5G 網路切片強化學習系統改善建議

概述

本文件整理了針對 5G 網路切片管理知識蒸餾（Knowledge Distillation）+ SAC 強化學習系統的三項關鍵改善建議，涵蓋代碼架構、可用性與物理模擬準確度。

---

1. 關鍵修復：分離訓練與評估環境

問題描述

在 `train_distillation.py` 中，目前使用同一個環境實例（`env`）同時供以下兩者使用：

- **訓練**：傳遞給 `DistilledSAC`
- **評估**：傳遞給 `EvalCallback`

具體影響

當 `EvalCallback` 定期重置環境以執行測試 episode 時，會發生以下問題：

1. **中斷訓練過程**：強制終止訓練代理正在進行的 episode
2. **損壞監控日誌**：`Monitor` 記錄錯誤的 episode 長度和獎勵值
3. **干擾學習算法**：代理接收混亂的數據，無法正確學習，影響模型收斂速度

解決方案

建立**兩個獨立的環境實例**：

方案示例

import gym

訓練環境

train_env = gym.make('your_env_id')

評估環境（獨立實例）

eval_env = gym.make('your_env_id')

傳遞給各自的元件

model = DistilledSAC(env=train_env, ...)
eval_callback = EvalCallback(eval_env=eval_env, ...)

預期效果

- 訓練日誌準確反映代理的真實學習進度
- 評估結果不受訓練干擾，更具參考價值
- 模型收斂穩定性提升

標準實踐：這是強化學習框架（如 Stable Baselines3）的推薦做法

---

2. 可用性改進：命令列參數支持

問題描述

在 evaluate.py 中，EXPERIMENT_DIR 是硬編碼（hardcoded）的常數。每次評估不同時間戳記的實驗資料夾時，都必須手動修改程式碼並重新保存。

現狀痛點

目前方式：每次都要改程式碼

EXPERIMENT_DIR = "./experiments/20260207_1630/" # ← 需手動修改
python evaluate.py

建議改進

使用 Python 標準庫的 **argparse** 模組，將目錄改為命令列參數：

import argparse

parser = argparse.ArgumentParser(
description="評估 5G 網路切片 SAC 模型"
)
parser.add_argument(
"experiment_dir",
help="實驗資料夾路徑（含時間戳記）",
type=str
)
parser.add_argument(
"--num_episodes",
help="評估 episode 數量",
type=int,
default=10
)
args = parser.parse_args()

EXPERIMENT_DIR = args.experiment_dir
NUM_EPISODES = args.num_episodes

使用範例

直接指定路徑，無需修改程式碼

python evaluate.py ./experiments/20260207_1630/

指定自訂 episode 數量

python evaluate.py ./experiments/20260207_1700/ --num_episodes 20

支援相對/絕對路徑

python evaluate.py /path/to/experiment_20260207_1800/

實際好處

方面	改善效果
開發效率	無需重複編輯和保存程式碼
自動化	可輕鬆集成至 bash 腳本或 Makefile
可追蹤性	命令歷史清楚記錄評估的實驗版本
專業度	符合 Python CLI 最佳實務
協作	其他人可直接運行，無需代碼理解

---

3. 物理模擬真實性提升

本節針對 base_station.py 和 buffers.py 提出兩項增強建議，以更貼近真實 5G NR 標準。

3.1 封包分段（Packet Segmentation）

現狀問題

buffers.py 中的註解提及策略：「Whole Packet Transmission」（完整封包傳輸）。若封包無法完整放入剩餘資源塊（RB），該邏輯會直接跳過，導致**資源浪費**。

真實 5G NR 標準

在真實 5G NR 中，**RLC Segmentation** 是標準機制，允許將大型 eMBB 封包分段傳輸，即使只剩小部分 RB 可用。

建議改進

在 `SliceBuffer.remove_packets()` 方法中實現**封包分片邏輯**：

def remove_packets(self, available_rbs, slice_type):
"""
移除緩衝區中的封包並進行傳輸

```
Args:
    available_rbs: 可用的資源塊數量
    slice_type: 切片類型（'eMBB', 'URLLC', 'mIoT'）

Returns:
    transmitted_bits: 實際傳輸位元數
"""

if not self.packets:
    return 0

transmitted_bits = 0
available_bits = available_rbs * self.bits_per_rb

while self.packets and available_bits > 0:
    current_packet = self.packets[0]

    if len(current_packet) <= available_bits:
        # 完整封包可傳輸
        transmitted_bits += len(current_packet)
        available_bits -= len(current_packet)
        self.packets.pop(0)
    else:
        # 分段傳輸（RLC Segmentation）
        transmitted_bits += available_bits
        # 保留未傳輸部分
        self.packets[0] = current_packet[available_bits:]
        available_bits = 0
```

```
return transmitted_bits
```

預期效果

- **頻譜效率提升**：充分利用剩餘資源
- **更真實的模擬**：貼近 3GPP 5G NR 標準
- **RL 代理決策質量提升**：更合理的切片資源分配

3.2 URLLC 效率懲罰（Efficiency Penalty）

現狀問題

在 base_station.py 中，URLLC 和 eMBB 使用**相同的效率計算**。然而真實 URLLC 需要更穩健的 MCS（調變編碼方案）來達成極高可靠性（99.999%），代價是頻譜效率降低。

真實 5G 標準

URLLC 性能需求：

- **可靠性**：99.999%（五個九）
- **時延**：1 ms 以下
- **代價**：須採用低碼率 MCS → 低頻譜效率

相比之下，eMBB 追求高吞吐量，採用高階 MCS。

建議改進

在 constants.py 新增效率因子：

constants.py

URLLC 頻譜效率懲罰因子

模擬 99.999% 可靠性所需的較低 MCS 碼率

URLLC_SPECTRAL_EFFICIENCY_FACTOR = 0.85 # 降低 15%

其他相關常數

EMBB_SPECTRAL_EFFICIENCY = 5.0 # bits/Hz（例值）
URLLC_SPECTRAL_EFFICIENCY = 5.0 * URLLC_SPECTRAL_EFFICIENCY_FACTOR

結果：4.25 bits/Hz

在 base_station.py 中應用此因子：

def calculate_capacity(self, available_rbs, slice_type):
"""
計算切片的傳輸容量
"""
bandwidth_per_rb = 180e3 # Hz
total_bandwidth = available_rbs * bandwidth_per_rb

```
if slice_type == "eMBB":
    spectral_efficiency = EMBB_SPECTRAL_EFFICIENCY
elif slice_type == "URLLC":
    # 應用 URLLC 效率懲罰
    spectral_efficiency = (
        EMBB_SPECTRAL_EFFICIENCY *
        URLLC_SPECTRAL_EFFICIENCY_FACTOR
    )
else:  # mIoT
    spectral_efficiency = MIOT_SPECTRAL_EFFICIENCY

capacity = total_bandwidth * spectral_efficiency
return capacity
```

預期效果

指標	改善效果
模擬真實性	URLLC 資源消耗更符合實際網路
RL 訓練質量	代理學習更合理的 URLLC-eMBB 資源權衡
論文說服力	基於 3GPP 標準的模擬參數
可遷移性	結果更可能推廣到實際部署

優先級建議

序號	項目	優先級	原因
1	分離訓練/評估環境	**高**	影響代碼正確性，可能導致錯誤結果
2	命令列參數支持	**中**	提升開發效率，便於實驗管理
3	物理模擬提升	**中**	提升論文說服力，改善 RL 訓練質量

實施檢查清單

- [ ] 為訓練和評估創建獨立環境實例
- [ ] 驗證訓練和評估日誌分離無干擾
- [ ] 在 `evaluate.py` 實現 `argparse` 支持
- [ ] 測試命令列參數功能
- [ ] 在 `constants.py` 新增 `URLLC_SPECTRAL_EFFICIENCY_FACTOR`
- [ ] 在 `buffers.py` 實現封包分段邏輯
- [ ] 在 `base_station.py` 應用 URLLC 效率懲罰
- [ ] 對比修改前後的評估結果
- [ ] 更新代碼文檔和註解

參考資源

相關技術文件

- Stable Baselines3 官方文檔：環境管理與回調函數
- 3GPP TS 38.322：5G NR RLC 協議規範
- 3GPP TS 38.300：5G NR 整體描述

相關主題

- 5G RLC Segmentation 機制
- MCS（調變編碼方案）選擇策略
- 強化學習評估最佳實踐

文件版本：1.0
更新日期：2026-02-07
適用於：5G 網路切片知識蒸餾 + SAC 強化學習系統