《高级软件工程》
课程项目指南
（2019 版）

罗铁坚、周文璋、俞永生

目录

一．项目要求

高级软件工程这门课不仅强调学生完成如何发明创造一个有价值的软件的认知升级，同时考察他们在思维升级后，能设计实现一个可以运行的软件原型来体现他的升级效果和学习能力。

我们设计了三类有价值的软件项目供学生选择挑战。

第一类是常规软件系统。这类系统的特点是，人类观察现实世界的业务需求和问题，并归纳出有价值的软件功能，使用程序设计语言实现软件系统（如选课系统、淘宝、微信等）。

第二类是智能软件系统。这类软件的特点是，核心功能（如博弈类，围棋、麻将、扑克牌或魔方等）人类经过长时间学习也不易到达高水平、而且人类也总结不出用算法，并用程序设计语言表达出来的能力）要通过机器学习才能获得的能力。因此，需要结合常规软件设计方法和机器学习（强化或监督）来共同完成的软件。

第三类是软件生产线软件（如 Dev/Ops, 或 Auto AI/ML。这类软件的特点是集成第一类或第二类软件的优势和特点，解决软件开发中的高效率和高质量问题。

无论是哪一类软件，学生完成的课程项目必须到达如下的目标：

1、 广泛调研、实证研究、分析对比，回答为什么你开发的这款软件是有价值的？

2、 软件需求定义清楚（版本 1 的三个主要功能的是什么？能用用户故事和 Cucumber 或 Rspec 等 BDD 需求描述语言表达）、软件架构合理、选择语言和应用框架得当。

3、 代码组织得当、所有代码必须有单元测试、系统测试和验收测试代码。

4、 全部代码必须托管到 GitHub 上，并且部署上云，作出演示。部署方式支持敏捷迭代开发（即如何新功能的代码修改后，马上能看到部署效果）。

5、 完成一份项目报告（或学术论文）和汇报 PPT。

6、 10 月底之前组织一次课程项目的开题汇报交流，根据上交的 PPT，选择 10 个组汇报。

二. 观察思考

1. 观察分析一款智能软件

阅读网站上的论文，体验网站的功能（http://deepcube.igb.uci.edu），这个网站是由 Forest Agostinelli， Stephen McAleer 等人在<<Solving the Rubik's Cube with Deep Reinforcement Learning and Search>>中魔方工作的可视化展示，主要用于打乱魔方的复原以及可视化复原

魔方的过程。在网址的下面，提供了作者相关工作和论文发布情况以及算法源码。

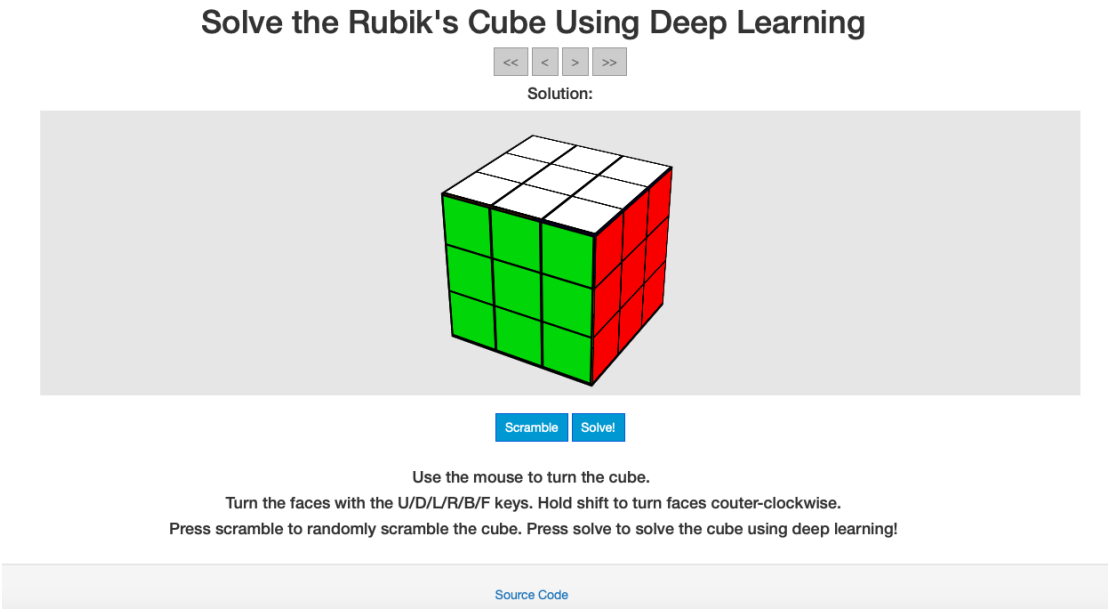


图 1.1 网站模型

2. 构思一款智能软件

本课程项目希望同学们通过对已有网站的工作进行思考，能够基于这篇论文的工作，开发具有创新有趣的系统。同时在完成课程项目过程，能够学习一些人工智能相关知识，以及体会软件工程在开发人工智能系统的整个流程。

三．选题建议

1. 给定条件

软件资源：

DeepCubeA 源码：为了更好的激发同学们的创新性，本工程将涉及到的算法源码的环境和结构进行详细介绍，同学们可以有更好的理解，在此基础上更容易开发更多创新功能的系统。

网站参考模型：本项目提供了一模版网站作为参考：http://deepcube.igb.uci.edu，主要提供能魔方的打乱，复原以及复原过程的可视化展示。根据这个网站，同学们可以开发具有类似功能或者更多有趣的功能的网站系统。

网站开发平台：魔方算法是基于 python 开发，为了兼容后台算法的运行，这里提供了网站开发平台 Django，并详细介绍 Django 的安装，工程建立，和网站发布流程，提供 Django 学习的网址。

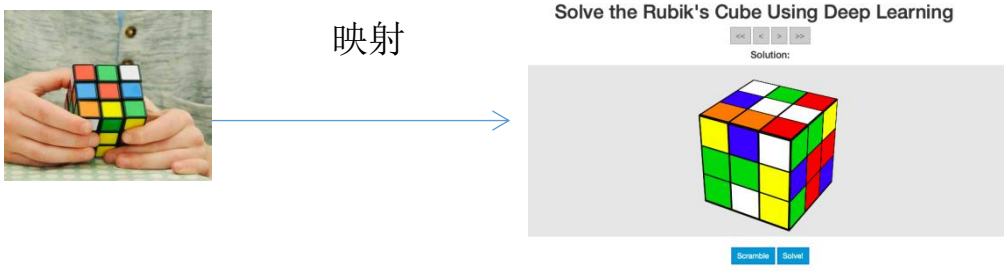
硬件资源：

本项目提供阿里云 P100 的服务器，包含显卡和大内存，用于学生网站的发布和运行。

2. 基本要求

(1). 参考网址模型，基于<<Solving the Rubik's Cube with Deep Reinforcement Learning and Search>>文中的工作，开发一套网站系统。这个系统具备基本功能：游戏的可视化；DeepCubeA 算法对游戏的求解并可视化算法完成游戏的过程。

(2). 参考网址，开始魔方的状态是由 Scramble 按钮随机生成的，然后运行 Solve!进行魔方复原。在现实中，人们打乱魔方肯能是随机的，我们希望能够将现实中某个魔方状态通过某种形式输入到网站中去，作为网站魔方的初始状态，然后通过 Solve!按钮就可以得到复原魔方的基本步骤。这样网站就能和现实人们进行互动，完成魔方相关教学。



实际魔方状态　　网站魔方状态

图 2.1 人机交互模型

(3). 参考网址中进行魔方复原的引擎使用的是论文中训练好的 3*3 魔方网络模型，论文提到这个方法可以解决在没有任何特定领域知识的情况下，在大的空间状态下，可以有效的

学习从目标状态反方向解决日益困难的状态。这个任务希望同学能够基于现有的条件，训练出其他魔方形式(如 5*5 的魔方)的网络模型，代替(1)和(2)中系统的魔方形式。

3. 验收标准

同学们将自己开发的系统提交到我们提供的服务器上并进行发布，或者自己可以发布自己的服务器上，向我们提供网站的对外网址 url，根据网址实现的功能进行打分，完成参考网站基本功能获取 80 分，如有有趣的创新功能可以进行一定的加分奖励。

为了验证基本要求(2)中的人机交互功能，我们会随机将魔方打乱，然后通过系统复原的动作，查看该魔方是否能够进行复原。完成此功能可获得 90 分。

对于要求更高的同学，可以完成基本要求(3)中的功能，此同学可以借鉴论文中测试魔方的指标：事件，最短路径长度和内存消耗等，作出文中类似的比较，提供我们相关文档。同时提供我们新魔方形式的网站，我们根据文档上面的功能进行测试。完成此功能可获得满分。

四. 参考实现

4.1 预备知识

4.1.1 网站平台开发

目前采用基于 python 的 Django 平台设计网站，可以兼容前后端以及 Cube 的深度学习后台代码(详细学习: https://www.djangoproject.com)。

①。 安装

Windows：

官网下载： http://www.djangoproject.com/download/

安装命令：python setup.py install

Linux：

pip install Django

验证是否正确安装：在后台运行 python，然后输入：

import django

print(django.VERSION)

打印出 django 版本：(1, 10, 1, 'final', 1)

②。 创建工程

使用 django-admin 来创建 项目，比如 django-admin startproject DeepCube。然后在 DeepCube 目录下，会看到以下目录和文件：DeepCube 和 manage.py。DeepCube 是项目容器，主要存放项目相关文件，配置文件和网站资源文件。Manage.py 是一个实用的命令行工具，可以与 Django 项目进行交互。

③。 启动服务器

使用 python3 manage.py runserver 0.0.0.0:8000 启动自己的网站服务，其中 0.0.0.0 让其它电脑可连接到开发服务器，8000 为端口号。如果不说明，那么端口号默认为 8000。在浏览器输入你服务器的 ip（这里输入我们的本机 IP 地址： 127.0.0.1:8000） 及端口号，就可以看见我们的网站：

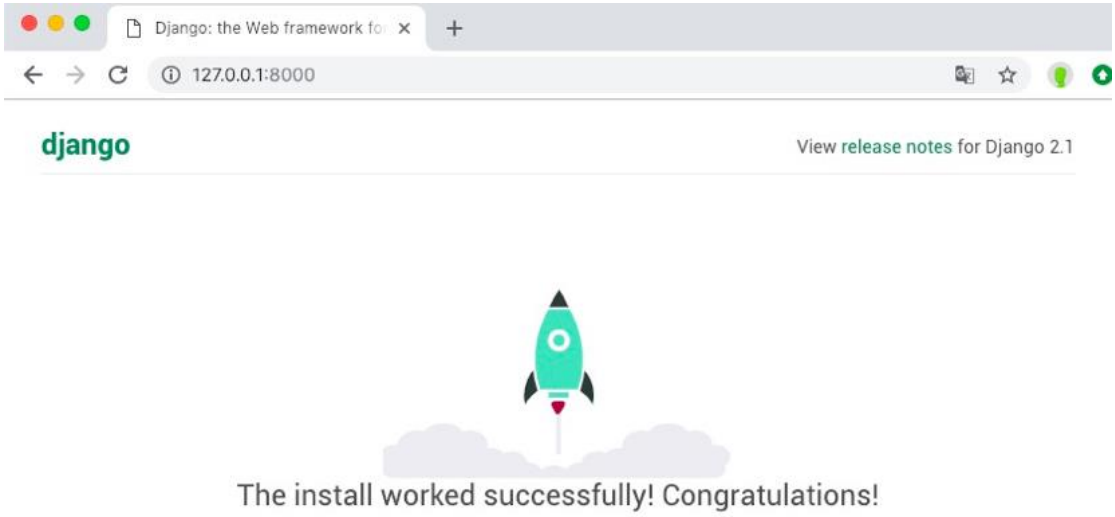


图 3.1 启动网站服务

(1)。DeepCubeA 代码

1). 相关工作的链接

deepcubea 相关论文：<<Solving the Rubik’s cube with deep reinforcement learning and search>>

源代码官网：https://codeocean.com/capsule/5723040/tree/v1

2). 环境配置

Docker Community Edition (CE): https://www.docker.com/community-edition

nvidia-docker: https://github.com/NVIDIA/nvidia-docker/

系统依赖库安装：

```
apt-get update \
 && apt-get install -y --no-install-recommends \
    "build-essential=12.4ubuntu1" \
    "libboost-all-dev=1.65.1.0ubuntu1" \
    "libboost-dev=1.65.1.0ubuntu1" \
 && rm -rf /var/lib/apt/lists/*
```

开发平台(安装了 conda):

```
conda install --yes \
    python==2.7.5 \
    tensorflow-gpu==1.8.0 \
 && conda clean --yes --all
```

平台依赖库：

```
pip install --upgrade \
```

dm-sonnet==1.10 \

matplotlib==2.2.3

3). 源码简单介绍

源码工程目录下有四个子目录：code，data，environment，metdata。metdata 主要介绍作者相关工作和论文发表情况；environment 主要介绍程序的开发环境和依赖库；data 存储了测试使用的输出数据；code 存储工程的程序，该目录下有：environments，extra，labeledData，ml_utils，savedModels，scripts，solvers 和 testData。其中 environments 为各种游戏环境程序，savedModels 用于保存训练好的网络模型，testData 用于保存测试的结果，scripts 存放训练和测试代码的入口程序，ml_utils 存放程序中依赖的库文件。

4.2 需求和设计

4.2.1 任务一(案例)

为了让同学们更容易理解和实现系统开发的过程，这里我们开发参考模型网站为例，同学们可以将此作为模版，开发自己的系统网站。首先该大作业的基本任务，即复现类似 http://deepcube.igb.uci.edu 的网页。需要最终满足以下两点：

1、 研究 DeepCubeA 源码，实现针对 3*3 魔方的端到端代码（原来的代码涉及了很多其他游戏）。令输入为 state，输出为 action

2、 形成可视化界面，将魔方 state 向量映射为魔方图形，将 action 显示到界面上，同时设计必要的显示框和按钮，参考原网页。

3、 需要分为前端和服务端，并编写出所需接口。

然后是在此基础上进行新功能的迭代，提示：界面可以 3D 旋转魔方图形，暂停键，可自己设置初始状态（照相自动识别），算法扩展为 5*5 魔方等。

(1). 系统功能

为了达成基础版本的 DeepCubeA，我们给出以下的开发流程：

1、 首先下载源码与模型。

2、 然后分析源码。课程要求的基础版本不涉及到算法改进，所以只用理清楚输入和输出，即魔方的当前状态（state）是如何作为参数传入计算函数中的，以及最终计算的结果，也就是接下来魔方要进行的操作（action）该如何获取。除此之外，state 和 action 的数据格式，需要自行分析。

3、 将 DeepCubeA 源码构建成一个端到端的接口，以便之后服务器端直接传入 state 参数并获取对应结果。是否需要新编写代码，还是直接整理源码形成 pipline，由各组

自行分析。

4、 实现前端（client 端）和服务器端之间的通信。方法可参考下面的(2)云上部署。

5、 最后参照原网页，实现可交互的项目展示网页。

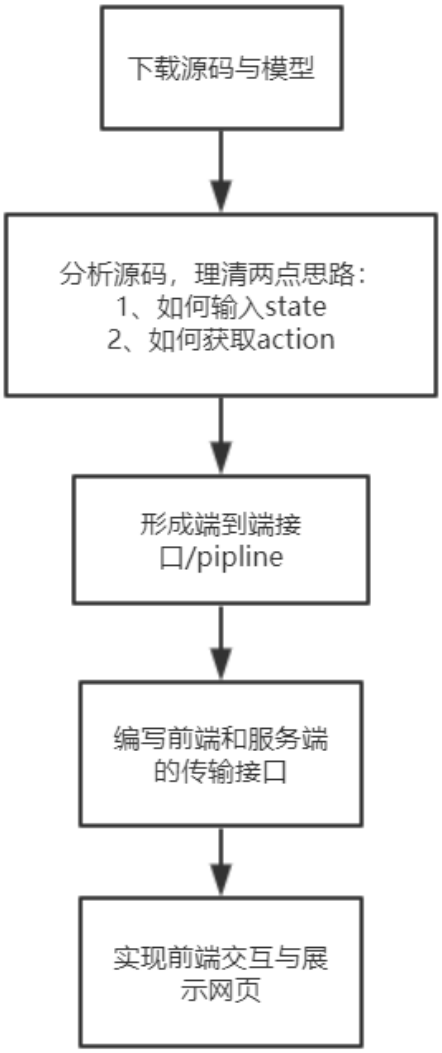


图 3.2 系统设计参考流程图

(2). 云上部署

前端部分放在本地机器上即可。后端部分，即 DeepCubeA 端到端模型放置于服务器（其实也是放在自己设备上，但必须写出接口）。

可以用第三方库 bottle 部署 server 端的调用接口，当前端用 request 库发送 state 的编码到 sever 端时，server 端再将 json 文件作为参数传入 DeepCubaA 端到端算法，并向前端返回计算结果。具体过程如图：

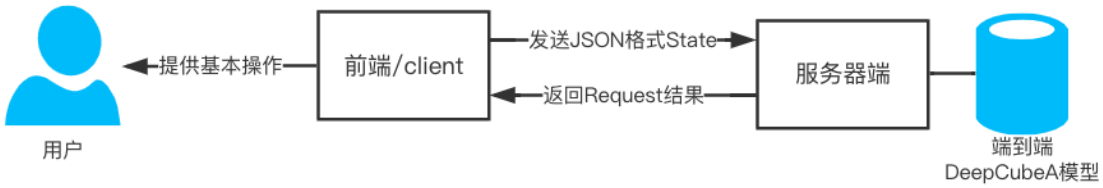


图 3.3 C/S 部署结构

4.3 系统实现

4.3.1 预期效果

(1) 网站界面

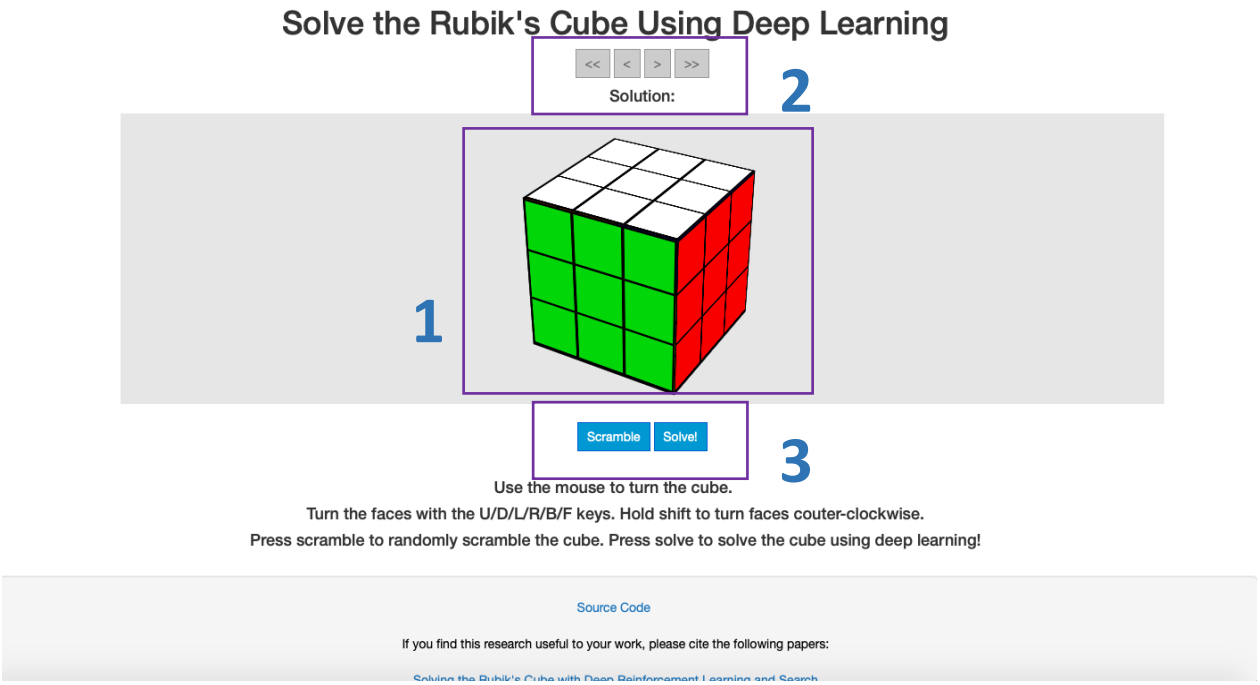


图 3.4 网站页面

图 3.4 的网站页面包含 3 个功能模块，分别为：1. 魔方可视化展示; 2. 魔方打乱和魔方复原功能；3. 魔方复原过程可视化展示。

1). 魔方打乱：通过点击 Scramble 按钮，程序会自动随机旋转魔方，生成某种随即状态，如图 3.5 所示。

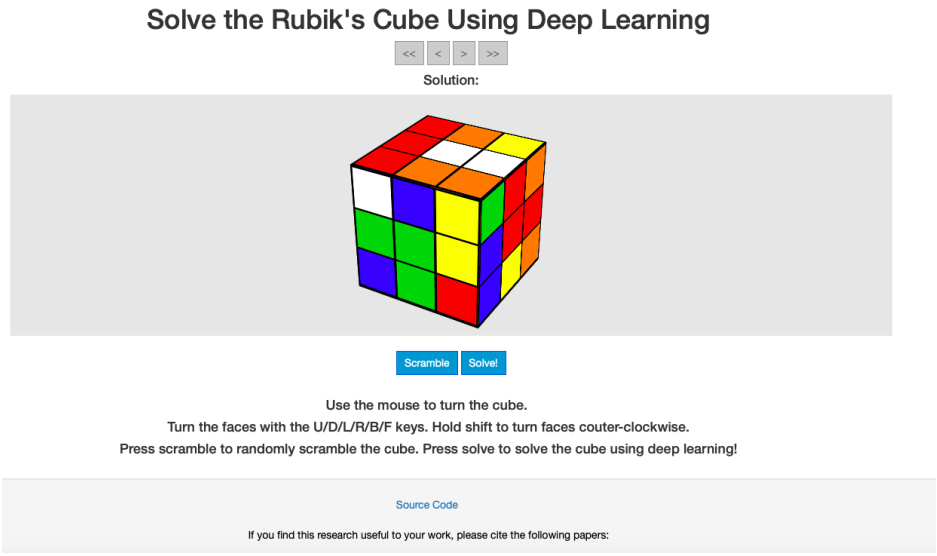


图 3.5 随机打乱魔方

其中 Scramble 按钮通过随机旋转步长以及随机每次旋转状态，从而生成魔方随机打乱的状态：

```
function scrambleCube() {
    disableInput();
    clearSoln();

    numMoves = randInt(100,200);
    for (var i = 0; i < numMoves; i++) {
        moves.push(legalMoves[randInt(0,legalMoves.length)]);
    }

    nextState(0);
}
```

图 3.6 随机打乱魔方程序

2). 魔方复原：通过点击 Solve!按钮，程序会向后台服务器发送请求，后台对魔方复原后将数据传送给前端，在图 3.4 的方框 2 中会出现上下左右前后旋转策略序列，并动态显示复原过程：

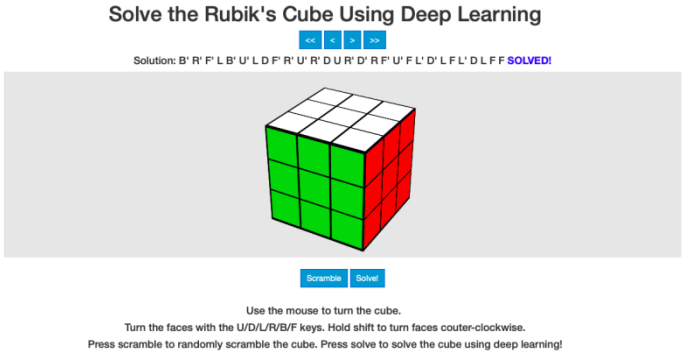


图 3.7 魔方复原

前端网页通过发送请求给服务器，服务器将魔方复原运行的结果储存在 json 文件中，并发送给前端网页，网页脚本进行解析，进行相应可视化：

```
function solveCube() {
    disableInput();
    clearSoln();
    document.getElementById("solution_text").innerHTML = "SOLVING..."
    $.ajax({
        url: '/solve',
        data: {"state": JSON.stringify(state)},
        type: 'POST',
        dataType: 'json',
        success: function(response) {
            solveStartState = JSON.parse(JSON.stringify(state))
            solveMoves = response["moves"];
            solveMoves_rev = response["moves_rev"];
            solution_text = response["solve_text"];
            solution_text.push("SOLVED!")
            setSolnText(true);

            moves = JSON.parse(JSON.stringify(solveMoves))

            setTimeout(function(){nextState(500)}, 500);
        },
        error: function(error) {
                console.log(error);
                document.getElementById("solution_text").innerHTML = "..."
                setTimeout(function(){solveCube()}, 500);
        },
    });
}
```

图 3.8 复原魔方接口

4.3.2 服务器接口设计

4.4 结论与展望

通过本课程项目设计，体会高级软件工程设计的基本要点，详细了解系统设计从需求分析到系统设计与实现的过程，同时了解人工智能相关知识以及智能软件开发过程。并通过拓展性任务激发同学兴趣和创造性。