基于DQN的俄罗斯方块游戏

本项目包含经典游戏《俄罗斯方块》的程序脚本。该程序基于强化学习进行训练。

Q-learning算法，本质上是一个决策选择，解决的是在当前状态S，应该采取动作A，达到最大收益R。在训练过程中，通过观察当前游戏状态和奖励信号来进行决策，并通过优化算法来更新其策略。这使得模型能够逐渐提升其游戏技能，并在不断尝试和反馈中改进其决策能力。

最终，经过训练的模型能够以惊人的效率和准确性玩俄罗斯方块游戏，并在游戏中取得高分。

文件结构

```
├── src
│   ├── deep_q_network.py
│   └── tetris.py
├── tensorboard
│   └── events.out.tfevents.1717144338.冰儿
├── trained_models
│   ├── tetris_500
│   ├── tetris_1000
│   ├── tetris_1059
│   ├── tetris_1500
│   ├── tetris_1634
│   ├── tetris_2000
│   ├── tetris_2239
│   ├── tetris_5493
│   ├── tetris_6657
│   ├── tetris_19422
│   └── tetris_19528
├── result.mp4
├── test.py
└── train.py
```

项目的主要代码文件夹为`src/`。其中，`tensorboard/` 包含训练过程的终端文本和 数据曲线；`trained_model/` 包括在不同阶段的模型权重文件，用于在`test.py` 中运行测试，观看两种智能代理在不同训练阶段的实际游戏效果。

运行指南

本项目基于 Python 编程语言，用到的外部代码库主要包括 Pygame、OpenAI cv、torch、numpy、matplotlib、tensorboard 等。以下配置过程已在 Windows 11 系统上测试通过。

环境配置

```
# 创建 conda 环境(在anaconda navigator)，将其命名为 rl,Python 版本 3.11.9
conda activate rl

# 安装外部代码库
pip install -r system_info.txt
```

运行测试

环境配置完成后，可以运行`test.py`进行测试，观察智能代理在不同训练阶段的实际表现。

```
cd [项目上级文件夹]/俄罗斯方块
python test.py
```

模型权重文件存储在`trained_models/`文件夹下。测试脚本默认调用训练完成后的模型。如果需要观察不同训练阶段的AI表现，可将测试脚本中的`MODEL_PATH`变量修改为其它模型的文件路径。

训练模型

如果需要重新训练模型，可以在运行 `train.py`

```
cd [项目上级文件夹]/俄罗斯方块
python train.py
```

查看曲线

项目中包含了训练过程的 Tensorboard 曲线图，可以使用 Tensorboard 查看其中的详细数据。

H3

```
cd [项目上级文件夹]/俄罗斯方块
tensorboard --
logdir=<tensorboard>
```

在浏览器中打开 Tensorboard 服务默认地址 `http://localhost:6006/`，即可查看训练过程的交互式曲线图。