论文《Learning to Teach and Follow in Repeated Games》阅读报告

张国良

（智能软件与工程学院，智能软件工程，23 级，231880038）

摘　要：阅读的论文核心目标是解决智能体在 2 人矩阵重复博弈（如囚徒困境、小鸡游戏）中的长期收益最大化问题。论文指出，现有重复博弈学习算法分为 “追随者算法”（被动适应对手策略，无法主动引导合作）和 “教导者算法”（主动引导对手，但难以接受新合作方案且惩罚成本高），两类算法单独使用均存在缺陷。为此，论文提出 SPaM 算法，通过融合 “教导者模块” 与 “追随者模块”，结合 “目标解计算”“愧疚值惩罚”“双效用函数” 等机制，实现 “既主动引导合作，又灵活适应对手合理策略” 的功能。实验结果表明，SPaM 在囚徒困境、小鸡游戏、Tricky 游戏中，与追随者算法（FP）、教导者相关算法（WoLF-PHC）及自身配对时，长期平均收益均显著更高，且初步验证了其与人类交互的有效性。

关键词：重复博弈；SPaM 算法；追随者算法；教导者算法；智能体协作学习

1 引言

在多智能体交互场景中，“重复博弈” 是典型任务之一 —— 智能体需通过多次博弈调整策略，最终实现长期收益最大化。这类博弈的核心挑战在于 “合作与收益的平衡”：例如在经典的 “囚徒困境” 中，智能体若仅追求短期收益选择 “背叛”，长期会陷入 “双方均背叛、收益极低” 的困境；若能建立合作，可实现 “双方均合作、收益最优”，但需解决 “如何引导对手合作” 与 “如何应对对手背叛” 的问题。

现有研究中，重复博弈的学习算法主要分为两类，但均存在明显局限性：一

是 “追随者算法”（如虚拟博弈 FP、Q - 学习），其逻辑是通过统计对手历史动作推断策略，选择自身短期最优响应，虽能适应稳定策略，但无法主动提出合作方案，易错失共赢机会；二是 “教导者算法”（如以牙还牙 TFT、Godfather++），其通过 “合作奖励、背叛惩罚” 引导对手，但无法接受对手的新合作方案，且惩罚常过度（如长期惩罚导致自身亏损），对不学习的静态对手无效。

为弥补两类算法的缺陷，论文提出 SPaM 算法，创新性地将 “教导” 与 “追随” 功能集成，让智能体既能主动塑造对手的合作行为，又能灵活适应对手的合理策略，同时通过精准的惩罚机制降低自身损失。该研究对多智能体协作（如机器人协作、分布式决策）具有重要意义，为解决 “社交困境类博弈” 提供了新的算法思路。

2 SPaM 方法

SPaM 算法的核心框架是 “双模块协同 + 精准调控机制”，通过明确 “合作目标”、“奖惩标准” 与 “动作选择逻辑”，实现 “教” 与 “跟” 的平衡。以下从研究设计、核心组件与执行流程三方面详细说明

2.1 核心设计理念

SPaM 的设计围绕三个关键目标：1）确定 “双方长期共赢的合作方案”（目标解）；2）通过 “愧疚值” 量化对手的背叛程度，实现 “惩罚不过度、自身损失最小”；3）通过 “双效用函数”（教导者效用、追随者效用）兼顾 “引导合作” 与 “保障自身收益”，避免单一模块的局限性。

2.2 核心组件与功能

1. 目标解（c）：“双方长期最赚的行动方案”

首先，SPaM 会先看清楚游戏规则（比如囚徒困境的收益表），计算一个 “目标解” —— 这是一套 “让双方长期收益最大化的联合动作序列”（通俗说就是 “谁都不会亏，还能一直赚” 的策略）。

比如在囚徒困境里，目标解就是 “双方每一轮都选合作” —— 因为这个组合下，双方每轮都得分，长期下来比 “都背叛” 或 “一合作一背叛” 赚得多。

2. 两个效用函数：“判断动作好不好的双重标准”

SPaM 用两个 “评分表” 来评估每个动作（合作 / 背叛）的好坏，分别对应 “教导” 和 “追随” 需求：

- **教导者效用（T）**：判断这个动作能不能 “引导对手走向目标解”
  - 比如对手没犯错（没偏离目标解）：SPaM 选 “合作”（符合目标解），T=1（好评）；选 “背叛”（偏离），T=-1（差评）。
  - 比如对手犯错了（偏离目标解）：SPaM 选 “能惩罚对手且自己亏

最少” 的动作，T = 正数（好评）；选 “惩罚不到位或自己太亏” 的动作，T = 负数（差评）。

- **追随者效用（F）**：判断这个动作 “当下能让自己拿多少分”（纯算自身收益，和追随者算法的逻辑一样）。

3. 愧疚值（G）：“对手犯错的‘债’，决定要不要惩罚、罚多久”

SPaM 会给对手算一个 “愧疚值”—— 相当于 “对手偏离目标解后，欠的‘惩罚债’”。愧疚值的高低决定了 SPaM 要不要惩罚、惩罚多狠，如果对手没有犯错就愧疚值清零或者不变；如果对手犯错了，此时如果 SPaM 没犯错就增加愧疚值，否则可能减少愧疚值；惩罚时候根据愧疚值来，直到愧疚值清零。

4. 动作选择规则：“大部分时候‘教’‘跟’兼顾，偶尔灵活探索”

SPaM 选动作时，不会死板地只按教导或追随逻辑，而是按一定的概率规则来，平衡 “引导合作” 和 “避免亏分”。

大概率是先从 “教导者效用合格（T≥0）” 的动作里，选 “追随者效用最高（F 最大）” 的动作；小概率不管教导者效用合不合格，直接选 “全局追随者效用最高” 的动作；极小概率随机选动作 —— 避免太死板，偶尔探索新可能。

2.3 完整执行流程

SPaM 的每轮博弈执行流程可分为 “初始化 - 动作选择 - 观测更新” 三步：

1. **初始化阶段**：输入博弈矩阵（如囚徒困境的收益表），计算目标解 c；初始化参数（双方愧疚值 G=0，教导者效用 T=0，追随者效用 F=0）；
2. **动作选择阶段**：根据当前对手的愧疚值（G_对手）判断场景：若 G_对手 = 0（对手无背叛），仅选择 “符合目标解的动作”（T≥0）；若 G_对手 > 0（对手有背叛），选择 “有效惩罚且自身收益最高的动作”（T≥0 且 F 最大），按一定概率规则确定最终动作；
3. **观测与更新阶段**：观测双方动作与收益，更新愧疚值（按对手是否回归合作调整 G）、教导者效用（按动作是否符合目标解 / 惩罚效果更新 T）、追随者效用（按当前收益统计更新 F）；
4. **迭代执行**：重复 “动作选择 - 观测更新” 步骤，直至博弈结束

3 复现实验、展示性结果

3.1 实验设置

```
(base) askiki@Ubuntu:~/learnFile/light of science$ tree .
.
├── FP_Agent.py
├── game.py
├── main.py
├── SPaM_Agent.py
├── train.py
└── WoLF_PHC_Agent.py
```

FP_Agent,SPaM_Agent,WoLF_PHC_Agent 分别实现了三种策略，即追随者算法，SPaM 算法，和教导者算法

train.py 实现了模型的训练函数，包含两个智能体博弈的函数

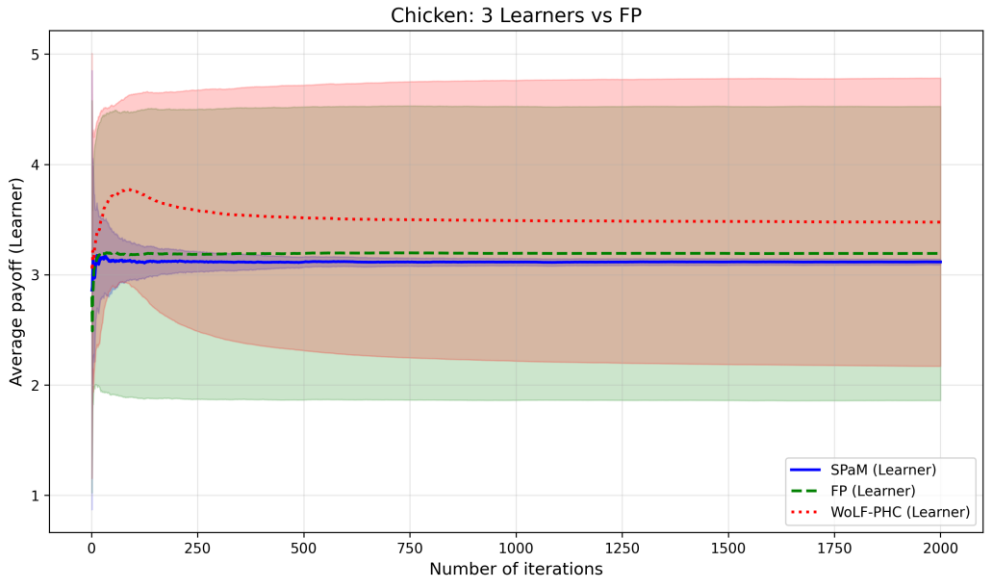
game.py 实现了博弈过程中可能用到的公用函数（动作选择）以及囚徒困境，小鸡游戏和 Tricky 游戏的游戏规则

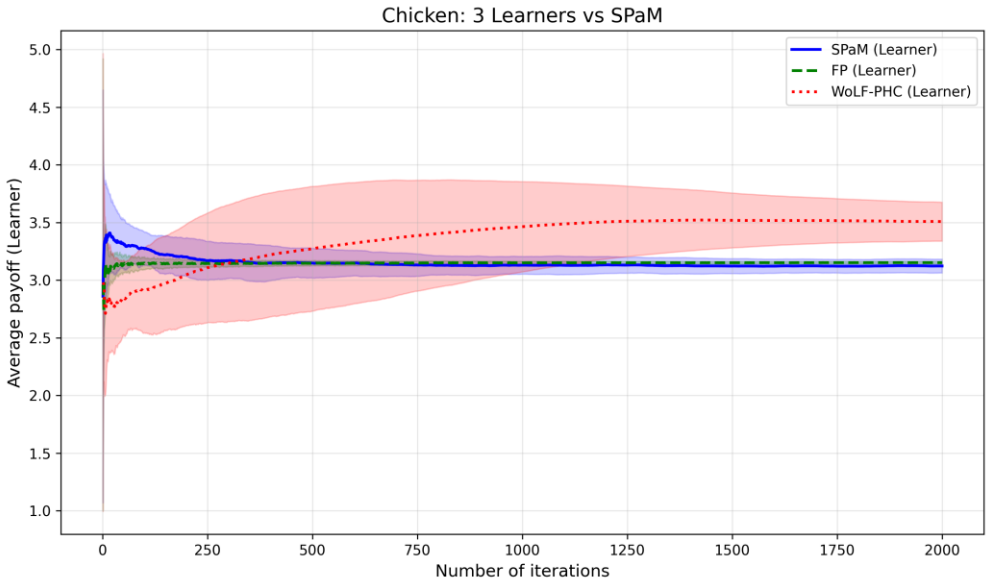
main.py 则实现了不同算法的博弈和性能对比，参数依据论文原文

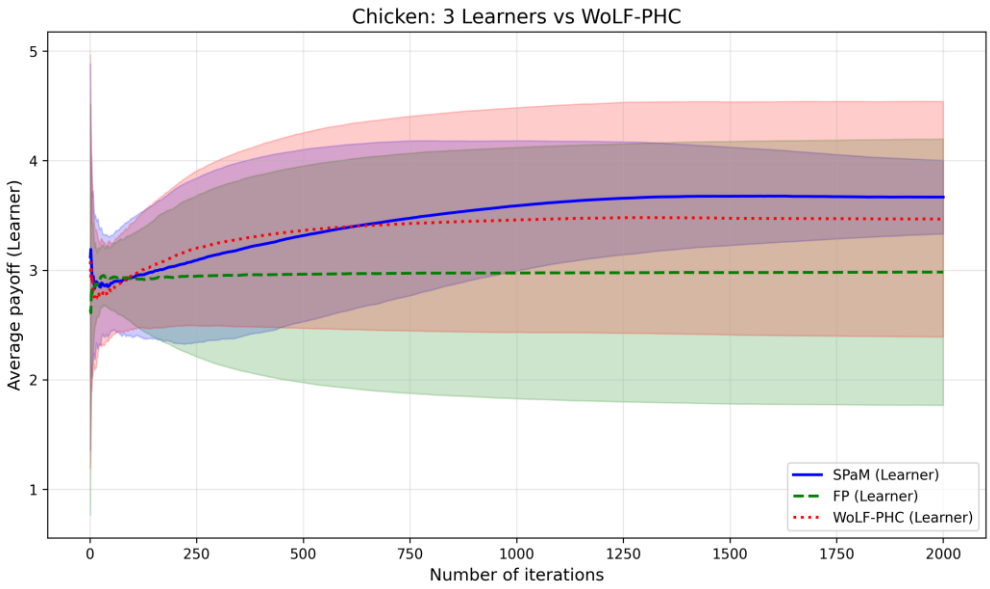
项目地址见：

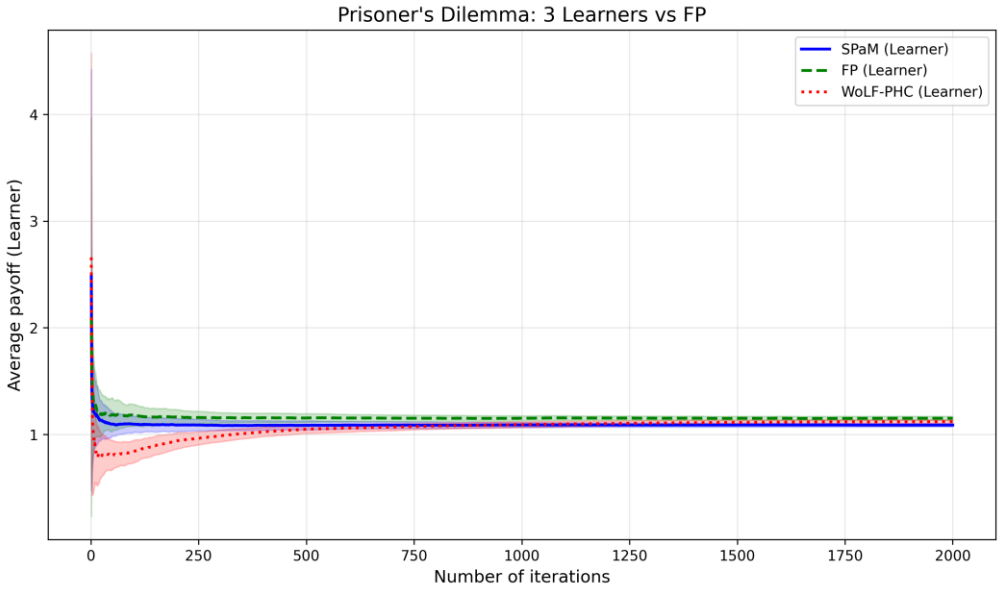
[askiki12/Learning-to-Teach-and-Follow-in-Repeated-Games-remake]

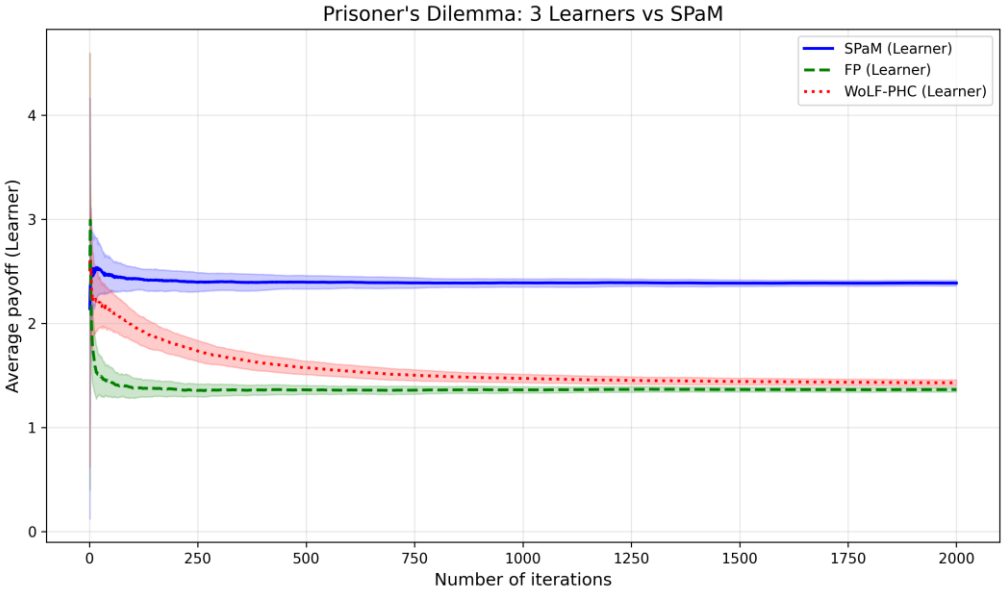
3.2 性能

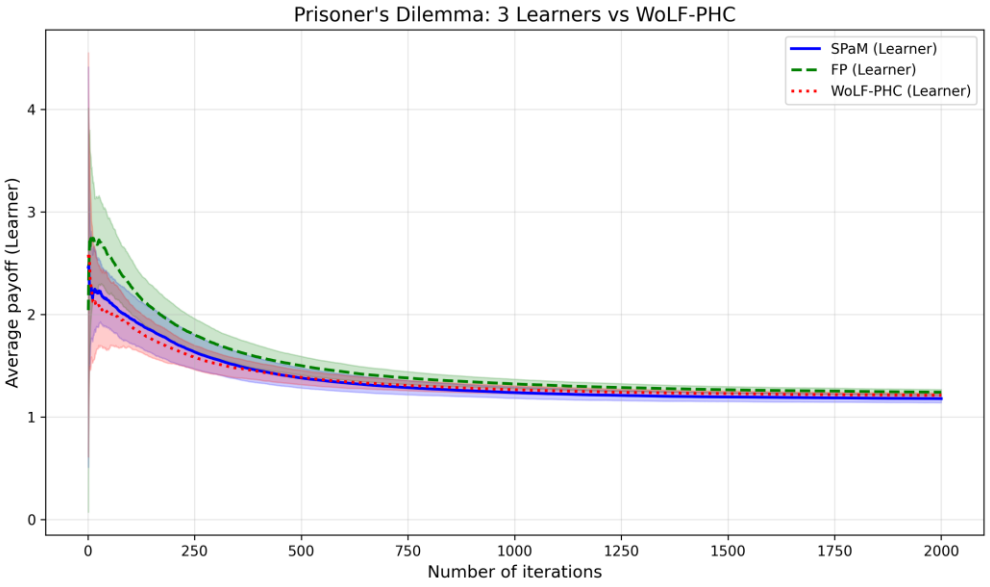


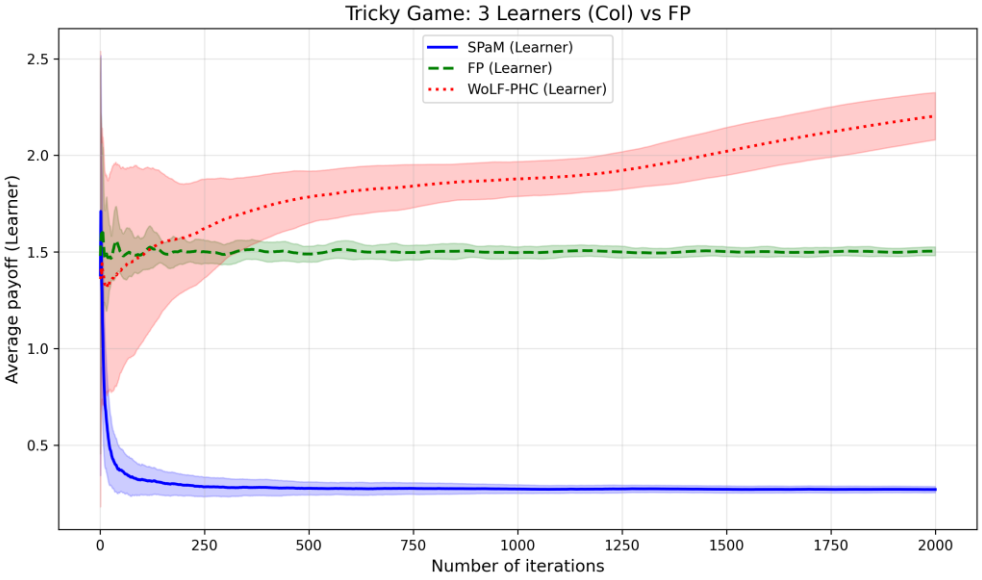

Chicken: 3 Learners vs SPaM
SPaM (Learner)
FP (Learner)
WoLF-PHC (Learner)
Average payoff (Learner)
Number of iterations

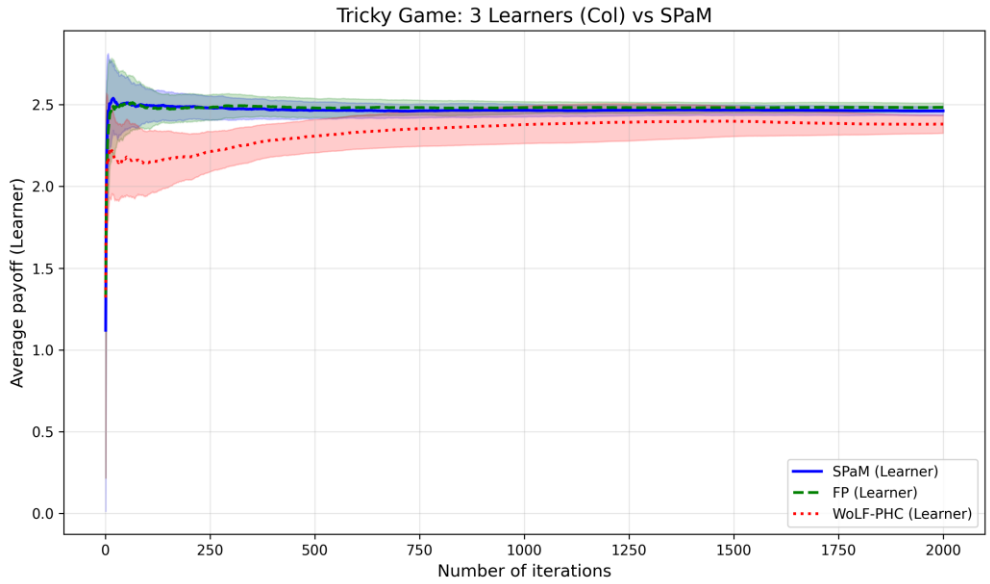

Chicken: 3 Learners vs WoLF-PHC
SPaM (Learner)
FP (Learner)
WoLF-PHC (Learner)
Average payoff (Learner)
Number of iterations

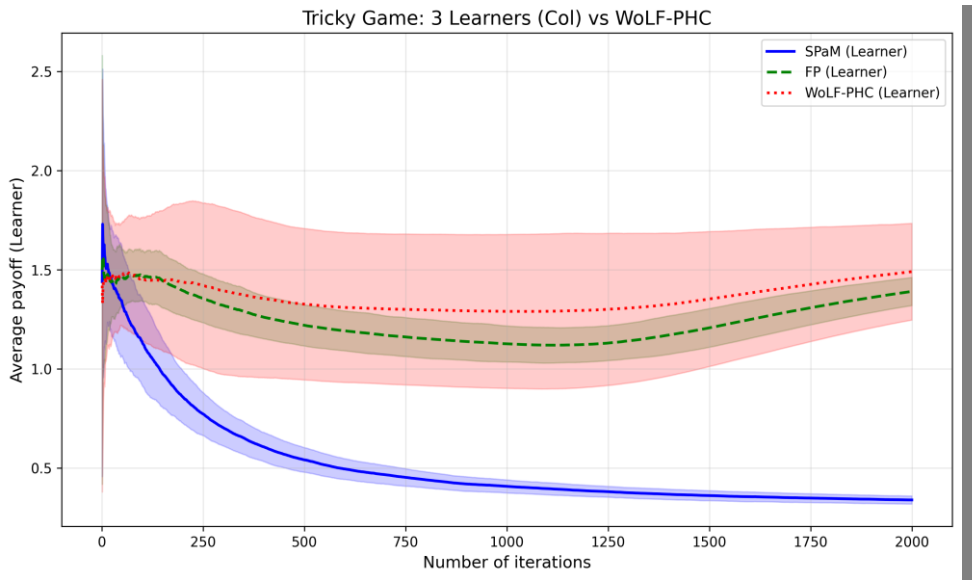

Prisoner's Dilemma: 3 Learners vs FP
SPaM (Learner)
FP (Learner)
WoLF-PHC (Learner)
Average payoff (Learner)
Number of iterations

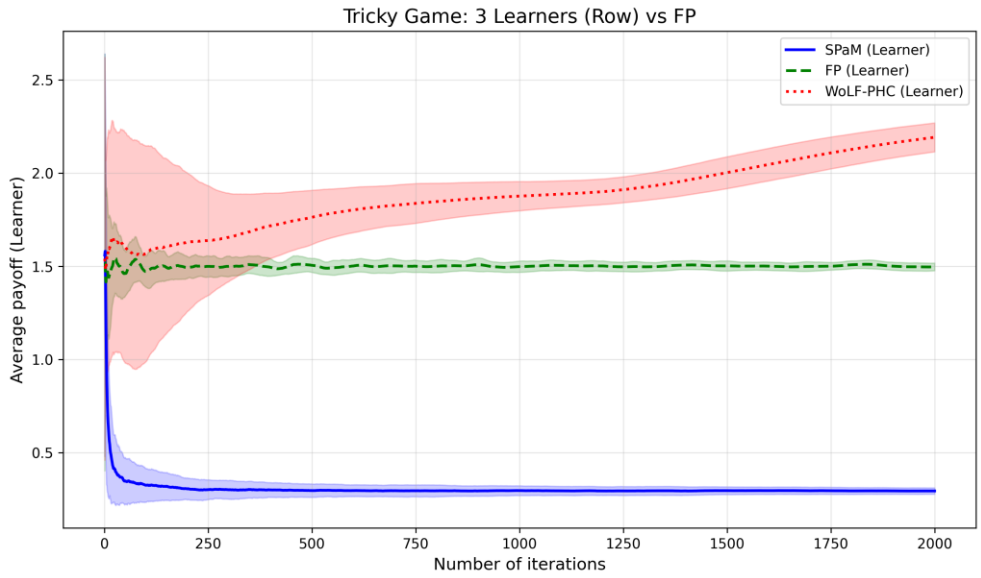

Prisoner's Dilemma: 3 Learners vs SPaM
SPaM (Learner)
FP (Learner)
WoLF-PHC (Learner)
Average payoff (Learner)
Number of iterations
0
1
2
3
4
0
250
500
750
1000
1250
1500
1750
2000

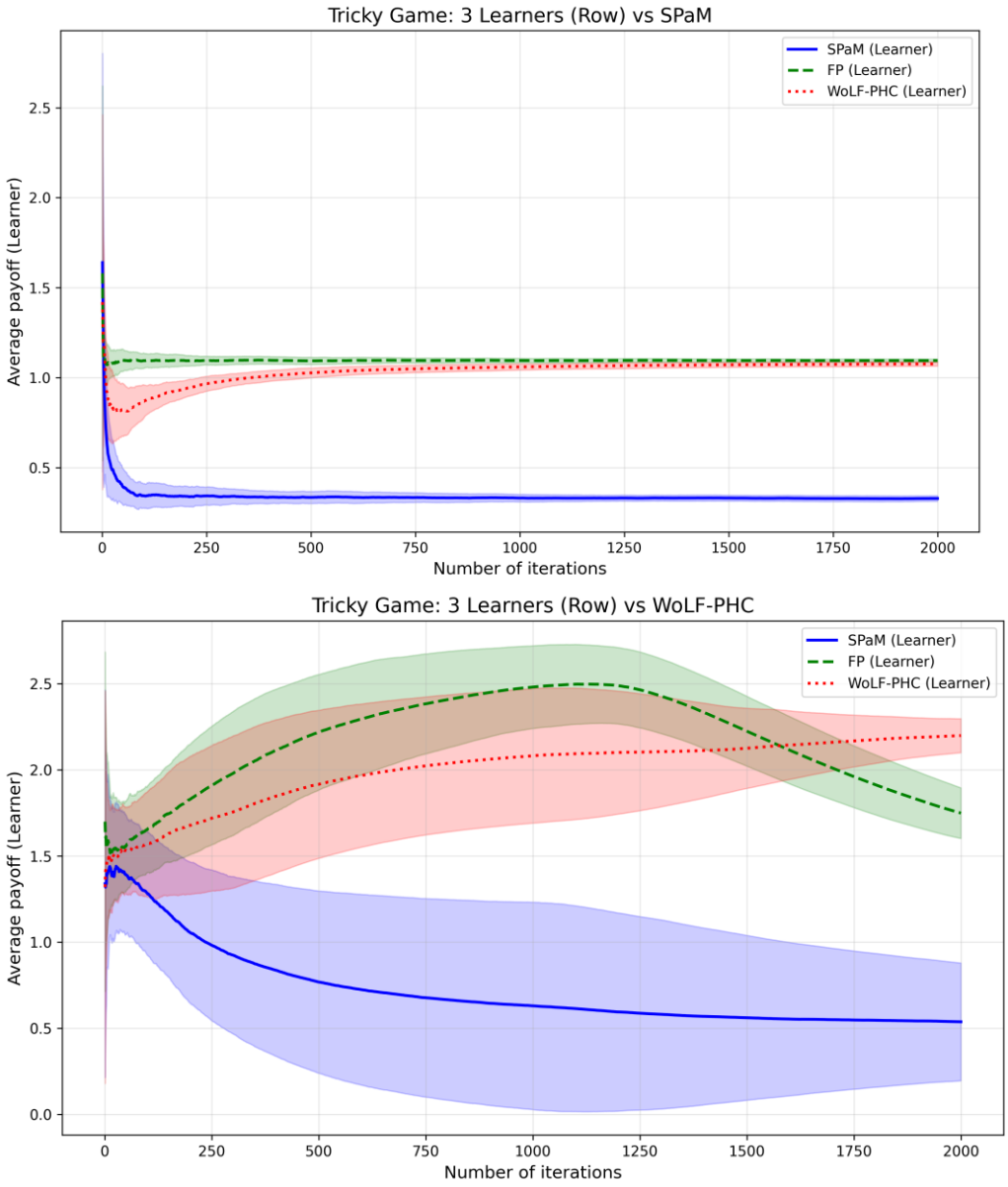

Prisoner's Dilemma: 3 Learners vs WoLF-PHC
SPaM (Learner)
FP (Learner)
WoLF-PHC (Learner)
Average payoff (Learner)
Number of iterations
0
1
2
3
4
0
250
500
750
1000
1250
1500
1750
2000


Tricky Game: 3 Learners (Col) vs FP
SPaM (Learner)
FP (Learner)
WoLF-PHC (Learner)
Average payoff (Learner)
Number of iterations
0.5
1.0
1.5
2.0
2.5
0
250
500
750
1000
1250
1500
1750
2000


Tricky Game: 3 Learners (Col) vs SPaM
Average payoff (Learner)
Number of iterations
SPaM (Learner)
FP (Learner)
WoLF-PHC (Learner)


Tricky Game: 3 Learners (Col) vs WoLF-PHC
Average payoff (Learner)
Number of iterations
SPaM (Learner)
FP (Learner)
WoLF-PHC (Learner)


Tricky Game: 3 Learners (Row) vs FP
Average payoff (Learner)
Number of iterations
SPaM (Learner)
FP (Learner)
WoLF-PHC (Learner)



4 总结和展望

该论文针对 2 人矩阵重复博弈中 “追随者算法被动、教导者算法惩罚过度” 的缺陷，提出 SPaM 算法。其创新融合 “教导者 - 追随者” 双模块，通过目标解计算锚定合作基准，愧疚值量化对手偏离程度，双效用函数平衡引导与收益，实现 “既主动塑合作，又灵活控损失”。实验在囚徒困境、小鸡游戏等场景验证，SPaM 与其他算法相比，有时候性能较优，有时候出现了偏移的情况，性能是否真的够好，我个人存疑。

未来可向多智能体场景扩展，设计多对手愧疚值网络；探索不完全信息博弈适配，结合强化学习估算对手收益；还可优化人类交互策略，提升算法在真实社

交困境中的实用性。

参考文献(参考格式)

[1] **Learning to Teach and Follow in Repeated Games**