建议修改的代码：

我们建议学生们可以从以下四个方向进行代码的修改，用以测试不同的强化学习效果：

1. Reward

`app/sgame_1v1/env/feature_process/config.json`里定义了reward的权重，同学们可以通过修改各个reward的权重去训练出玩法风格截然不同的agent，比如提高击杀奖励，agent会变得更好战，提高金钱奖励agent会提高刷钱效率。同学们可以通过消融实验的方法，去测试每个reward对agent训练的影响。但是不同reward之间不是相互独立的，同时修改多个reward的效果不能简单的用单个reward的实验结果进行简单的线性叠加。如何能达到一个相对平衡的点并取得游戏的胜利，是同学们可以思考的方向。

我们并不推荐同学们去修改reward的生成逻辑和计算方法。

2. Hyperparameters

`app/sgame_1v1/common/configs/config.py`和`conf/configue.ini`里包含了很多参数的配置，同学们需要首先学习并了解每个参数的含义以及对训练/推演的影响，判断出哪些是可以去优化的，再利用实验去验证。

超参数的调整一直被认为是深度学习中的"玄学"，同样的算法，同样的参数，针对不同的应用场景都可能会有较大的表现差异。

3. Graph

`app/sgame_1v1/common/models/model.py`里有对Graph的定义，具体参考`_build_infer_graph`和`_inference`函数，同学们可以在这里进行修改graph的操作；`app/sgame_1v1/actor_learner/game_controller.py`里有对Graph的引用。

4. Model

`app/sgame_1v1/common/models/model.py`里定义了model，包括loss的计算和inference函数。同学们如果想修改PPO算法，可以从这里入手，但我们只建议进行简单的微调，比如optimizer的调整，或添加小的trick。由于当前框架是针对PPO算法搭建的，所以同学们如果想要尝试PPO以外的算法难度会较大，比如目前不支持sample过程的调整， 涉及到exploration上优化的算法或者value-based的算法都很难实施。

Algorithm负责强化学习算法，监控数据主要反映了几个重要的算法指标。

`loss`：损失函数计算出的损失值，包含policy_loss，entropy_loss, 和value_loss，随时间推移，loss应该呈下降趋势。

`all_loss_value`: 上述三个loss的加权求和

`reward`：奖励值，随时间推移，reward应该呈上升趋势。

`policy_log_p_mean`：由以下公式计算得到

$$\nabla_\theta log P(\tau|\theta)$$

其中θ是policy函数的参数，τ是Trajectory。这个指标可以类似policy的熵，如果是随机的policy，各个动作的概率分布是平均的，该值小；反之随着训练的推移，部分动作的概率会增大，概率分布不再平均，该值增大。

Battle

Battle负责的是对战，采用self-play的模式，监控数据主要反映了对战过程的评估指标。

`win`：对局胜率

`frame`：对局帧数

`kda`：对局中的击杀-死亡-助攻比

`hurt`：包含hurt_by_hero和hurt_to_hero两个数据，分别指受到敌方英雄的伤害和对敌方英雄造成的伤害

`money`：游戏中的金钱