC（Actor-Critic）算法同时输出价值函数和动作概率分布，我们可以将 AC 算法下的小网络损失函数定义为下式。

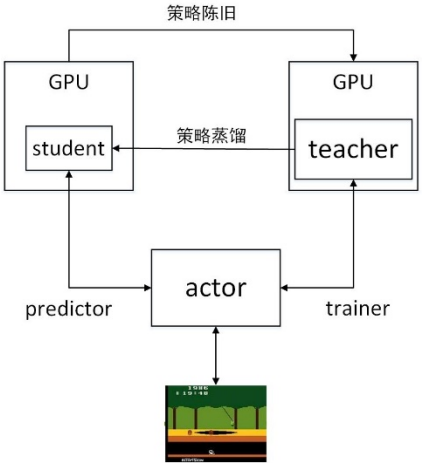

Figure 4 NNC-DRL

$$L_{student}(\theta_{student}) = D_{kl}(\pi(a|x_t;\theta_{teacher})||\pi(a|x_t;\theta_{student}))$$
$$+\frac{1}{2}\left(V(x_t;\theta_{student}) - V(x_t;\theta_{teacher})\right)^2 - \beta H(\pi(a|x_t;\theta_{student}))$$

虽然小网络一直在学习大网络的输出，但并不能保证两个网络的输出完全一致。采样的策

略与训练的策略不一致将会导致收敛性问题（有时会不收敛）。所以这里需要引入重要采样技术，通过重要性权值弥补两个策略差异导致的收敛性问题。重要采样是通过一个概率分布 p 的样本去估计另一个概率分布 q 上的函数 f(x)。通过重要采样，我们将 AC 算法中损失函数定义为下式，其中 ρt 表示重要性权值，即小网络与大网络动作概率比值。

$$L_{teacher}(\theta_{teacher}) = \rho_t log\pi(a|x_t,\theta_{teacher})(r_t + \gamma v_{t+1} - V_\theta(x_t))$$
$$+\frac{1}{2}(V_t - V_\theta(x_t))^2 - \beta H(\pi(a|x_t,\theta_{teacher}))$$

通过优化这些损失函数就能在保证收敛性的前提下加快采样，进而加速整个 DDRL 训练过程。

（3）可行性分析

神经网络压缩与加速技术目前已经很成熟，在保证神经网络预测准确率的情况下达到几十甚至几百倍的压缩，同时得到可观的加速效果。神经网络压缩技术也在强化学习领域体现出效果，比如压缩训练好的 DQN 网络，在将 DQN 网络压缩到原来的 7%时仍然保持原有的策略质量。我们在并行 DRL 框架 GA3C 上实现了该算法，在 GPU 服务器以及 CPU 服务器上均做了可行性验证。我们使用强化学习任务是 Atari 2600 游戏。首先测试了整个训练过程中大小神经网络预测、环境状态改变以及神经网络训练的时间占比。由于小网络预测速度是大网络的两倍，理论上整个训练过程会有 20%左右的加速。在 PPS（Predict Per Second）和 TPS（Training Per Second）指标上，我们的算法大约提速 17%。实验结果表明，神经网络压缩并没有影响最终策略质量，如图 5 所示。其中 NNC-DRL 是基于网络压缩的算法，其他两个分别是直接训练大网络和小网络的效果。

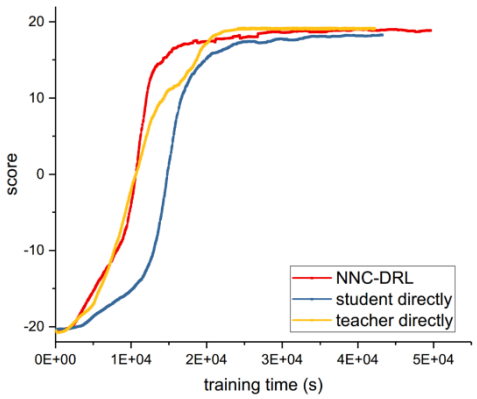

Figure 5 game pong

实验结果中，我们的压缩方法使用更少的时间去训练的目标策略网络，并且没有策略质量损失。其原因是，我们有效的加速了神经网络的预测过程，使得产生样本的时间缩短约一半，而最终的 DRL 训练时间缩短约 20%。同时，我们利用重要性采样技术在一定程度上保证了策略的质量，从实验结果中也可以看出其效果是很明显的。

研究内容二：分阶段加速 DDRL 训练

（1）研究动机

强化学习（弱监督）训练和监督学习训练存在的本质区别是训练样本。监督学习的数据是通过人类专家标记好的，数据质量非常好。而强化学习中，唯一引导学习的信号就是奖励，没有针对每个输入的标签。而且强化学习的任务一般是稀疏奖励的，大部分时间没有奖励信息，样本质量相对差很多。数据特点导致两种学习在训练过程中表现出不同特点。如图 6 所示，分别显示了强化学习（左）和监督学习（右）的收敛过程。

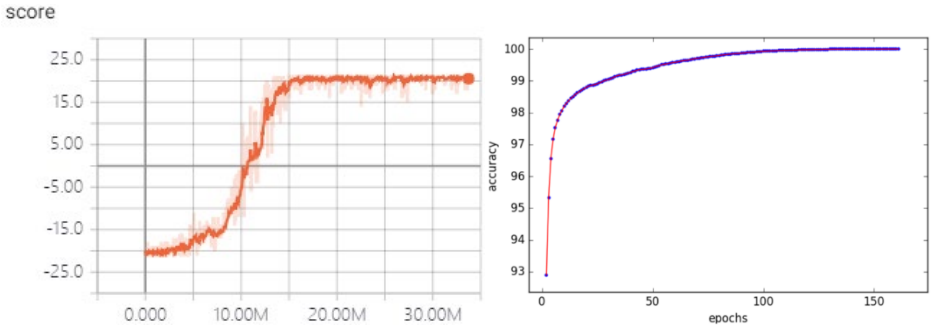

Figure 6 RL 与 SL 收敛曲线

强化学习解决的任务是 Atari 游戏，监督学习解决的任务是手写数字识别。明显的，强化学习训练在前期是缓慢增长的，中期快速增长，后期趋于收敛不再增长。而监督学习，前期是快速增长的阶段，缓慢趋于收敛。强化学习前期是随机策略，随机的在环境中探索，偶尔能碰到有奖励的状态。大部分时间所获得的样本都是没有奖励的，那么对于学习策略也是没有任何帮助的。所以前期强化学习很难提升策略。对于手写数字识别，样本已有正确标签，每个样本中都能学得有意义的知识。所以监督学习前提准确率的提升是非常快的。强化学习训练过程中的收敛特点启发我们可以将其划分为三个阶段，即前期随机探索阶段，中期策略提升阶段以及后期策略稳定阶段。不同的阶段我们根据其特点分别进行分析和优化，加速整个 DDRL 的训练。现有 DRL 训练加速方案都是针对整个训练过程，并没有类似的阶段划分。

（2）技术路线

这里我们的 DDRL 训练框架限制为数据并行式的模式，如图 2 所示。其中参数服务器保存最新的神经网络模型参数，各计算节点 worker 完成神经网络的梯度计算。由于训练数据是与环境交互产生并保存在本地内存，我们将利用本地数据完成该 worker 节点的梯度计算任务。这里我们并没有使用分布式的经验数据存储方案，为了防止数据样本之间相关性对训练神经网络的影响，我们利用参数服务器搜集各节点梯度，这样即可减少样本相关性的影响。在通信模式上，我们针对异步优化方式（异步随机梯度下降），各 worker 节点与参数服务器异步通信。这样的好处是具有更大吞吐量，但会引入梯度陈旧问题，所以我们不能使用太多的计算节点，以减少梯度陈旧程度。在第三个研究内容里，我们会详细讨论如何解决可扩展性问题。

我们的方案是将 DDRL 的训练过程划分为前中后三个阶段，那么第一个需要解决的问题是如何划分阶段。对于已经训练完的任务，我们能直观的将其划分为三个阶段。但新的任务，甚至不知道奖励范围的情况下，我们需要一定的方法去判断当前处于哪个阶段。我们知道强化学习前期是随机探索的，能在环境中获得的奖励一定是维持在某个范围内。我们可以设置合适的阈值，当平均奖励在这个阈值范围内就认为是前期阶段。随着策略的提升，所能获得的平均奖励逐渐增加。直到策略趋于成熟，平均奖励不再增长。以此我们划分出该任务的前中后阶段，并能判断算法当前处于哪个阶段。

在前期随机探索阶段，所能采集的样本大部分都是无奖励的。无奖励的样本其实对强化学习训练是没有价值的，毕竟引导强化学习的信号只有奖励。TD-error 作为许多强化学习优化目标，其形式化表示为下式。

$$TD-error = r_t + \gamma V(S_{t+1}) - V(S_t) \approx 0$$

当即时奖励 rt 为 0 并且价值函数 V 的输出接近，那么整个 TD-error 几乎为 0，那么对于学习价值函数没有帮助。虽然这些样本没有奖励信息，但是仍然可以通过无监督的学习算法训练神经网络。深度神经网络的前面若干层可以看作是在做特征提取，从原始输入提取出抽象特征，然后用于预测。通过无监督学习的方式，能对神经网络的前面若干层参数进行

优化。比如自编码器，无监督目标检测，图像分割等。通过自编码器，我们以原始输入作为目标输出，学习神经网络参数。当然自编码器也有其缺点，就是对任务无感知，对于不同的强化学习任务，神经网络的关注点应该是不一样的。可以通过类似注意力的机制实现对特定目标的感知。

在中期策略提升阶段，智能体能采集的经验数据中有奖励的样本（主要是正奖励）开始逐渐增多。但对于稀疏奖励的任务来说，有奖励的样本仍然占少数。为了提高这个阶段的样本利用率，加快神经网络训练，可以设计类似优先级经验数据池的机制。对价值更大的样本赋予更高的优先级。同样的更新次数，理论上是能达到更快的策略提升。但优先级经验数据池的技术仍然有其缺陷，如相关研究中讨论的，只适用于 off-policy 这类有经验数据池的算法。对于 on-policy 的算法无法直接使用，目前能做的就是将其强行转化为 off-policy，然后通过重要性采样技术弥补策略陈旧问题。

中期的训练过程中，不仅有大量 0 奖励样本，同样还有许多负奖励样本。在强化学习中，负奖励是很容易获得的，比如在走迷宫的时候，很容易走到死胡同而得到负奖励。通过利用这些负奖励数据能有效提升策略。在传统的强化学习学习中，已经包含了对负奖励的利用，比如 AC 算法。在 AC 算法中当奖励为负时，会降低该动作的概率，而增加其他动作的概率，使其避免下次仍得到负奖励。最近出现的一类经验数据池的变种，Hindsight Experience Replay，使得负奖励的样本得到更大的利用。通过将交互序列的最终状态作为目标状态，将负奖励变为正奖励，即可学习到达该状态的策略。但是这类方法的使用范围限制较大，只能用于目标状态与其他状态相似的任务。当目标状态与其他状态完全不相似，也就不能使策略泛化到目标状态，该方法也就失效了。而且对于没有经验数据池的 on-policy 算法，我们仍然需要使用其他技术来利用负奖励样本。

在后期策略稳定阶段，神经网络趋于收敛，策略不再快速提升。在分布式训练的场景下，我们知道每个计算节点 worker 都有局部模型。单个神经网络的策略不再提升，但可以通过集成学习结合多个局部模型的策略，得到质量更高的策略。但无论是 bagging 还是 boosting 的集成算法，都会增大最终策略网络的体积（n 路模型集成得到 n 倍大的神经网络）。这里可以通过知识蒸馏的方式将其压缩到单个模型大小。在知识蒸馏的过程中保持局部模型的差异性（多样性），便可继续做集成压缩。最终使得每个局部模型都有提升。

（3）可行性分析

为了验证前期随机探索阶段无监督学习的可行性，我们在并行框架 A3C 上用自编码器训练神经网络的前若干层。采用的任务还是 Atari 游戏，使用卷积神经网络提取特征。通过卷积反卷积我们可以训练卷积层的参数。前期过后，仍然使用 A3C 算法进行神经网络训练。实验效果如图 7 所示，我们在 1M 步的时候停止自编码器。下方的曲线表示原始 A3C 算法的收敛曲线，上方的曲线是前期用自编码器的收敛曲线。我们看到通过自编码训练前期阶段能带来显著的训练加速效果。

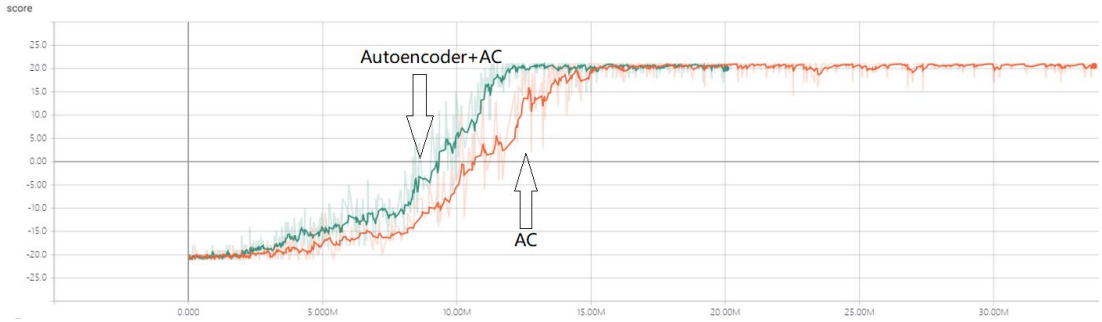

Figure 7 AE+AC 效果

自编码训练前期阶段的策略网络能有效提升整体的训练速度，我们分析其原因，应该

是通过自编码，我们的策略网络有了更好的初始化参数。当策略开始提升，我们的初始位置将提升收敛速度。

为了验证后期策略集成的效果，我们在策略进入稳定阶段之前使用原始 A3C 进行训练，后期采用策略集成和压缩的技术继续训练。对比了继续使用 A3C 进行训练的效果，最终每个局部模型的策略是有明显提升的。

研究内容三：基于策略集成的 DDRL 训练框架

（1）研究动机

典型的分布式训练框架（深度学习领域）包括数据并行和模型并行两种，其中数据并行在实际场景中用得比较多。数据并行是将训练数据拆分成若干份，每份交给一个计算节点使用，每个计算节点保存一份模型副本并计算本地梯度值。所有计算节点的梯度需要上传给中心化的服务器（参数服务器）合并梯度并更新神经网络。DDRL 框架 A3C 以及 DDQL 都是基于数据并行的方式。在研究背景中我们讨论了数据并行的优缺点。在深度强化学习领域，研究人员提出另一种分布式训练框架。将采样（Actor）和训练（Learner）物理上拆分开，放置在不同计算节点上。DDRL 框架 GA3C、ELF、IMPALA、Ape-X 等均是分离式的。这种框架具有多种优势，比如没有所谓的梯度陈旧或者短板问题，因为不涉及到梯度传输问题。同时，分离式的框架能有效处理多任务学习，不同 Actor 可以与不同任务交互，通过 Learner 学习多个任务的策略。但 Actor 和 Learner 分离会导致策略陈旧问题（异步执行），所以其扩展性会受到影响。Actor 和 Learner 传递神经网络参数和样本的代价大，也同样影响到可扩展性。

集成学习在分布式训练中逐渐体现出优势。各个局部模型之间通过知识蒸馏能有效提升模型质量，同时没有像数据并行模式下的梯度问题。集成技术在分布式训练中问题也是很明显的，主要是单个计算节点训练局部模型缓慢。已有的方案中，每个计算节点通过更新 K 步参数训练局部模型，然后相互之间交换模型实现知识分享。为了将集成技术更好的应用于分布式深度强化学习训练，提升可扩展性，我们需要设计新的训练框架。

（2）技术路线

我们将整个计算集群进行分组，如图 8 所示。组内采用典型的 Actor 和 Learner 分离式的 DDRL 训练框架，组间定期交换组内的局部模型并进行策略集成（知识蒸馏）。这样的设计具有多种优势，首先，通过合适的分组，组内节点不会太多，所以策略陈旧问题可以忽略。其次，整个框架中不通信梯度，也就没有所谓的梯度陈旧或者短板问题。再者，为了更好的利用集成学习的优势，我们需要保证各组内局部模型的好而不同，保持多样性，可在各组内使用不同的强化学习算法以及神经网络模型。

具体的数据和算法的划分是这样的，首先，在各组内我们使用了 Actor 和 Learner 分离式的训练框架。Actor 上需要运行模拟环境以及神经网络预测，产生的经验数据通过网络发送至 Learner 节点，在 Learner 上需要运行神经网络训练进程。同时，定期的组间模型交流需要相互发送神经网络参数。Learner 节点上还需要完成各组模型的集成学习。这里的通信瓶颈主要集中在组内的 Actor 和 Learner 之间。通过减少组内节点数量，使得其通信代价减少，达到组内训练的最大效率。

在集成训练时，我们采用知识蒸馏技术。将所有局部模型的策略分布进行混合，并作为局部模型训练中的软目标。这里有很多种集成技术可以使用，我们先利用传统的 bagging 技术对其进行实现和验证，即局部模型策略分布的加权平均。每个局部模型的权值使用价值函数进行计算，直接使用 c=softmax(value|state)表示各局部模型的权值。之所以使用价值函数计算权值，是因为价值函数体现了该局部模型在当前状态下所采用策略能获得的累积奖励大小。价值函数越高，说明该模型采用的策略越好，应该赋予越大的权值。那么集成策略表示为以下混合概率分布。

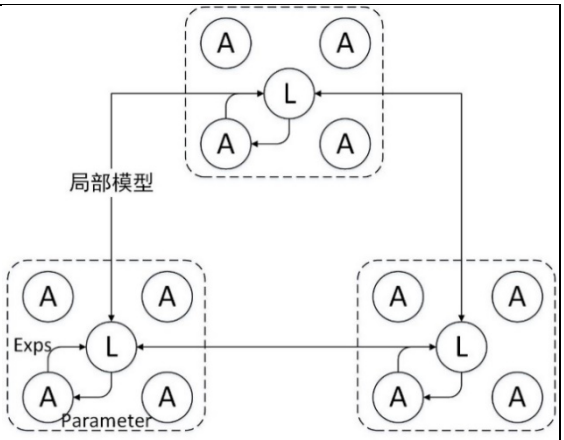

Figure 8 集成策略框架

$$\pi_{ens}(a|s) = \sum_{i=1}^{K} c_i \pi_i(a|s)$$

在策略集成时，各组内的神经网络损失函数定义为下式。

$$Loss(\theta) = loss_{org}(\theta) + \beta D_{kl}(\pi_{ens}(a|s)||\pi(a|s;\theta))$$

式中前半部分是原始强化学习的损失函数，后半部分是需要局部模型的策略尽量趋近于集成策略。

（3）可行性分析

我们实现了策略集成的原型框架，共有 8 组，其中每组 2 个计算节点，用于实现 Actor 和 Learner。Learner 采用 AC 算法，并且各组的神经网络参数都一样，只是初始化不同。为了说明策略集成的有效性，我们对比了组间策略集成以及不集成的情况，如图 9 所示。

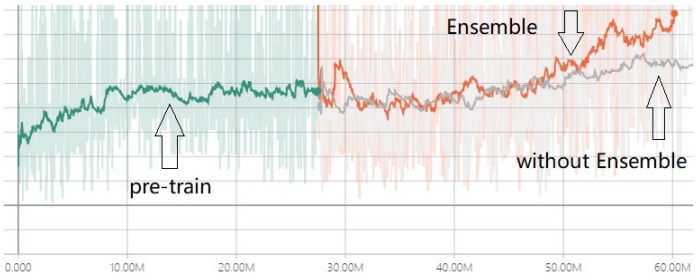

Figure 9 集成实验效果

其中前半部分表示组内局部模型的训练，随着迭代步数得分逐渐增长。上方曲线表示通过策略集成训练，而下方曲线表示不使用策略集成。结果也是比较明显的，通过策略集成确实能有效提高各个局部模型的策略质量。

预期成果

- 基于神经网络压缩技术加速采样，从而加速 DDRL 的训练过程。
- 研究分阶段加速 DDRL 的方法，提高整体训练速度和最终策略质量。
- 提出基于策略集成的 DDRL 训练框架，克服已有框架的缺陷。
- 在国际高水平会议期刊上发表 3 篇学术论文。

四．课题研究的创新之处

研究内容、拟采用的研究方法、技术路线等方面有哪些创新之处。

- 研究内容一创新之处：我们提出一种新的 DDRL 训练模式，通过神经网络压缩加速采样，在保证收敛性的前提下加速整个 DDRL 训练过程。
- 研究内容二创新之处：我们通过将整个 DDRL 训练过程进行分阶段，不同阶段采用不同优化技术，达到加速训练的效果。
- 研究内容三创新之处：我们设计了一种新的基于策略集成的 DDRL 训练框架，具有较好的可扩展性。

五．研究工作进度安排

- 2018 年 12 月～2019 年 2 月：完成研究内容一的论文并投稿 IJCNN 2019。
- 2019 年 3 月～6 月：实现研究内容二的算法，做实验对比已有工作；把该算法的内容撰写成论文，投至国际期刊。
- 2019 年 7 月～10 月：实现研究内容三的方法，做实验验证方法效果；把该方法的内容撰写成论文，投至国际期刊。
- 2019 年 11 月～2020 年 4 月：完成博士毕业论文。

研究生本人签名：_____

年　　月　　日