**种子杯报告**

1. 方案选择

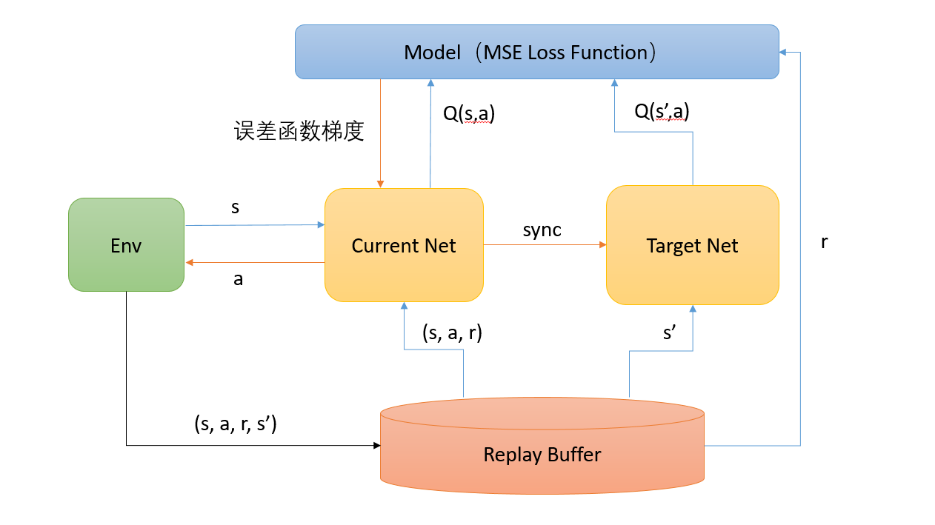
游戏算法的选择我们主要选择了两种方案。

首先，在看到游戏赛题后，我们首先想到了强化学习算法，并使用了适合离散动作的d3qn算法，并进行改进引入rainbow dqn算法，同时不断训练。

尝试过强化学习算法后，我们发现算法效果不佳，后选择另一种算法，使得占有地块，击杀数和取得buff达到平衡，取得了较好得效果。

1. 强化学习算法
2. 算法介绍

我们选择了适用于离散动作空间的d3qn算法，算法中主要是Qnet，Memory,Agent。



图一dqn算法图

Agent模块下有act动作和learn学习两个小模块，通过act不断做出动作，计算该状态（state）做某动作（action）得到的奖励（reward）,并存到replaybuffer（memory缓存区），然后不断学习，从replaybuffer抽出（s,a,r,s’）,让current net和target net计算预测q值和目标q值，再计算MSE Loss,不断优化，使agent能做出最佳决策。

同时，我们还写了actionrespinfo,actionreq,config几个模块，使之不断能向服务器发送request，并解析收到的response。

1. 问题

在训练过程中，我们遇到了击杀数，占格子，取buff三者之间无法达到平衡的难题，同时，人物喜欢待在自己的区域，喜欢在墙角，等问题，我们尝试加大惩罚，扣除reward,但经过多天训练的模型，最终得到的效果充满了随机性，难以继续提高分数。

1. 最终算法
2. 对于“对点”追踪的算法（追踪敌人、追踪buff）：

**对敌人：**选择在自己周围一圈的格子内离敌人最近的那个，前进一步并攻击。

**对buff：**选择离自己最近的buff，运用对敌人的同等方法进行追踪。得到加速和回血buff

1. 对于“对面积”追踪的算法（占不属于自己的地）：

选择离自己最近的“他块”，且过滤掉离自己的欧氏距离小于2的（避免左右横跳），运用对敌人的同等方法进行追踪。

1. 最终算法：
2. 开局立即占取凹槽部分的地（因为这些地方在之后的对局中很难到达，先取即得）。
3. 立即开启以追踪为主的自动策略：
   1. 如果敌人还活着：追踪敌人并释放武器击杀。
   2. 如果敌人死亡或者在无敌状态：
      1. 如果自己hp小于等于70：
         1. 如果地图上有hp buff：追踪hp buff。
         2. 如果地图上没有hp buff：占不属于自己的地。
      2. 如果自己hp大于70：
         1. 如果地图上有speed buff且当前的移动CD高于2：追踪speed buff。
         2. 如果地图上有hp buff且自己的hp小于100：追踪hp buff。
         3. 如果上面二者都不满足：占不属于自己的地。
4. 算法策略伪代码
5. TAKE-ACTION(PRO, CON):
6. # PRO表示己方选手， CON表示敌方选手
7. # PRO:list, CON:list
9. # 将己方选手同距离近的对手对应起来，方便追击
10. **if** len(PRO) + len(CON) >= 3:
11. **if** len(PRO) == 1:
12. **if** Distance(PRO[0], CON[0]) > Distance(PRO[0], CON[1]):
13. Switch(CON[0], CON[1])
14. **else**:
15. **if** Distance(PRO[0], CON[0]) > Distance(PRO[1], CON[0]):
16. Switch(PRO[0], PRO[1])
18. **for** p **in**  PRO:
19. # hp <= 60 时做的策略
20. # 追击目标优先级： hp\_buff > speed\_buff > vacant\_space > competitor
21. **if** p.hp <= 60:
22. **if** there **is** hp\_buff:
23. let p chase **for** hp\_buff
24. **else** **if** there **is** speed\_buff:
25. let p chase **for** speed\_buff
26. **else** **if** there **is** vacant\_space:
27. let p chase **for** vacant\_space
28. **else** **if** there **is** competitor:
29. let p chase **for** competitor
30. **else**
31. let p take a random step

34. # hp > 60 时做的策略
35. # 追击目标优先级： competitor > speed\_buff > hp\_buff > vacant\_space
36. **else** **if** p.hp > 60:
37. **if** there **is** competitor:
38. let p chase **for** competitor
39. **else** **if** there **is** speed\_buff:
40. let p chase **for** speed\_buff
41. **else** **if** there **is** hp\_buff **and** p.hp != 100:
42. let p chase **for** hp\_buff
43. **else** **if** there **is** vacant\_space:
44. let p chase **for** vacant\_space
45. **else**
46. let p take a random step
48. # 防止两个队友重位
49. **if** len(PRO) == 2 **and** Distance(PRO[0], PRO[1]) **is** too short:
50. let PRO[1] get away **from** PRO[0]