吉林大学珠海学院

毕 业 论 文

基于SpringCloud和深度学习的“新冠”肺炎智能辅助诊断系统

学　　院：　计算机学院
专　业　名　称：　软件工程
学　生　姓　名：　洪佳滢
学　　号：　04191727
指导教师姓名、职称：　郭晓燕 讲师

完成日期：2023 年 2 月 27 日

基于 SpringCloud 和深度学习的“新冠”肺炎智能辅助诊断系统

摘　要

由于新型冠状肺炎的侵袭，其高传染力致使很多人难以逃脱，一线医生的工作强度也随之增长。影像学检测相较于核酸检测，更能够满足快速诊断的需求。并且诊断结果作为医疗数据，其私密性与安全系数较高。因此，针对上述的需求，本系统在拥有高安全性的前提下实现以智能诊断为主等功能。

为了实现上述功能，本系统将以 Java 和 Python 作为开发语言，采用微服务的架构理念，并使用 SpringCloud 加 SpringBoot 作为微服务架构的具体技术实现。使用 PyTorch 作为深度学习框架，使用 ResNets18 的网络结构并引入迁移学习，构建智能诊断模型。本课题将从 SpringCloud 和深度学习出发，针对服务网关、服务注册与发现、应用通信和配置中心等方面，研究该系统的设计与实现。文章将从研究背景入手，从需求分析再到总体设计，探索系统构建的完整过程，并从多方面进行论述，包括系统最后的测试，都有详细的过程记录。最后，本文对系统整个研究、设计及实现过程进行了总结，并对系统今后的进一步建设和发展的方向进行了讨论。

关键词：“新冠”肺炎诊断；SpringCloud；深度学习；辅助诊断

An intelligent auxiliary diagnosis system for COVID-19 based on SpringCloud and deep learning

Abstract

Due to the invasion of the novel coronavirus, its high infectivity makes it difficult for many people to escape. In the situation of explosive infection, the work intensity of the front-line doctors also increases. Compared with nucleic acid detection, imaging detection can better meet the needs of rapid diagnosis. What is more,as medical data, diagnosis results are highly private and secure. Therefore, in view of the above requirements, the system under the premise of high security to achieve intelligent diagnosis and other functions.

In order to realize the above functions, this system will use Java and Python as development languages, adopt the concept of micro-service architecture, and use SpringCloud and SpringBoot as specific technology implementation of micro-service architecture. PyTorch is used as the deep learning framework, ResNets18 network structure is used and transfer learning is introduced to build the intelligent diagnosis model. Starting from SpringCloud and deep learning, this topic will study the design and implementation of the system in terms of service gateway, service registration and discovery, application communication and configuration center, etc. This paper will start from the research background, from the requirements analysis to the overall design, explore the complete process of system construction, and discuss from various aspects, including the final test of the system, with detailed process records. Finally, the whole research, design and implementation process of the system are summarized, and the future construction and development direction of the system are discussed.

Key words: Diagnosis of COVID-19; SpringCloud; Deep learning; Auxiliary diagnosis

目　录

1 绪 论

1.1 研究背景与意义

2019 年下半年，一种被命名为新型冠状肺炎病毒（Corona Virus Disease 2019，COVID-10）[1]逐渐席卷全球，而现如今，该病毒也已经从原始毒株经历了多次变异，目前流行的奥密可戎毒株，其传播力明显增强，毒力虽明显减弱，但是仍存在重症的情况[2]。截至 2022 年 12 月 5 日，全球感染新冠肺炎的人数超 6 亿例。

目前，新型冠状病毒的诊断依据主要是核酸检测和医学影像检测。两者并用更有利于诊断。胸部 X 光技术作为医学影响检测的主要手段，其技术成熟、价格适中、成像质量清晰且对人体辐射低等优点，成为了这次抗击“新冠”疫情的主力军。

随着“新冠”肺炎疫情防控形势的发展，大量疑似病例出现，由于新冠肺炎核酸检测假阴性比率高，影像鉴别难度大，严重影响了新冠肺炎的快速识别诊断，因此爆发式快速诊断需求让一线医生不堪重负。并且，从医学临床角度上来看，“新冠”肺炎和其他普通的病毒性肺炎的症状也是相似的，有发热、乏力、咳嗽及可能出现胸闷、呼吸困难等症状。所以单从临床表现上区分“新冠”肺炎与其他病毒性肺炎也是不容易的事情。在此大背景下，“新冠”肺炎辅助诊断系统的投入使用对快速有效的临床诊断有很大帮助，人工智能已逐步应用于医学影像领域。

在医疗软件的开发中，良好的操作系统对医生诊疗效率与正确率有明显的提升，而性能不足的系统则会引起相反的作用。随着技术的成熟，“新冠”肺炎辅助诊断系统的投入使用使得因病人过多、图片精度与数量增大而导致的检阅效率不佳的问题得到改善；智能化技术的应用程度提高，也让医生的工作重复性与机械化操作降低。人工智能辅助诊断系统的架构和应用，有望为“新冠”病毒病例筛查提供先驱医疗诊断，提供有效即时性智能辅助，节约诊断时间，同时借助深度学习为“新冠”病毒的影像学研究提供更多有效支持。

基于现在系统的发展形势，以及本人对相关领域的了解，希望对系统的发展贡献自己的一份力量。

影像学诊断与人工智能相互结合的“新冠”肺炎辅助诊断系统，可以快速诊断病情，为疫情防控争取宝贵的时间，它能够通过深度学习进行影像的深度研读、秒级运算、出具报告的时间从几十分钟加速到了几分钟。

在“肺炎”辅助诊断系统中，信息量较大，数据的安全性和保密性较高且数据积累与精准数据保障等问题层出不穷。随着分布式架构、微服务等新兴技术的发展，系统构建方式的进步为解决相关问题创造了新的契机。在解决系统间互联互通的系统整合以及系统的扩展性可维护性等多种场景提供了很好的支持。

1.2 研究开发现状分析

1996 年，医疗卫生领域的第一个智能诊断模型是由 Ledley 设计的[3]，其模型根据输入的信息判断患者是否患有肺癌。随着智能技术的不断优化升级，我国即将进入“互联网+医疗健康”3.0 时代，该阶段特征是医疗健康大数据、智能化辅助诊断系统的广泛应用。由天坛医院、解放军总医院、BioMind 联合推出的“新冠” AI 定性诊断系统，不仅能够实现“肺炎诊断”，还能实现新冠肺炎与其他肺炎（病毒性肺炎、细菌性肺炎等）的进一步鉴别诊断[4]。人工智能与影像学相互结合，缓解了因爆发式“新冠”肺炎疫情所带来的快速诊断需求的压力。目前，该系统已陆续投入使用。

现阶段，医院信息系统常见的有 PACS（影像归档和通信系统）和 RIS（放射科信息系统）等，PACS 主要是将各种医学影像通过接口保存起来，为后续调回使用。而 RIS 则是放射科工作流程的管理系统，包括从预约到产片、审核等完整流程。传统的 PACS 系统虽拥有图像管理与资源共享等技术，但是缺乏更敏捷的智能诊断功能；同理，传统的 RIS 系统也有如此可改进的地方[5]。而本课题研究的“新冠”肺炎智能辅助诊断系统将会很好地为这两个系统作为一种补足，并且根据现阶段“新冠”肺炎疫情的形势预测，其将可能呈现阶段性爆发式的情形。快速诊断的需求也将会成为一种趋势，随着业务需求与数据量的增加，该系统必然是疫情防控不可或缺的有效工具。

1.3 主要研究内容

该课题的主要目的是通过将人工智能技术与影像学的相互融合，为医院构建一个“新冠”肺炎智能辅助诊断系统。该系统通过图像识别肺部 X 光片，以达到快速筛查肺炎的目的。所以，该课题的主要研究内容就是如何利用相关技术设计并实现这样的系统，为用户提供高质量、高可靠并且可拓展、易维护的系统。

主要研究的内容如下：

1. 相关理论和技术的研究。即对于构建这样的系统需要的理论知识与技术的研究，以及对于搭建该系统的框架认识与了解。并且考虑后续系统所可能的扩展。

2. 系统各方面的需求分析。从医生与系统管理者的角度出发，详细分析系统所应当拥有的功能模块。

3. 系统的设计与实现。在对系统进行需求分析后，需要对系统进行整体的架构设计，应当考虑到如何用微服务架构解决系统问题。并且需要对系统子模块进行划分，设计模块间的耦合关系以及对系统进行数据组织。还应该考虑如何选取相适应的深度学习算法，来提高该系统的准确率。之后，依据这些设计对系统进行落地实现。

4. 系统的测试。在该系统实现后，需要对各模块的功能进行用例测试，并且还需要对系统进行性能测试，以确定其是否达到预期水平。

5. 对系统在研究设计与开发过程中的总结，对系统下一步的发展方向进行讨论，并

提出后续工作计划和工作重点。

2 相关理论和技术

通过第一章对研究背景、研究内容的深入了解，能够发现研究该课题的必要性。本课题的主要内容是设计并实现基于 SpringCloud 和深度学习“新冠”肺炎智能辅助诊断系统，涉及的关键技术大体上可以分为两个层面：

首先是系统架构方面所涉及到的关键技术，作为整个系统基础支撑的分布式微服务架构 SpringCloud；另一个是该课题所采用的深度神经网络——ResNets18 算法框架以及其基本架构 ResNets，并且引入迁移学习的方法。因此本章主要阐述深度学习中的图像识别技术以及微服务架构 SpringCloud 的相关技术理论，获得部分先验知识。

2.1 肺炎图像识别技术

2.1.1 图像处理

图像处理狭义上是对输入图像进行某种变换得到输出图像，是一种图像到图像的过程。主要指对图像进行各种操作以改善图像的视觉效果，或对图像进行压缩编码以减少所需存储空间或传输时间、传输通路的要求，包括图像增强、图像恢复和重建，以及图像编码。常见的图像处理手段包括图像平滑、锐化、伪彩、放大缩小、旋转拖移等。

良好的图像预处理是特征提取的前提，可以大大降低特征提取的难度，从而提高图像识别率。该课题采用的图像预处理的方法是图像标准化处理，通过去均值的方式，可以让数据集中落在指定范围之内，使模型在构建过程中，能够拥有更快的收敛速度，且能够更好地搜索最优解，更容易取得训练之后的泛化效果。

该课题所采用的数据标准化公式 2-1 如下：

$$x = \frac{x - mean}{std} \qquad (2\text{-}1)$$

mean 表示各通道的均值，std 表示各通道的方差。

各通道的均值和方差要在图像标准化处理之前先计算好再传进去，要不然每次图像标准化都需要把所有图片都读取一遍计算均值和方差。

2.1.2 图像分类

图像分类是输入一副或多副图像，输出对该图像内容分类的方法，预测其分类结果，根据预测结果判定图像类别[6]。

2.1.2.1 ResNets 网络

在 2015 年的 ImageNet 图片分类比赛中，ResNets（残差网络）夺魁到达顶峰。随着

深度神经网络的发展，因其网络层数之多而冠以“深度”之名，但是层数的增加，训练效果就一定越来越好吗？并且过深的网络结构应该如何进行优化，随着层数的增加，又该如何避免梯度消失和梯度爆炸。这一系列的问题接踵而至，科学家何恺明[7]在实验室发现后，与同事开发了 ResNets（残差网络）。

ResNets（Residual Networks）增加了一个短路层（shortcut connection），隔层相连，弱化了每一层之间的联系，梯度在一般的神经网络中传播需要进行一定程度上的衰减，但是有了短路层之后，梯度可以避免一定的衰减，换言之，残差网络在某种程度上能够保留短路层之前的网络学习效果。

因此，ResNets 的基本框架就由如下图 2-1 的残差块所组成。

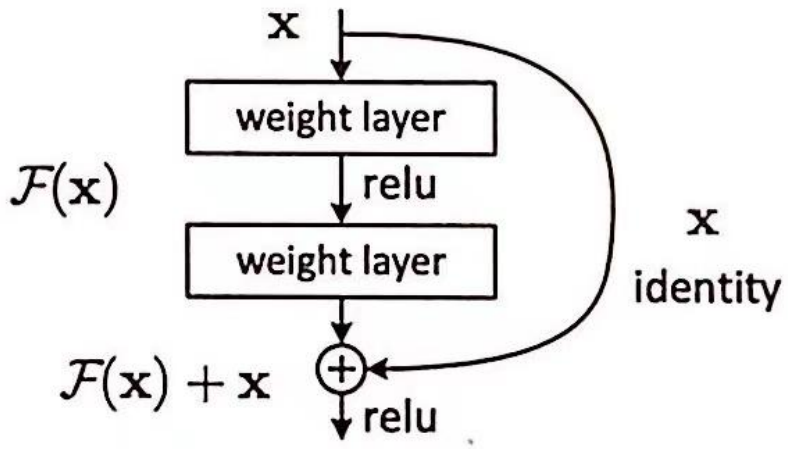

图 2-1 残差块结构

该残差块的数学表达式可以写成如下公式 2-2：

$$y=\sigma\left(F\left(x,W\right)+x\right) \tag{2-2}$$

其中，$\sigma(\cdot)$代表激活函数，$F(\cdot)$代表残差函数，W 代表残差块内的所有权重。

在一般的神经网络中，若每一层的误差梯度小于 1，反向传播时，网络越深，梯度越趋近于 0；反之，梯度将会越来越大。这样的梯度弥散和梯度爆炸会造成灾难性的损失。在拥有了残差块之后，由于恒等函数的导数恒为 1，梯度回传时，先前的梯度仍然可以反向传播，信号可以轻松地在整个网络中传播，从而避免了梯度弥散。

ResNets 使用 4 个由残差块组成的模块，每个模块使用若干个同样输出通道数的残差块。 第一个模块的通道数同输入通道数一致。之后的每个模块在第一个残差块里将上一个模块的通道数翻倍，并将高和宽减半。

ResNets 的一个重要的设计原则时：当特征映射大小降低一半时，特征映射的数量增加一倍，这保持了网络层的复杂度。因此 ResNets 网络结构很好的解决了深度网络的退化问题。

2.1.2.2 ResNets18 网络结构

ResNets18 网络结构的基本含义是以 ResNets 作为基本架构，拥有 18 层带有权重的网络深度，包括卷积层和全连接层，但不包括池化层和 BN 层。将 18 层分为 6 个组件，即由四个残差块以及连接块的输入输出相关操作层组成。

其具体的网络结构图 2-2 如下：

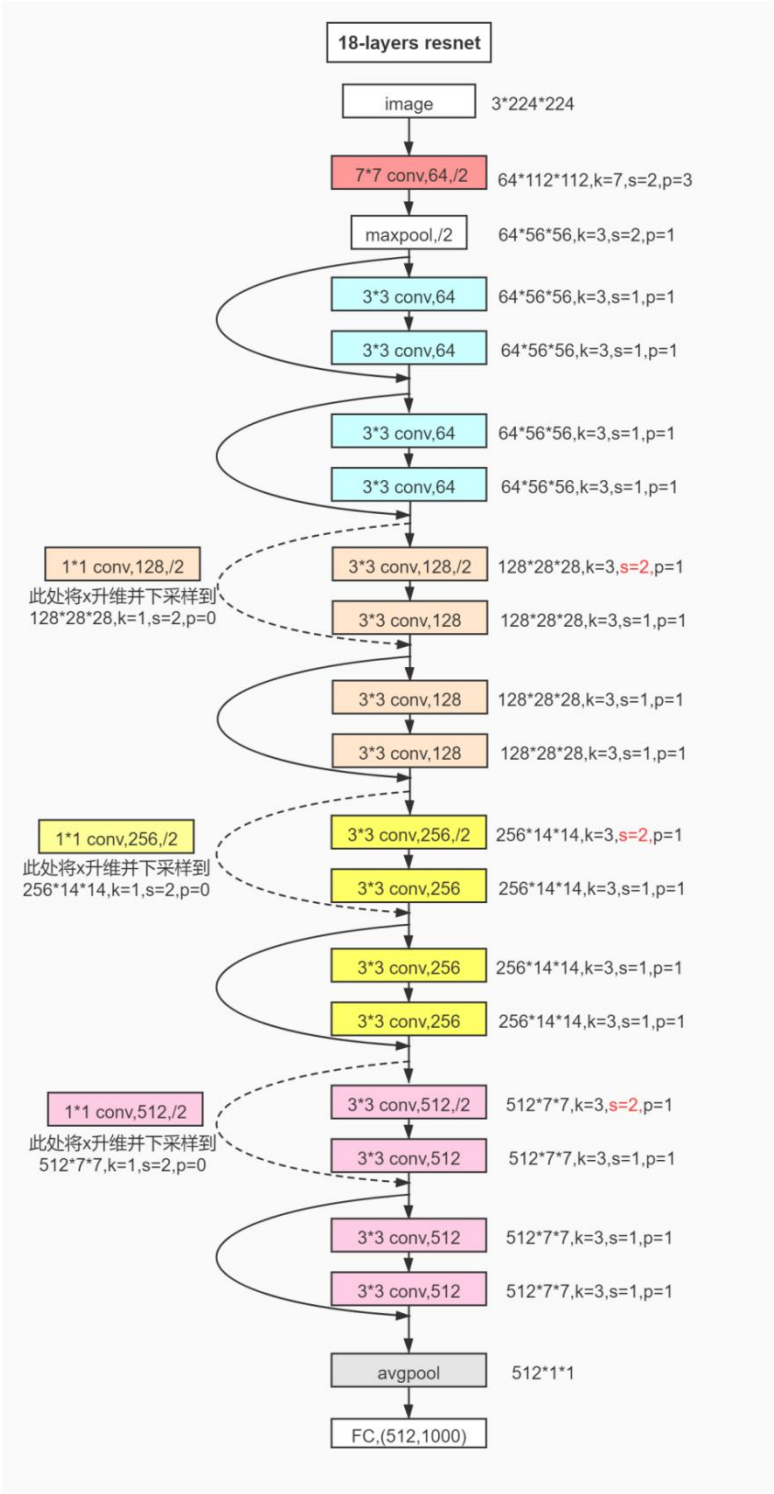

图 2-2 RestNets18 网络结构

假设输入数据的大小为 3*224*224，即通道数为 3，大小为 224*224。在经过卷积核大

小为 7*7 的卷积层之后，输出的数据大小为 64*224*224。再经过池化层之后提取特征，将数据量减半，此过程并不会改变数据的通道数。紧接着数据将会经过 4 个由残差块组成的模块，每个残差块都会使数据的通道数翻倍，并且输出数据的大小全部减少一倍。最后，经过平均池化层与全连接层完成数据的分类。

ResNets18 的网络结构因其深度，可以做到预测更复杂的模式与特征；其次，提升了网络的性能，因为该网络能够有效地利用较少的特征；该结构也提供了更良好的参数优化，在不同的数据集中也能够拥有良好的性能。

2.1.2.3 迁移学习

在深度神经网络算法的应用过程中，如果遇到数据规模较大的问题，是必须要花费大量的算力和时间去训练模型和优化参数，而往往在消耗大量时间去构建模型之后得到的结果却不尽人意，为此其性价比非常的低。如果用训练好的模型能够去解决一类问题而不仅仅是一个问题时，模型的性价比就会有较大的提升[8]。这就促使了迁移模型的出现，通过对一个已经训练好的模型进行微调，去解决相似问题，并获得较好的结果。

通常情况下当使用的数据量与计算资源无法支持模型的训练时，迁移学习会被用来辅助计算。迁移学习的总体思路可以概括为：开发算法来最大限度地利用有标注的领域的知识，来辅助目标领域的知识获取和学习。其核心就是找到源领域和目标领域之间的相似性并加以合理利用。

2.2 SpringCloud 分布式微服务架构

本课题的研究目标是构建一个能为医生提供良好体验的“新冠”肺炎智能辅助诊断系统，该系统不仅仅是提供辅助诊断功能，还拥有医患信息管理等功能。为此，该系统需要将医院的数据特点作为构建系统的重要考虑要素。

随着医疗相关的信息系统逐渐发展，整体的医疗水平与管理水平也逐步提高，但是医院信息系统所涉及的数据种类繁多，包括患者的诊断信息与隐私数据，并且医疗系统还需要应对疾病的变化，就如“新冠”疫情的肺炎筛查需求迸发，业务规模的大幅度增长，传统的单体架构已无法满足需求的快速变更。微服务结构模式的处理可以很好地解决此类问题。本课题依托 SpringCloud，将服务原子化拆分，以此解决系统间互联互通的系统整合以及系统的扩展性可维护性等。

2.2.1 注册中心 Eureka

在大数据及高并发的环境下，一般不再使用单一服务器而是使用服务器集群完成工作，因为在微服务架构中，一个业务可能需要有多个微服务配合来共同完成。当业务越来越多且复杂的情况下，微服务的数量也会越来越庞大，各微服务之间地调用关系也越来越复杂。

要想能够快速找到相应的微服务地址，在多个相同微服务提供者中快速找到最适合的，这需要注册中心的支持。它可以记录每一个服务的 IP 地址、端口以及所对应的服务。当需要某一服务时，只需到注册中心去查找最佳的微服务提供者地址和端口即可。SpringCloud 中比较典型的微服务注册中心有 Eureka、Nacos、ZooKeeper 等，本课题所采用的是 Eureka。

Eureka 是 Netflix 公司开源的产品，是一种基于 REST 的服务，提供了完整的 Service Registry 和 Service Discovery 实现。它主要有两个组件组成：Eureka 服务端和 Eureka 客户端。Eureka（Server）服务端就是注册中心，提供服务的注册和发现功能，同时通过心跳包检查服务的运行状态。客户端通过在 Eureka 服务端得到注册的服务列表，找到对应的服务地址使用。

当服务注册中心出现宕机时，其他客户端可能会出现服务调用失败的风险，为了不影响整个微服务架构的稳定性，通常采用 Eureka Server 集群的方式来提高整个架构的高可用性。Eureka 的高可用生产建议至少两台以上，通过 Eureka Server 两两相互注册的方式，避免其他服务调用失败的问题。

Eureka 的高可用方式如下图 2-3 所示：

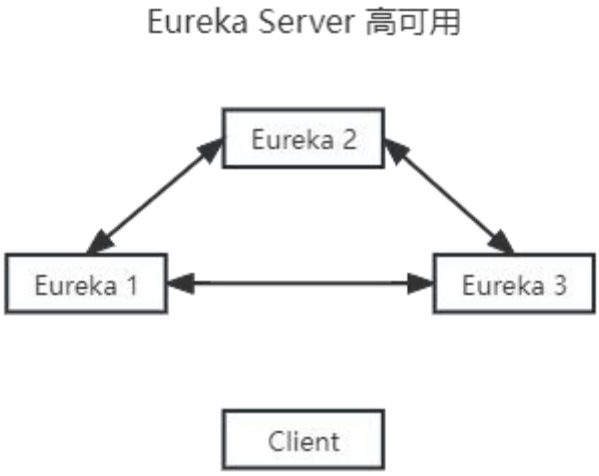

图 2-3 Eureka 的高可用

分布式系统中，服务注册中心是最重要的基础部分。

2.2.2 配置中心 Spring Cloud Config

微服务架构是由多个微服务组成，因此不方便进行维护。对于简单的项目来说，服务相关的配置都单独存储在各自的配置文件当中，但是这样的方式会产生一个问题，就是当配置发生改变时，都需要将这个服务重新启动，而相关联的服务也需要跟随重启。由以上问题得出，统一配置中心尤为重要。

将统一配置中心作为一个单独的微服务，将其配置文件放在远端的 Git 当中，同时，其他服务的相关配置也放在远端 Git 当中，当需要修改相关配置时，只需要在 Git 上面修

改相关的配置文件即可。

以下图 2-4 是统一配置中心的服务模式：

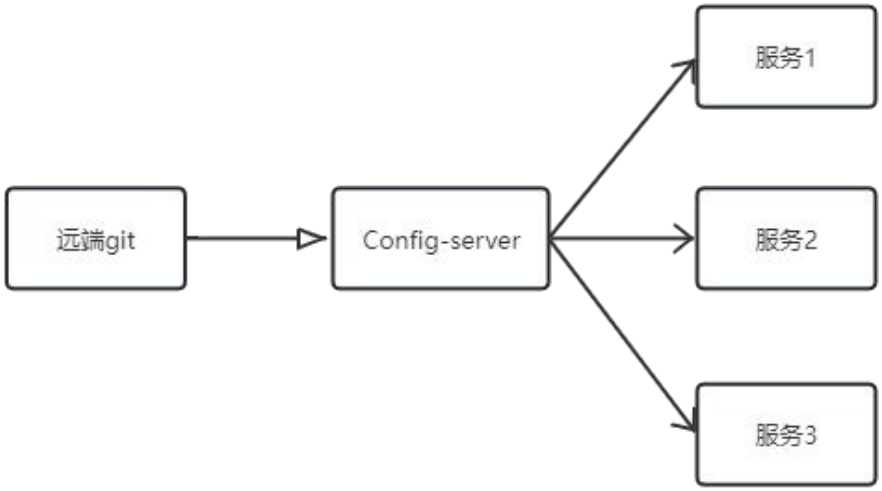

图 2-4 统一配置中心的服务模式

2.2.3 服务网关 Gateway

由于多个微服务拥有各自的地址，当需要调用时，需要记住每个服务的调用地址，这个工作量很大，要想统一管理路径，服务网关可以充当服务请求的统一接口。服务网关的常见功能有路由转发、权限校验、限流控制等作用。其基本功能如下图 2-5：

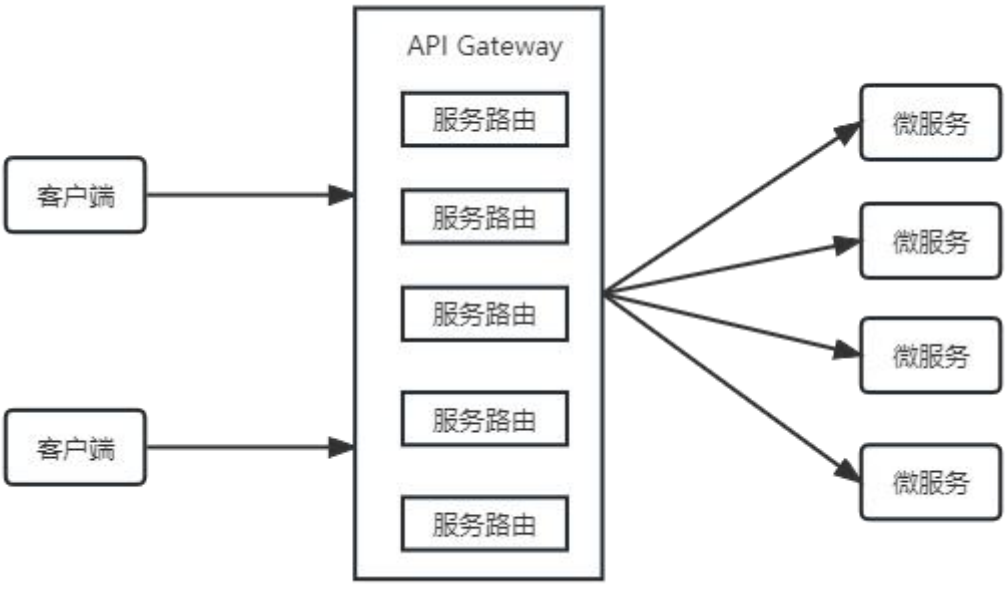

图 2-5 服务网关的基本功能

所有的请求都先经过服务网关，由网关将请求路由到合适的微服务。其好处有以下三点：

1. 简化客户端工作。客户端只需与网关交互，而不用记住各个服务的地址。

2. 降低函数间的耦合度。当接口修改时，只需修改服务网关的路由策略，从而降低了程序间的耦合性。

3. 开发人员可以专注于业务逻辑的实现,因为服务网关统一实行服务路由、访问控制、流控熔断降级等非业务相关功能。

2.2.4 应用通信与负载均衡 Feign

服务间的通信主要分为两种，HTTP 与 RPC 两种方式，其主要代表为 SpringCloud 以及阿里系 Dubbo。在 SpringCloud 中服务间的两种 Restful 调用方式分别是 RestTemplate 和 Feign。本课题所采用的是 Feign 作为应用间的通信。

以下是 Feign 跟 RestTemplate 的区别：

1. 请求方式不一样。RestTemplate 需要每个请求都拼接 url、参数以及类文件，灵活性高，但是消息封装臃肿。Feign 可以将 rest 的请求进行隐藏，不用再自行进行拼接，可以便捷的调用 HTTP 接口。

2. 底层实现方式不一样。RestTemplate 可以不需要经过注册中心，直接通过指定地址与端口号的方式进行请求接口。而 Feign 的底层实现的是动态代理，Feign 通过注解的方式，针对接口创建一个动态代理的对象，通过注解中的 name 值，在注册中心找到对应的服务，后通过映射路径请求地址，实现 HTTP 的远程调用。

根据上述的对比分析，采用 Feign 的方式会提高系统开发过程中的便利性。

同时，Feign 内部也应用了 Ribbon 作为负载均衡。当有多个服务提供者时，服务消费者将会根据负载均衡的算法，挑选服务提供者。其目标是将网络流量平均分发到多个服务器上，以提高系统整体的响应速度和可用性。

3 需求分析

本系统是基于 SpringCloud 和深度学习开发的“新冠”肺炎智能辅助诊断系统。由于疫情防范趋势呈现阶段性迸发，肺炎快速筛查需求能够帮助医生缓解压力。且因医疗数据的安全性与保障性问题，系统研发的要求也相应提升。

为解决这一系列问题，本章将详细分析该系统的各类需求，并从功能、非功能以及其他层面需求出发进行详细分析，构建系统用例分析。

3.1 功能需求分析

本系统功能需求分析主要是根据如今疫情局势的前期调研得出，根据调研结果总结系统所需要的功能模块以及能够完成哪些任务。以下主要针对系统的四个模块进行需求分析。

3.1.1 单点登陆模块需求分析

登陆模块作为一个系统的基础模块，之所以将其单独作为一个模块进行需求分析，是因为该系统将采用微服务架构的方式进行，各业务之间通过微服务进行调用，假设每个业务都需要用户登陆作为前提，那么有多少个微服务就需要进行相应次数的登陆操作，这样的功能会降低用户的体验感。

因此，普通的登陆服务已无法适应于当前的系统架构于用户的体验需求。为了让用户能够在登陆界面登录之后，对不同的业务内容进行免登录访问，需要对登录功能模块进行再研究与设计。

由于系统只存在两类参与者：医生和系统管理员。所以登录模块主要针对的是这两类对象。在用户登陆过程中不需要其他基本信息，只需要登陆所需的密码以及登陆相关的信息即可，可以单独抽离作为一个功能模块。良好的用户体验以及登陆数据的安全性将会是登陆模块所考虑的关键要素。

3.1.2 用户管理模块需求分析

医生与系统管理员作为该系统的用户需要有相应的信息管理，方便医疗系统的维护与人员联系。该系统主要针对的是“新冠”肺炎辅助诊断，应用影像学资料作为诊断依据，而在肺部影像部分，通常也只有指定的科室会应用该功能。因此医生的信息不仅仅只是局限于个人的基本信息，其科室的情况也将纳入用户管理功能之中，方便后续能够对医患关系进行更好的管理。同时为进行后续系统维护问题的责任处理，系统管理员也将作为被管理对象，被系统管理员所管理。

3.1.3 智能诊断模块需求分析

本系统主要是为医生提供“新冠”肺炎辅助诊断功能，医生只需要将患者的肺部 X 光片作为输入，以秒级的速度获取到患者是否为“新冠”阳性病例，以达到快速筛查的目的。由于普通肺炎与“新冠”肺炎在临床表现与影像学上差别不大，所以要在短时间内人为做出区分会有些困难，因此系统的诊断准确率与速率要求高且快。当诊断的预测结果出现时，医生能够快速获取信息，查阅影像资料，依据影像资料做出更深入的分析，撰写并生成诊断报告，方便患者后续的医疗程序。

3.1.4 患者信息模块需求分析

患者是作为该系统被诊断的主题对象，其信息量的庞大以及私密性被作为系统设计的重点。医生可以查询患者的历史诊断报告，以方便做出相应的医学判断。同时，系统管理员也可以对患者信息做出相应的管理维护，防止数据信息的丢失。医生还可以对患者进行随访对比，即当前的诊断结果与影像学资料同以往记录进行对比，通过对比，医生可以更好的了解患者的情况。

3.2 系统用例分析

3.2.1 系统参与者

如表 3-1 所示：

表 3-1 系统参与者

角色名称	职责描述	使用功能
系统管理者	管理后台用户	登陆系统，对后台用户信息进行管理，包括患者信息以及医生对应的科室信息
医生	对患者进行医学诊断	登陆系统，根据系统做出的肺炎诊断情况，对患者的影像学资料信息做出详细的检查，然后撰写并生成相应的诊断报告；同时医生可对患者进行随访对比的检查。除对患者的诊断之外，医生还可以对自己的个人信息进行修改。

3.2.2 确定系统用例

1. 系统管理员请求服务的用例

(1) 登陆系统

(2) 后台用户的信息管理

(3) 患者信息管理，包括影像资料的管理，但是无法改动诊断信息

(4) 管理科室信息，包括科室状态

2. 医生请求服务的用例

(1) 登陆系统

(2) 获取“新冠”肺炎辅助诊断结果

(3) 查阅患者肺部影像学资料

(4) 撰写诊断报告

(5) 随访对比

系统用例图如图 3-1 所示：

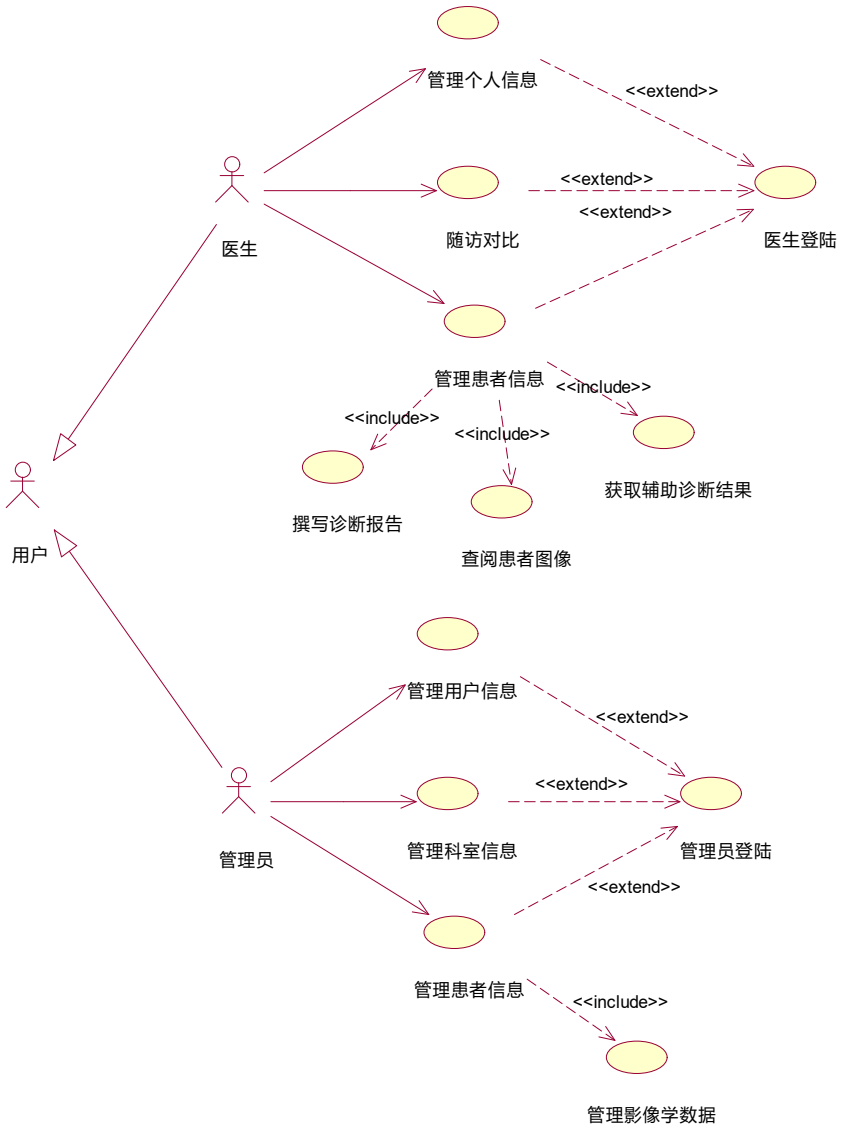

图 3-1 系统用例图

部分用例描述如下表 3-2 和表 3-3 所示:

表 3-2 辅助诊断功能用例描述

用例名称	获取辅助诊断结果
参与者	医生
简要说明	医生通过上传患者的肺部 X 光片，然后通过系统自动诊断，获取肺炎预测结果
前置条件	用户已经在系统成功登录并且拥有患者的肺部影像数据
后置条件	成功获取“新冠”肺炎诊断结果，患者影像诊断结果更新
基本事件流	1.上传患者的肺部 X 光片 2.点击诊断预测的按钮 3.获取诊断结果
扩展点	1a. 该患者当前没有肺部 X 光片
补充说明	1.若上传的不是患者胸部 X 光片，则会出现错误的诊断结果

表 3-3 随访对比功能用例描述

用例名称	随访对比
参与者	医生
简要说明	医生将患者的影像诊断数据同以往的记录进行对比
前置条件	用户已经在系统成功登录并且患者拥有以往的肺部影像数据
后置条件	随访对比后，由医生成功提交随访对比记录
基本事件流	1.选择患者的此次影像诊断记录 2.点击随访对比的按钮 3.界面显示此次影像资料与上一次的影像资料 4.撰写对比记录 5.点击提交随访对比记录
扩展点	2a. 系统显示“不存在上次诊断记录”
补充说明	2.当患者没有上一次诊断记录时，无法进行随访对比

根据上述的功能模块的分析，可以得出以下的系统功能分析图：

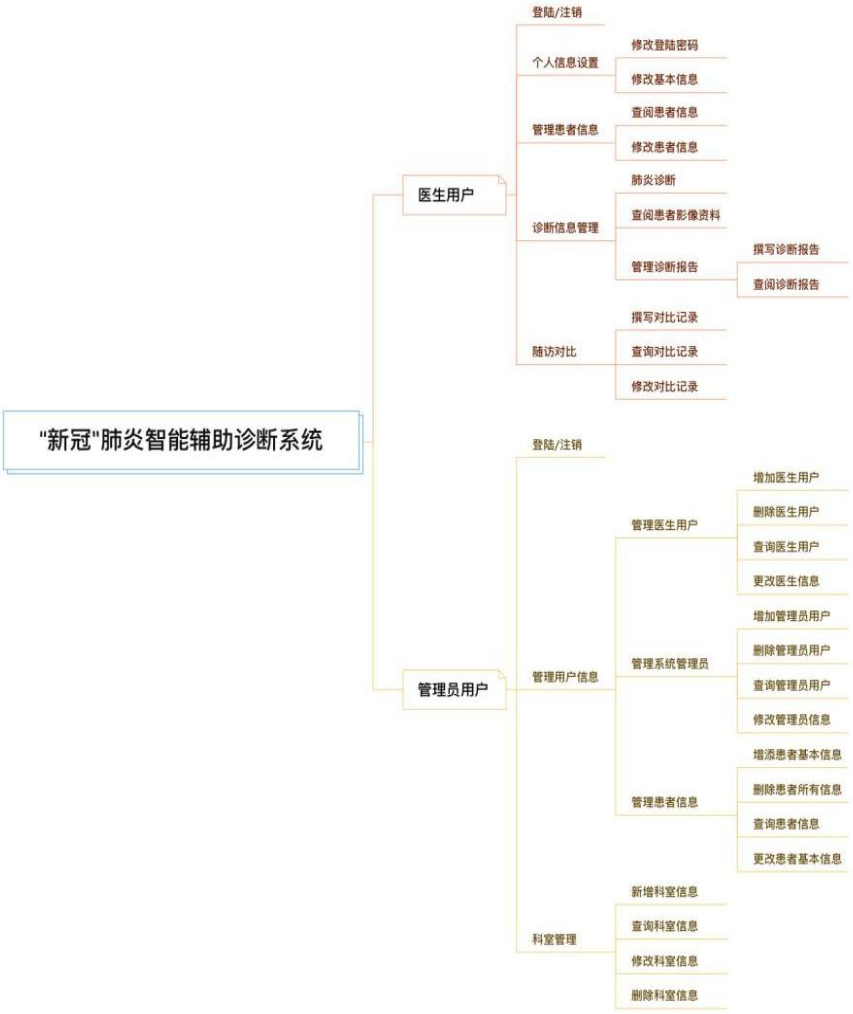

图 3-2 系统功能分析图

3.3 非功能需求分析

3.3.1 性能方面的需求分析

系统的性能是用户体验的直接保障，辅助诊断功能在性能方面是保证快速性、准确性。系统对肺部 X 光片的诊断过程中，要求诊断结果必须迅速且拥有较高的准确率，让医生能够在第一时间获得患者较精确的诊断信息，以达到快速筛查的目的，方便患者进行后续的诊断。当医生对系统作出服务请求时，需要较高的响应要求，以免妨碍医生的工作进程，提高用户体验感。由于本系统是与医院信息相关的系统，故对其数据有一定的输入要求，只有数据格式输入正确，操作才能够被允许，服务器才会响应对应的需求。

3.3.2 可靠性的需求分析

作为医疗相关的系统，在上线之后需要 24 小时甚至全年无休提供服务，因此，要求系统拥有良好的稳定性与可靠性。即在外界环境没有较大变化时，系统能够稳定且持续为用户提供服务。

3.3.3 安全方面的需求分析

医疗数据的安全性与私密性比起其他的数据类型，拥有较高的要求。系统在保证正常运行的情况下，需要做好防止外部侵入与盗取数据的准备，因此系统需要有权限认证，加密处理等作为保护。同时，数据的完整性与可用性也是本系统的要求，在读取方面应做到数据不被污染。

3.4 其它需求分析

除了上述所分析的功能与性能层面的需求，本系统仍需考虑其他方面的需求，主要从系统的可拓展性需求出发。人类活动带来的环境恶化，以及人们自身生活习惯的变化在一定程度上导致新的疾病、病毒层出不穷，就正如新型冠状病毒。为了应对疾病的威胁，满足人们日益增长的健康需求，作为医疗系统，当需求产生时，能够灵活地增加相应的功能。因此，系统的架构设计上需要考虑延展性问题。

4 系统总体设计

上一章是对系统需求各方面的分析，本章节将依据上述分析结果，从技术、业务逻辑、实现功能等角度详细介绍系统各部分的设计。

4.1 系统总体架构设计

依照上述功能模块以及性能方面等需求的剖释，系统需要强大的服务能力、高可用性以及稳定性，所以在系统整体架构方面选用了分布式微服务架构。在第二章中介绍的 SpringCloud 相关理论知识，其中集成了各种微服务功能组件，因此本系统采用 SpringCloud 实现。系统的整体架构设计图如下图 4-1 所示：

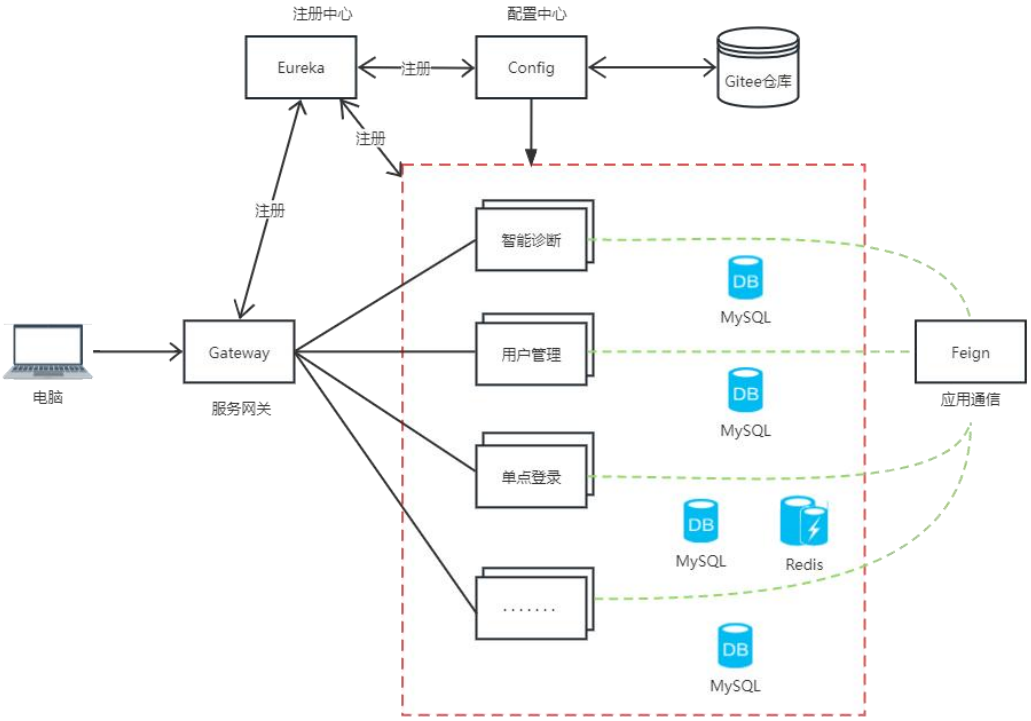

图 4-1 系统整体架构图

该系统的一切请求首先经过服务网关 Gateway，服务网关不负责处理业务，只是按照某种规则，将请求路由到目标服务之中，而请求的地址也从各自不同的地址转变成统一的请求地址。服务网关作为一个单独的微服务，同其他服务一样要注册在注册中心 Eureka 中，方便统一管理。由于各微服务的配置有所不同，考虑到系统的可维护性，采用配置中心 Config 的方式进行配置的统一管理，将所有配置文件放置在远端 Gitee 之中，当有程序变动或者多人维护时，只需要在远端 Gitee 更改相应的配置文件即可，同样的道理，配置中心也将作为一个独立的微服务在 Eureka 服务端进行注册。

在上图的红色虚线框内，主要放置的是系统的业务层以及数据层。业务层主要根据业

务逻辑与功能需求进行划分，如下图 4- 2 所示为系统的功能模块图：

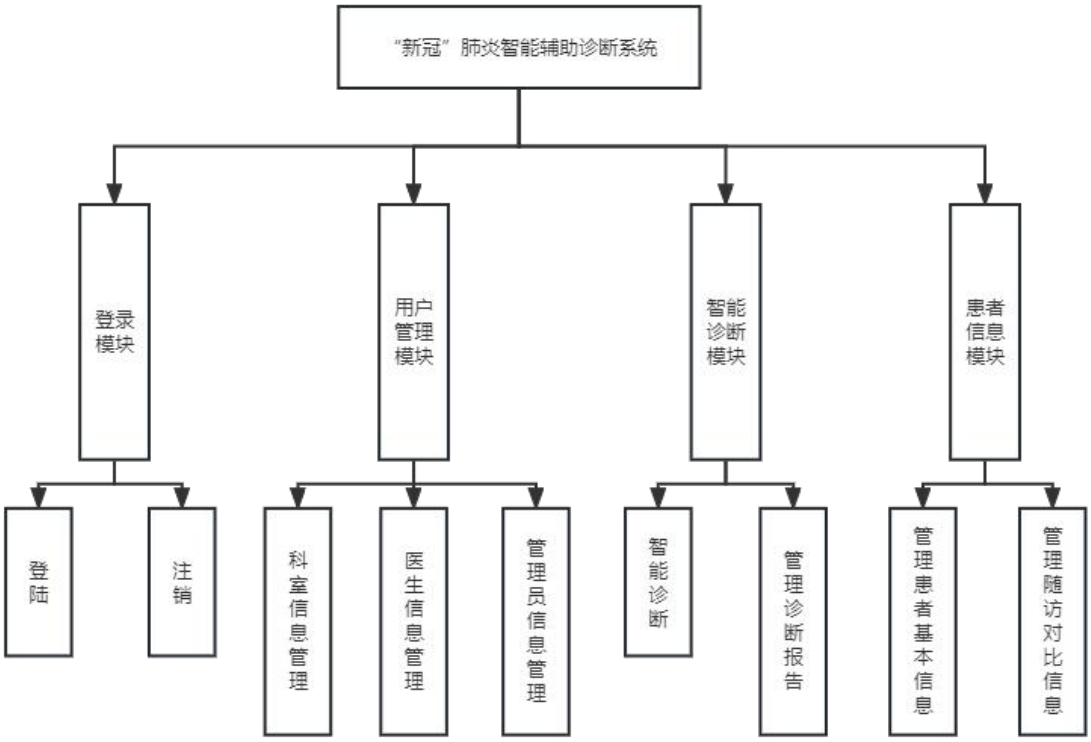

图 4- 2 系统功能模块图

该系统的详细功能将由这四个大的模块展开进行，每个模块作为独立的微服务应用，各自拥有自己的数据库，本系统将运用关系型数据库 MySQL 与非关系型数据库 Redis 作为缓存。因为在单点登录作为独立的应用需要将登录信息存储于缓存中，使数据持久，旨在其他微服务方便获取进行验证登陆。各个微服务之间的应用通信采用 Feign 进行，即某个业务涉及两个甚至多个应用，此时数据之间的调用就可以采用 Feign 的方式进行，而这里的调用并非数据库的直接调用，而是将对外服务的类封装成接口，这样既方便应用间的通信，又保证了数据的安全。

4.2 数据存储设计

由于本系统存储大量的医疗数据，该类数据与其他类型的数据做对比，此数据的完整性和一致性要求较高。而关系型数据库拥有较高的数据逻辑性，因为数据以行和列的方式进行存储，并且表与表之间的关系紧密，故使用关系型数据库能够使数据逻辑更完善与统一。

系统是以微服务架构作为支撑，所以传统的登陆系统已无法满足多应用访问的需求，单点登录成为设计的首选模式，该设计思想不再是已 Session 的机制，而是采用 Token 机

制，将 Token 与用户信息存储在缓存中，因此，该系统采用 Redis 作为缓存来解决问题。

4.2.1 系统 E-R 图

单个实体图如下图所示：

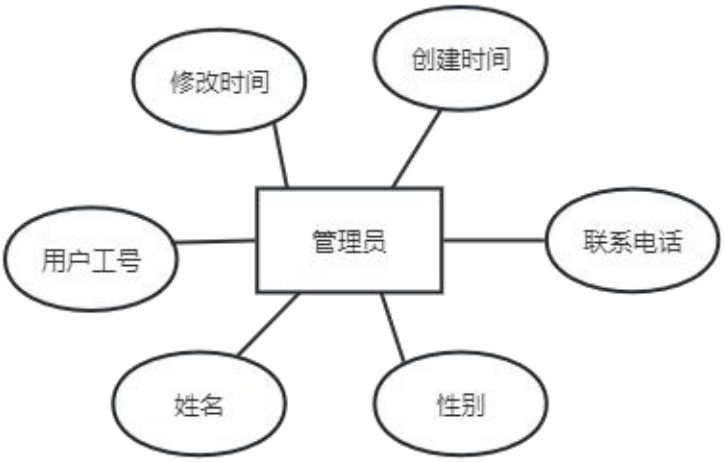

图 4- 3 管理员 E-R 图

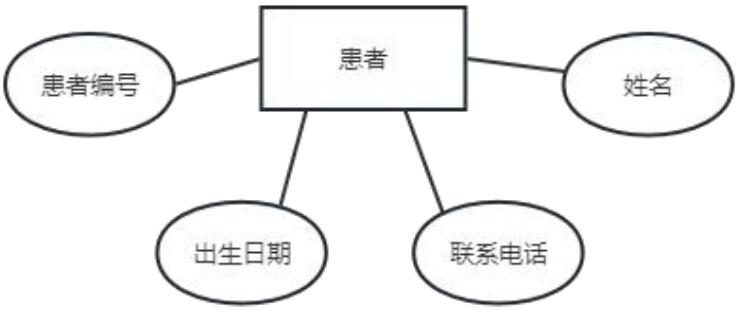

图 4- 4 患者 E-R 图

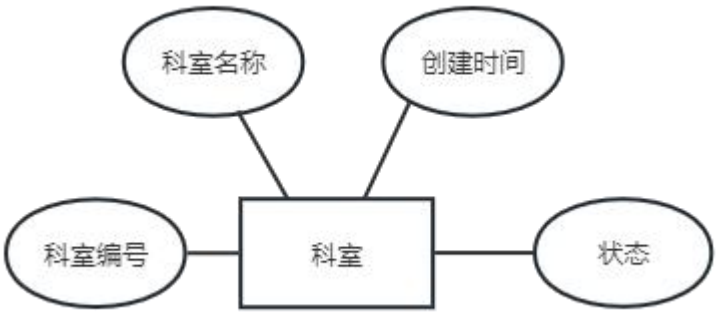

图 4- 5 科室 E-R 图

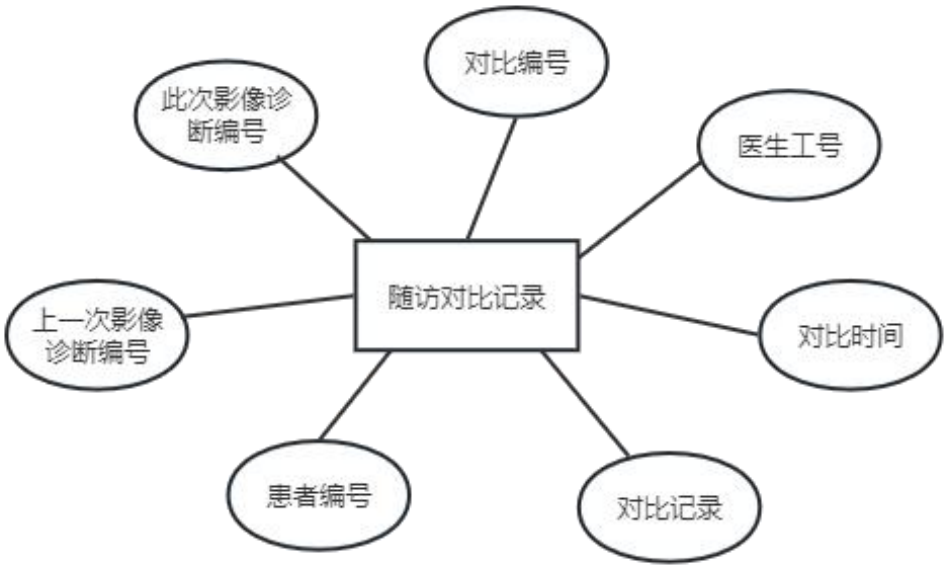

图 4- 6 随访对比记录 E-R 图

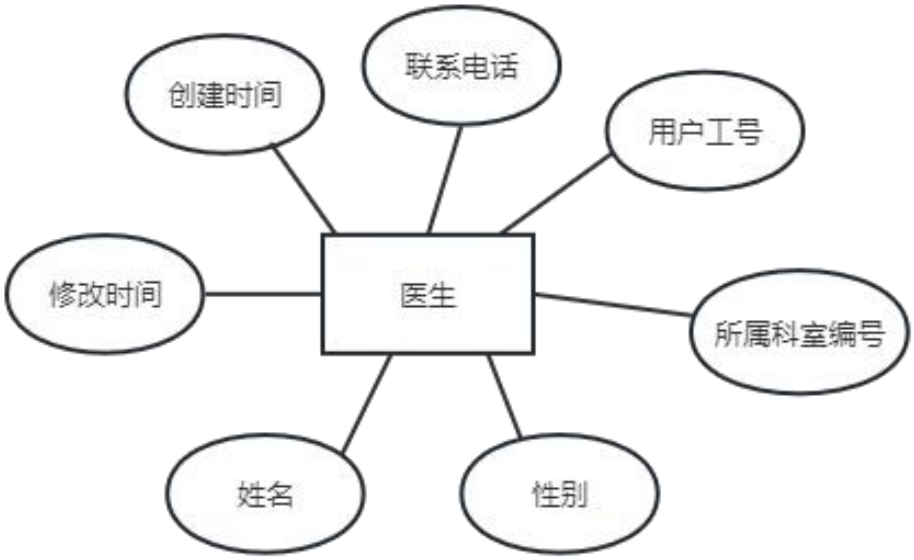

图 4- 7 医生 E-R 图

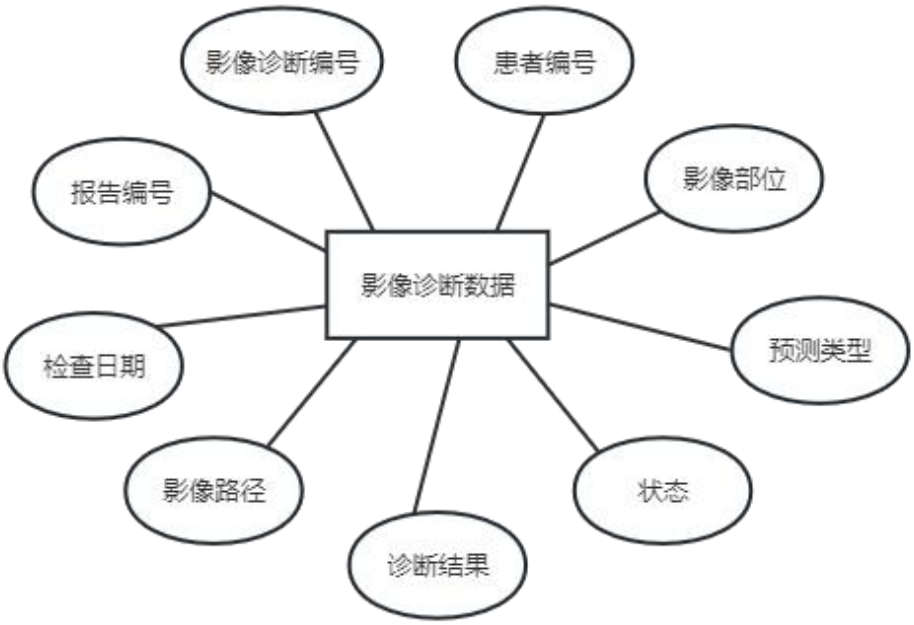

图 4- 8 影像诊断 E-R 图

图 4- 9 诊断报告 E-R 图

系统 E-R 图如下图 4- 11 所示：

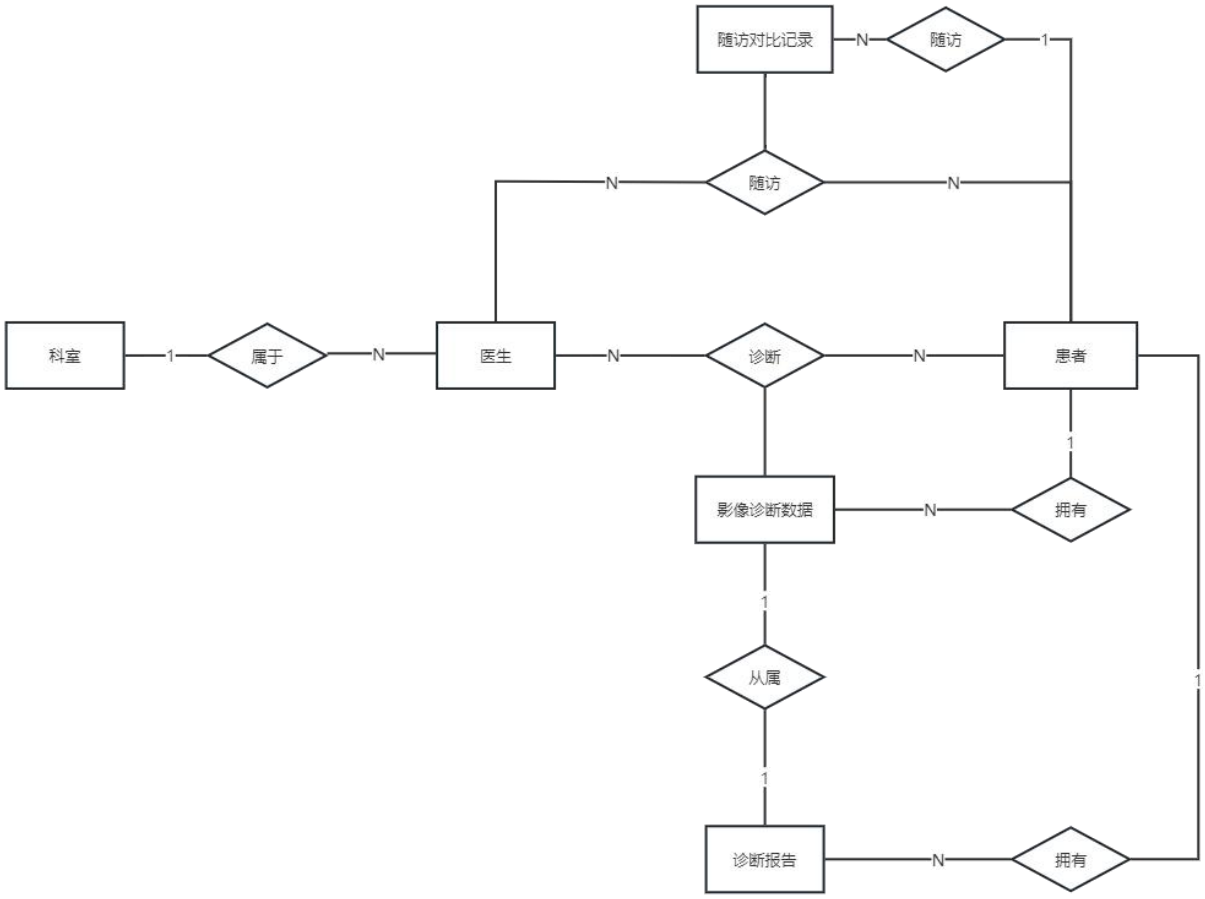

图 4- 10 系统 E-R 图

4.2.2 数据设计

为了增强数据的可读性，减少数据冗余，数据存储仓库选择 MySQL 数据库，保持安全、可延展、规范化、完整一致性等设计原则，设计能够实现快速读写的数据库。由于该系统是应用微服务架构，所以根据模块划分成四个独立的子系统，因此数据库将分为四个，分别是登陆信息数据库（spc_user_login）、用户信息管理数据库（spc_user_manage）、智能诊断信息数据库（spc_ai_diagnose）、患者信息管理数据库（spc_patient_manage）。登录信息数据库主要包含用户信息登陆表（login）；用户信息管理数据库主要包含三个数据库表，分别是医生信息表（doctor）、管理员信息表（manager）以及科室信息表（dept）；在智能诊断信息数据库下含有诊断报告信息表（diag_report）和影像诊断信息表（image_diagnose）；患者信息管理数据库主要存储的是患者基本信息表（patient）和随访对比记录表（compare_repo）。

用户信息登陆表（login）的数据字典如下表 4- 1 所示：

表 4- 1 用户信息登陆表

列名	数据类型	长度	可否为空	说明
work_id	varchar	10	否	主键，用户工号
en_pwd	varchar	64	否	加密密码
role_id	int		否	角色 ID（1 医生/2 管理员）
create_time	timestamp		否	创建时间
correct_time	timestamp			修改时间

医生信息表（doctor）的数据字典如下表 4- 2 所示：

表 4- 2 医生信息表

列名	数据类型	长度	可否为空	说明
work_id	varchar	10	否	主键，用户工号
doctor_name	varchar	10	否	姓名
doctor_gender	int		否	性别（1 女/2 男）
dept_id	int		否	所属科室编号
doctor_tel	varchar	15	否	联系电话
create_time	timestamp		否	创建时间
correct_time	timestamp		否	修改时间

管理员信息表（manager）的数据字典如下表 4- 3 所示：

表 4- 3 管理员信息表

列名	数据类型	长度	可否为空	说明
work_id	varchar	10	否	主键，用户工号
admin_name	varchar	10	否	姓名
admin_gender	int		否	性别（1 女/2 男）
admin_tel	varchar	15	否	联系电话
create_time	timestamp		否	创建时间
correct_time	timestamp		否	修改时间

科室信息表（dept）的数据字典如下表 4- 4 所示：

表 4- 4 科室信息表

列名	数据类型	长度	可否为空	说明
dept_id	int		否	主键自增，科室编号
dept_name	varchar	20	否	科室名称
status	int		否	科室状态(1 就诊中/2 停诊）
create_time	timestamp		否	创建时间

诊断报告信息表（diag_report）的数据字典如下表 4- 5 所示：

表 4- 5 诊断报告信息表

列名	数据类型	长度	可否为空	说明
repo_id	int		否	主键自增，报告编号
content	text		否	报告内容
correct_time	timestamp		否	修改时间
doctor_id	varchar	10	否	医生工号

影像诊断信息表（image_diagnose）的数据字典如下表 4- 6 所示：

表 4- 6 影像诊断信息表

列名	数据类型	长度	可否为空	说明
img_dia_id	int		否	主键，影像诊断编号
patient_id	varchar	15	否	患者编号
img_part	varchar	10	否	影像部位
diag_type	int		否	预测类型(1 新冠/2 普通/3 正常/4 阴影）
diag_result	int		否	诊断结果(1 阴性/2 阳性）
img_url	varchar	30	否	影像路径
check_date	timestamp		否	检查日期
repo_id	int		否	报告编号
status	int		否	状态（1 已阅，2 未阅）

患者基本信息表（patient）的数据字典如下表 4- 7 所示：

表 4- 7 患者基本信息表

列名	数据类型	长度	可否为空	说明
patient_id	varchar	15	否	主键，患者编号
patient_name	varchar	20	否	姓名
birth_date	timestamp		否	出生日期
patient_tel	varchar	15	否	联系电话

随访对比记录表（compare_repo）的数据字典如下表 4- 8 所示：

表 4- 8 随访对比记录表

列名	数据类型	长度	可否为空	说明
Compare_id	int		否	主键自增，对比编号
Patient_id	varchar	15	否	患者编号
Pre_img_id	int		否	上一次影像诊断编号
Now_img_id	int		否	此次影像诊断编号
Com_record	text		否	对比记录
Compare_time	timestamp		否	对比时间
Doctor_id	varchar	10	否	医生工号

5 系统详细设计与实现

上一章是对系统整体的设计，包括架构设计与数据组织。而这一章是基于上述的研究，对其技术做出更详细的策划与实现，并阐述实现过程所遇的问题以及改进方案。

5.1 图像识别算法

本系统的“新冠”肺炎智能辅助诊断功能的实现是基于图像识别的算法，即将患者的肺部 X 光片作为输入，通过算法模型的计算，预测出该患者是否患有“新冠”肺炎，若否，是否属于其他类别的肺炎。因此，该图像识别的算法模型的建立要求拥有一定数量的数据集，并且数据集含有的数据内容需要有正常的肺部 X 光片、患有“新冠”肺炎的患者 X 光片以及其他类型肺炎的 X 光片。

本课题所使用的数据集是来自于 AI Studio 社区当中。该数据集其中包括 3616 个 COVID-19 阳性病例以及 10192 个正常、6012 个肺部浑浊（非 COVID 肺部感染）和 1345 个病毒性肺炎图像。所有图像均采用便携式网络图形（PNG）文件格式，分辨率为 299*299 的像素。综上，此数据集可以用于将“新冠”肺炎病例和其他肺炎病例以及正常的案例进行分类，以加快确诊的速度和效率。

在获取到的数据集中，X 光图像已经被分类放在四个文件夹当中，分别是 COVID（“新冠”肺炎）文件夹、Lung_Opacity（肺部浑浊）文件夹、Normal（正常）文件夹以及 Viral_Pneumonia（病毒性肺炎）文件夹。而图像的标签即为对应文件夹的名称，为了方便记录，将标签从文件夹名称更改为数字 0 至 3，0 代表患者患有“新冠”肺炎，1 代表患者肺部浑浊，2 代表正常，3 代表患者患有病毒性肺炎（非“新冠”肺炎）。

本系统采用 Python 语言构建图像识别的模型，以 PyTorch 作为深度学习框架，Pycharm 作为开发环境。X 光图像的显示将采用 Visdom 实时可视化工具包。以下图 5- 1 为数据集在 Visdom 上的显示结果：

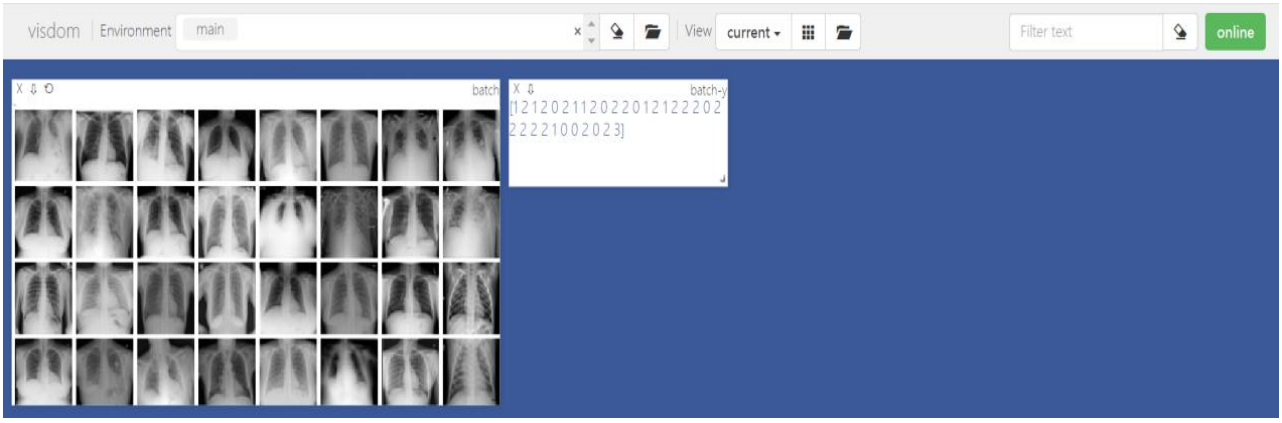

图 5- 1 数据集在 Visdom 上的显示结果

左边窗口中显示的是从数据集随机显示的 32 张 X 光片，右边窗口是图像所对应的标

签。

模型建立最基础的一个步骤就是数据处理。首先，本文将原始数据图像更改为三个通道 RGB 的格式，然后就是将数据进行标准化处理，应用 transform.Normalize()函数。本论文采用标准化处理的原因是考虑到神经网络学习其权重的原理。也就是通过向整个网络的各个权重的矩阵添加其反向传播计算的梯度误差乘以学习率来进行学习。如果不按照比例缩放输入的训练向量，那么每个特征值分布的特征范围将有所不同，因此学习率将导致每个维度的矫正互不相同，且在某个权重维度上会出现过度补偿或者补偿不足。这与期望结果并不相符合，因为权重空间无法达到更好的最大值，处在振荡阶段或者处于缓慢移动的状态。综上，在图像作为神经网络的输入之前进行标准化处理有一定的必要性。

在第二章中所提及的相关理论技术之中，标准化需要有均值和方差，在数据标准化过程中，都采用已经计算好的均值和方差作为参数传入函数之中，本文所采用的均值和方差是已经从 ImageNet 训练集中抽样计算出来的。在数据进行标准化之前，需要将数据先经过 torchvision.transforms.ToTensor，其作用是将数据归一化到[0,1]，transforms.ToTensor（）会把 H*W*C 会变成 C *H *W，即顺叙为 RGB 的形式。如果只是将数据经过 ToTensor 归一化处理，那么实际上的偏置会较大，而模型在初始化的时候偏置为零，这样会导致神经网络收敛比较慢，经过 Normalize 函数之后，可以加快模型的收敛速度。在实验过程中，需要将数据集分为训练集、测试集和验证集，划定的比例为 6：2：2。训练集的数据是用来进行模型的训练，验证集是用来对模型参数每一次训练后的校验，测试集是用在模型训练完成之后准确率以及鲁棒性的验证。

在进行完数据处理后，接下来就是模型的建立。在模型创建之初，采用的是个根据 ResNets18 建立的网络模型。一开始创建此模型，是由于其残差块的优点，创建时借鉴其网络层与设计核心思想，将数据大小为[3,299,299]的图像作为模型的输入，然后经过卷积核大小为 3*3 的卷积层，后经过残差模块，因图像数据大小的原因，第一个残差块通道数为 16 输入，然后在模块里经过卷积层、归一化处理，数据大小成倍增加输出，在总共经过四个残差块后，数据以[256，H，W]的大小输出，最后经过池化层和全连接层输出，将数据分成上述定义的四类。在模型训练之后，最好的模型准确率达到 86.7%，在模型训练完后应用测试集测试的准确率达到 87.3%。其训练过程中应用 Adam 作为优化函数，损失函数应用交叉熵损失函数。模型训练过程中，其损失值变化如下图 5- 2 所示：

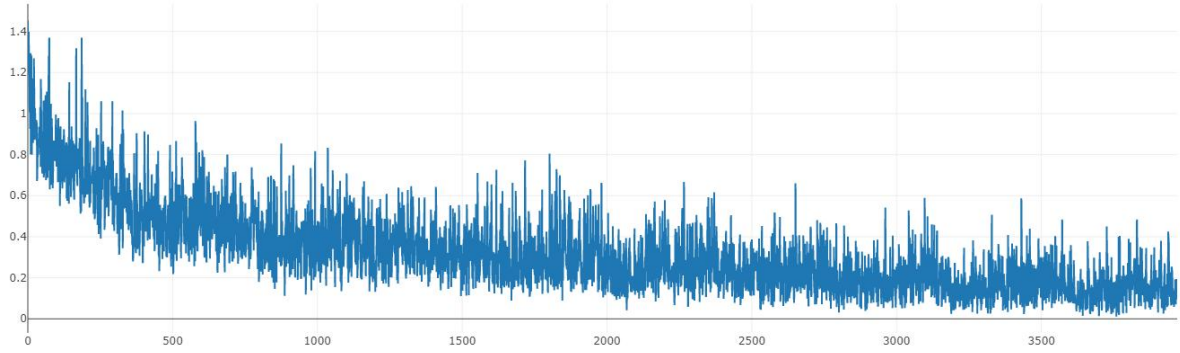

图 5- 2ResNets18 训练中损失值的变化

根据上图以及模型训练结果进行分析可知，模型损失值波动较大，且准确率并没有达到理想要求。故需要对模型算法等方面做出改变。

损失值剧烈震荡的可能原因有如下几点：

(1) 数据集太少

(2) 没有选择合理的数据增强

(3) 损失函数和激活函数设计不合理

(4) 学习率太大

由于在经过调整损失函数和激活函数之后，准确率并没有提升，并且震动仍然较大，因此，模型将从输入的数据入手，即数据集。经过观察，数据集存在的问题有数据分布不均匀，各类别差别较大，正常 X 光片的数量远超于 COVID-19 阳性病例 X 光片的数量，因此当模型进行训练时，会对训练结果产生影像，难以让模型找到最优解。

迁移学习能够很好的解决数据方面的训练问题，因为迁移学习是将已经训练好的模型进行调整，换言之，其模型拥有一定的训练基础，在相似特征上进行再训练。所以在 ResNets18 的基础上引用迁移学习。

其训练过程中的模型的最优准确率是在 92.5%，测试集的准确率是 93.3%。如下图 5-3 所示为模型训练过程中的损失值变动：

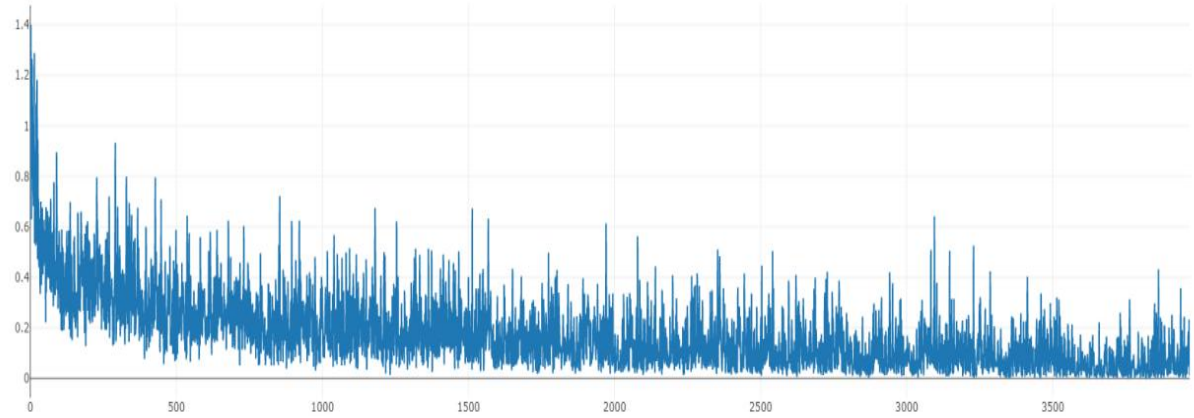

图 5- 3 迁移学习训练中损失值的变化

根据上述结果可以发现，模型的准确率有了较大的提升，但是也在思考准确性如何能够再进一步提高，在复盘整个模型的训练过程以及参数的输入的时候发现，数据在一开始训练的时候进行了数据增强中的随机调整图像的亮度和对比度，而由于初始图像的亮度已经较高，若再进行调高亮度和降低对比度的操作之后，输出的部分图像难以辨别，这样将会影响数据特征提取的过程，因此将数据增强中的亮度和对比度调整去除，重新对模型进行训练后可以发现，准确性有了小幅度的提升。其准确率在训练过程中大致走势如下图 5-4 所示：

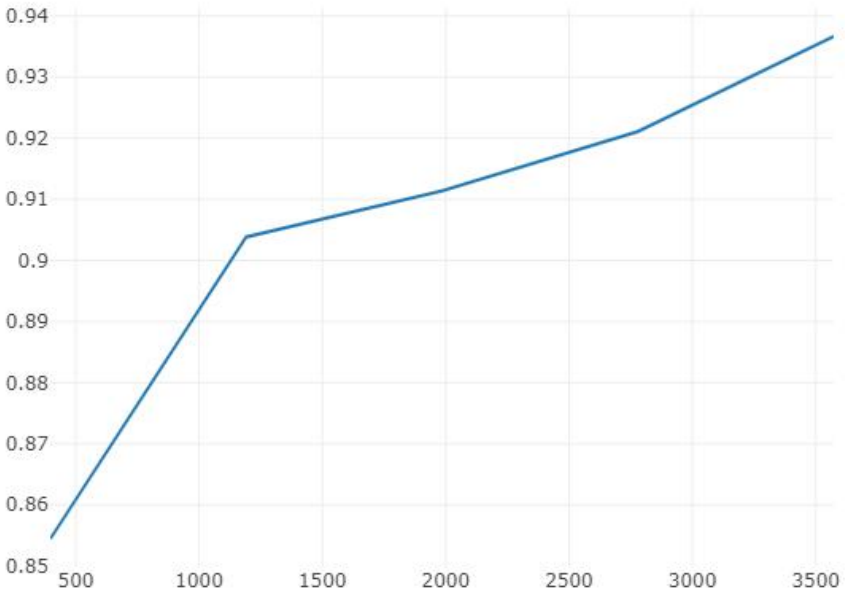

图 5-4 训练中的准确率

改进后的模型准确率能够达到 93.8%。与一开始改进的 ResNets18 网络结构算法相比，准确率有了较大的提升，但是模型中损失函数波动较大的问题仍然是需要进一步做出改进

的地方。

5.2 模块设计

根据微服务架构将系统划分为不同模块作为子系统，拥有独立的数据库。以下是根据上述功能需求的分析进行的子系统详细设计。

5.2.1 单点登陆模块的设计与实现

该功能所针对的对象是所有用户，即医生和系统管理员。登录模块作为安全系数要求较高的模块，需要为整个系统提供服务的前提与保障。并且根据第三章的需求分析中可知，登陆模块的设计上与普通登陆功能有些不同，因为需要考虑到后续功能模块的流畅度。

一般情况下，单系统的登录功能是通过将用户信息存放于 Session 上配合 Cookie 来实现的；在微服务架构中，采用 Token 的身份验证方式实现登陆功能。

Token 机制相较于 Cookie 机制的好处是：Token 不存在跨域访问的问题，服务端在签发 Token 之后保存在缓存之中，等待客户端再次请求资源时，带着 Token 进行验证。Token 的验证与解析也将会节省许多时间。

除了功能性上的设计，该模块还需要在数据安全方面进行研究。用户在登陆过程中只需要输入对应的工号以及预先设置好的密码，验证成功之后就可以对后续功能进行访问，而密码加密处理可以为用户信息盗用做出保障。如果单纯的密码加密解密算法进行密码设置的话，仍存在被破解的风险。本功能采用的是 PasswordEncoder 加密以及验证密码，而此方法的解密处理并不是真正意义上的将密文还原为明文，而是将加密内容与输入的明文内容作生成策略的匹配，因此可以达到密码验证解密的目的。

用户在登录成功之后对后续系统的访问需要权限作为第二重安全屏障，即医生只能访问医生相关功能界面，而系统管理员则只能拥有维护功能。因此当用户在进行登陆操作时，可以将指定带有用户标识放入 Cookie 之中，在后续访问子系统时可以取出，并进行验证。而鉴权的过程应用 Gateway 作为过滤器使用。

在用户登陆成功之后，Cookie 内将拥有 Authorization 的参数，若用户为医生，则其参数值为 doctor；若是管理员用户，则值为 admin。应用 Gateway 作为过滤器，本系统使用全局自定义过滤器，在管理员登陆上进行权限判断，观察 Authorization 参数值是否为 admin，若是则放行。

由于系统是基于微服务架构处理，各个子系统的端口号不同，可能会出现跨域问题，即浏览器禁止请求的发起者与服务端发生跨域 Ajax 请求。解决的方法是进行跨域配置，进行全局的跨域处理。设置的内容包括允许跨域请求的网站地址、允许跨域 Ajax 的请求方式、允许请求中携带的头信息以及是否允许携带 Cookie 等。

5.2.2 智能诊断模块的设计与实现

智能诊断模块主要结合了图像识别的功能，所以其应用对象是医生。

该模块的主要功能有对患者影像进行诊断识别，撰写诊断报告。当医生获取到患者肺部 X 光片时，上传该图像信息，然后进行智能诊断，根据上述所生成的模型，对图片信息进行分类预测，后将诊断结果以及相关的影像诊断内容记录起来，一并提交到该患者的影像诊断表之中，一个患者可以拥有多个影像诊断表，并且在智能诊断之后，医生需根据结果与影像信息进行进一步的诊断，给出医学建议，撰写诊断报告，并提交存储。因此，一个影像诊断信息就会对应一份诊断报告。

该功能的业务流程图如下图 5- 5 所示：

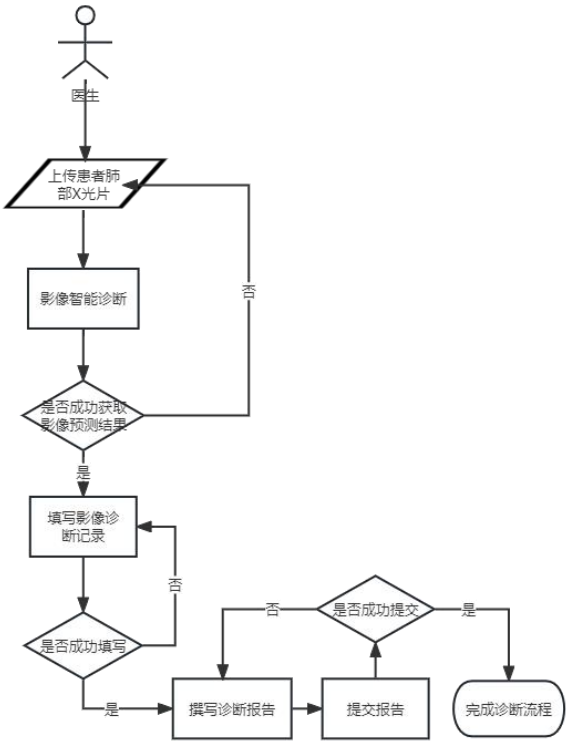

图 5- 5 智能诊断功能的业务流程图

根据上述业务流程图，抽离其中的数据和表单，画出数据流图，如下图 5- 6 所示：

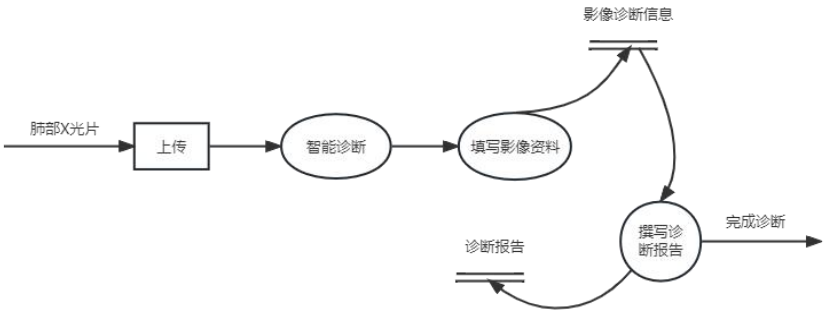

图 5- 6 数据流图

从上述流程中可知道，医生在上传完图片之后，图片需要经过所建立好的算法模型，通过计算后得出计算结果。由于系统架构应用 Java 语言实现，而算法模型是应用 Python 语言建立，因此，需要在 Java 中调用 Python 脚本。本系统将使用 Runtime.getRuntime() 执行 Python 脚本文件。

在实现过程中，应用 Java 调用 Python 有两种形式。一种是以引入 Jython 的 jar 包进行调用；另一种方式是使用 Java 原生的 Runtime 方法。Jython 在编码过程中曾遇到过的问题就是 Python 脚本文件中的第三方库无法使用，因为 Jython 调用的 Python 库资源有限，因此存在部分模型无法找到的错误。而 java.lang.Runtime.exec 方法是以调用外部程序，并重定向外部程序的标准输入、标准输出和标准错误到缓冲池的原理。综上所述，在 Java 中执行带有多种第三方库的脚本文件时，使用 Java 的原生方法更为适合。

在前后端的交互设计上使用的是 Ajax 异步技术，无需等待请求数据完全返回界面方可进行后续的代码。因为在智能诊断过程中会拥有一定的等待时间，且智能诊断结果作为辅助的参考功能，在此等待期间医生可以继续为患者提供诊断服务，达到了节省时间效率的效果。

如下图 5- 7 所示，是该功能模块的界面图：

图 5- 7 智能诊断功能界面图

根据该功能模块的需求以及流程设计，实现图如下图 5- 8 所示：

图 5- 8 智能诊断功能模块实现图

5.2.3 用户管理模块的设计与实现

用户管理模块主要是对医生信息一级系统管理员信息的管理，包括登录信息。因此，作为单独的用户管理模块的子系统，需要调用单点登陆的子系统中的数据库信息，本系统将采用 Feign 进行应用间的通信。由于微服务架构的子系统不直接调用其它系统的数据库，所以与数据库交互的接口会进行数据再封装，使得数据暴露在外的接口只是经过封装的，并不与数据库直接相连。

应用间通信有两种方式，一种是 RestTemplate，另一种是 Feign。RestTemplate 需要每个请求都采用拼接的方式，灵活性高，但是消息封装臃肿。而对于 Feign 来说则不需要进行拼接，只是将 rest 请求进行隐藏，可以便捷地调用 HTTP API。

对某个接口进行了@FeignClient 注解的声明，Feign 就会针对这个接口创建一个动态代理的对象，在调用这个接口的时候，其实就是调用这个接口的代理对象，代理对象根据@FeignClient 注解中 name 的值在服务注册中心找到对应的服务，然后再根据@RequestMapping 等其他注解的映射路径构造出请求的地址，针对这个地址，再从本地实现 HTTP 的远程调用。在调用方地主启动类上，同时需要加上注解@EnableFeignClients 扫描所引入的登陆系统打的包。然后直接在 Controller 中应用注解的形式引入相应的接口类，方便后续的调用。

在该模块实现的过程中遇到过这样一个问题：当模块加入 Feign 等注解之后，Controller 层的所有请求都会出现 404 的错误，并且在控制台上并未出现相关的错误信息。因此在错误排查上有一定的困难。

被调用模块被打包后依赖于用户管理模块，在调用过程中，该调用的接口类也作为第三方类文件。在用户管理模块中发现该文件的接口类的@FeignClient 并未作为关键字进行标记，所以问题的产生将可能出现在此处。经过一系列详细的排查后发现，被调用的接口类中的@FeignClient 注解应用的是 Netflix 的 Feign 包，但是在用户管理模块中的

@EnableFeignClients 应用的是 openfeign 包。即使 pom 文件中的依赖都是应用 openfeign 包下的依赖，但是由于版本号是 RELEASE，属于修正版，在被调用模块仍然应用 Netflix 包下的依赖，但是在打包到用户管理模块时，会更新成 3.1.4 版本，因此在该模块中无法应用 Netflix 包下的注解，自然@FeignClient 将不会产生其相应的功能。

问题发现之后，将对被调用模块的 pom 文件进行修改。剔除原始版本的 openfeign 依赖中的 feign-core 文件，然后重新导入更新后版本号为 3.1.4 的 feign-core 文件。更改后对模块进行打包重新依赖于用户管理模块，因此将能保证用户管理模块与被调用模块的注解应用的是同一个包下的文件。

问题解决之后，该模块能够正常调用登录模块中的信息，如下图 5-9 所示为在 Postman 上管理员基本信息调用管理员的登录信息的返回结果。

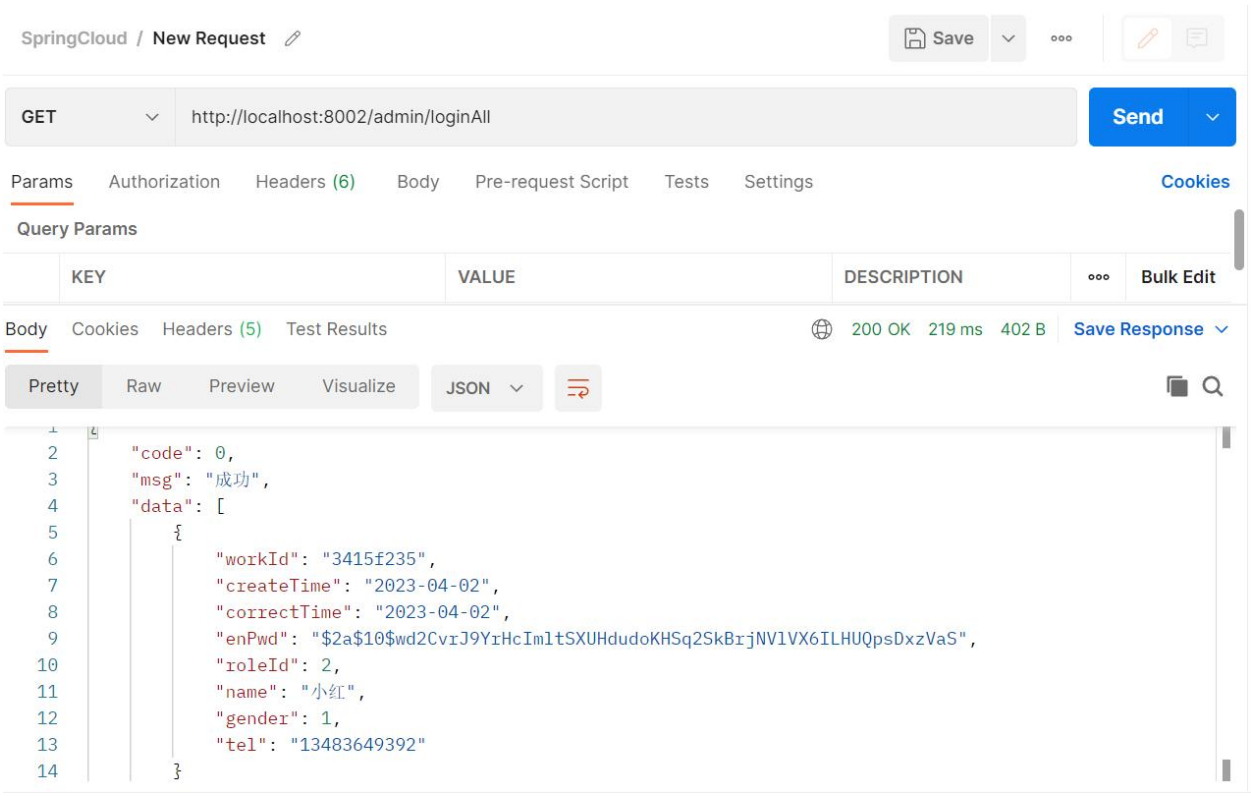

图 5-9 管理员基本信息调用登陆信息模块

5.2.4 患者信息模块的设计与实现

患者信息管理除了基本信息管理，还包括医生对患者的随访对比记录。

根据患者信息模块的需求分析与总体设计可知，患者信息模块需要调用智能诊断模块的数据库信息，即医生在浏览患者信息时，可以查看该患者的每一条详细的患者诊断记录，而且每一条诊断记录都可以被详细查看，包括其影像学信息以及诊断报告。同时，医生也可以查看某一位患者的每一条随访记录，并且查阅其对应的详细信息。在观察患者图像的时候，医生可以选择是否进行随访对比，以检查患者的病情发展。

查询患者信息是根据患者编号进行查询，查询的内容可以包括所有与诊断相关的信息，因此，该模块同样需要调用其他子系统的数据，采用 Feign 实现。由于系统的各个子系统

都采用多模块方式开发，所以在子系统通信中的依赖处理，只需要将系统对外提供服务的模块进行打包即可。在本系统开发中，每一个子系统都分成三个模块，分别是 Client，Common，Server。Client 是作为对外暴露的接口模块，Feign 接口存放于此；Server 模块内放置所有的业务逻辑，Common 用于存储公用的对象。在实现过程中，将 Client 模块打包后作为依赖放入要调用的系统中。对于患者信息模块，需要将智能诊断模块中的诊断信息接口进行打包后依赖于患者信息模块，应用 Feign 的相关注解实现系统间的通信。

如下图 5- 10 所示为患者信息模块的主界面：

图 5- 10 患者信息模块的主界面

6 系统测试

本章将讲述对于“新冠”肺炎智能诊断系统的测试，一方面需要对智能诊断算法的测试，另一方面，因为该系统是基于微服务架构，因此对系统功能架构的检测也十分有必要。以下就是对系统各功能进行测试，以及测试结果的分析。

6.1 测试环境

本计算机的硬件配置：Windows 系统，8 核 CPU，16GB 内存。

该系统在测试过程中应用的软件：

(1) 图像识别算法：Pycharm 开发环境，Visdom 可视化工具

(2) 微服务架构：IntelliJ IDEA 2020.3.2 x64 开发环境，Postman 接口调试工具，Junit 测试工具。

6.2 测试方法

6.2.1 图像识别算法的测试

依照第五章中对图像识别算法的实现，需要对其准确性进行进行测试，随机应用一定数量的肺部 X 光片作为模型的输入，测试模型的预测结果是否与真实情况吻合，通过 Visdom 工具可视化，观察结果。如下图 6- 1 所示为算法测试结果：

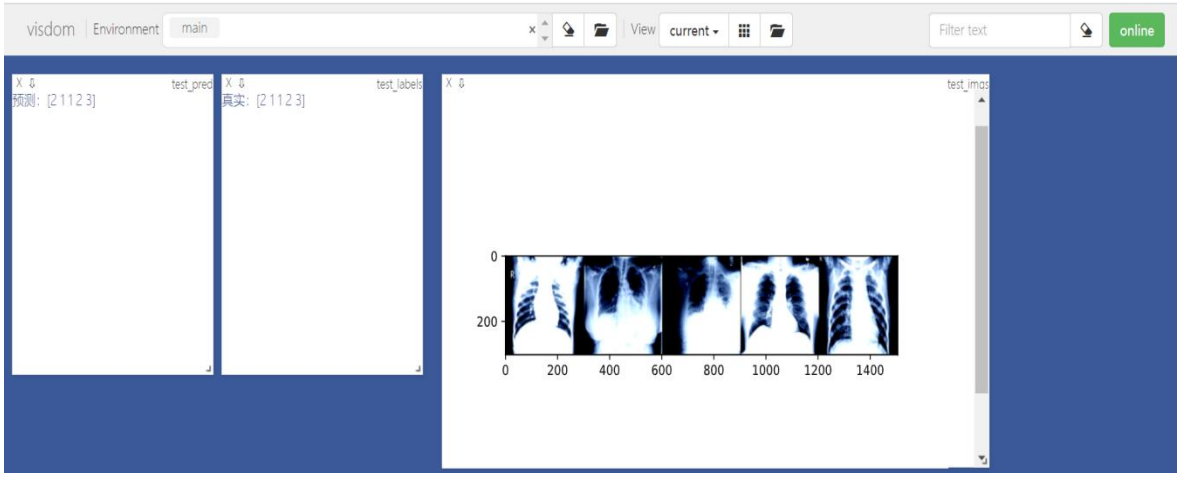

图 6- 1 图像识别测试结果

6.2.2 系统部分功能的测试

由于系统应用功能较多，这里就以登录模块功能为例，进行测试。

测试的过程主要分为以下三个步骤：

(1) 设置测试数据

使用等价类方法进行黑盒测试，如下表 6- 1 所示：

表 6- 1 等价类划分

输入条件	有效等价类	编号	无效等价类	编号
用户名	用户名存在	1	用户名不存在	3
			空	4
密码	密码正确	2	密码错误	5
			空	6

结合错误推测法设计测试用例，如下表 6- 2 所示：

表 6- 2 用例设计

测试用例编号	输入数据		预期结果	覆盖等价类
	用户名	密码		
1	2d710184	123456	登陆成功	1，2
2	2d710184	1234567890	登陆失败，显示提示信息	5
3	2d710184		登陆失败，显示提示信息	6
4	2d710184	12345	登陆失败，显示提示信息	5
5	2d71018	123456	登陆失败，显示提示信息	3
6		123456	登陆失败，显示提示信息	4
7			登陆失败，显示提示信息	4，6

(2) 在测试中调用被测试模块

根据上述的用例设计，将数据作为参数放进接口中进行测试，在 Postman 中进行，除了需要观察测试结果是否符合预期结果，还应该查看 Cookie 中是否带有 Token 并且带有用户标识，在缓存 Redis 中查看用户信息是否已经存入。

(3) 判断返回的结果是否符合预期

根据上述用例的测试，其测试结果都与预期结果保持一致，并且能够在 Postman 中查看到对应的 Cookie，而且在缓存中也能够找到用户信息。如图所示，是用户登陆成功之后在 Postman 中显示的 Cookie 值与缓存内容。如图 6- 2 和图 6- 3 所示：

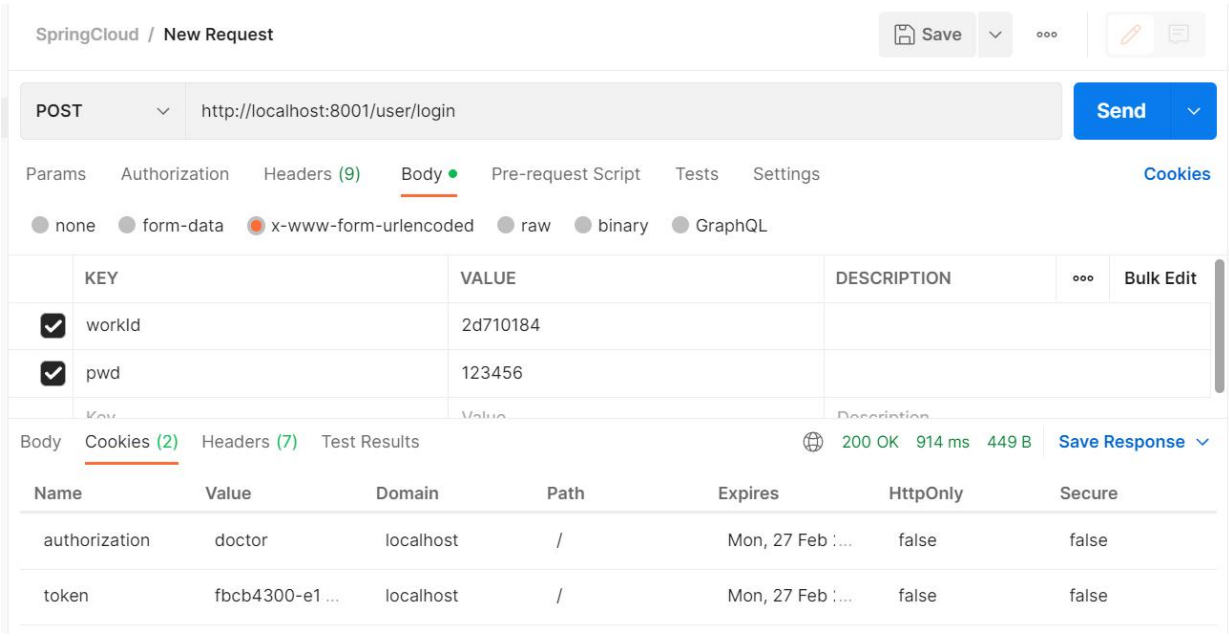

图 6- 2 Postman 工具 Cookie 测试

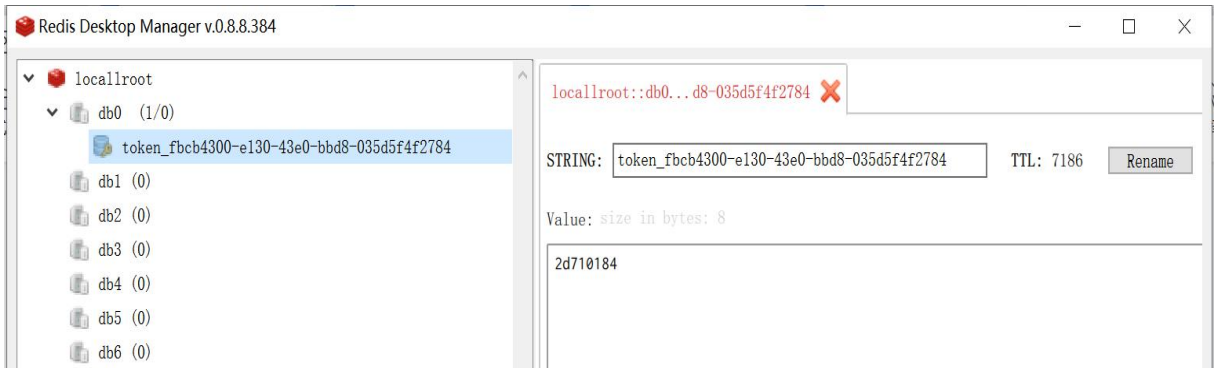

图 6- 3 缓存内容显示

7 总结与展望

本章将主要对基于 SpringCloud 和深度学习的“新冠”肺炎智能辅助诊断系统在研究、设计及实现过程中应用或涉及的理论技术方面进行总结，并结合实际情况，对系统未来的发展、研究方向进行进一步的讨论。

7.1 总结

在基于微服务和深度学习的“新冠”肺炎智能辅助诊断系统的研究、设计与实现过程中，应用并且涉及了许多当今主流的技术与知识理论。

首先在图像识别方面，根据模型的多次筛选与调整，选择准确率相对较高的方法训练模型。满足了智能诊断功能所要求的诊断正确率要高的问题。

其次根据需求分析以及对系统进行整体的微服务功能架构的设计，由于传统的单体架构已经不能很好的为用户提供大量的需求，而且在系统功能延展方面受到诸多阻碍，因此，选择微服务作为系统架构，满足其高可用性、稳定性、可拓展性。

根据分析结果与设计内容，对系统各功能模块进行实现。

最后，借助测试工具，对系统进行全面的测试。

7.2 展望

随着影像学和人工智能技术的发展，疫情形势的不断变化，阶段式需求的激增使得智能诊断技术在现实生活中的可用性增强。在可预见的未来，该系统在医疗市场也将会拥有很大的发展空间和发展潜力。

对于本系统来说，在今后长期运营的过程中，随着用户数目的增加，以及系统需求的扩大，一部分内容可采用更加先进的技术。例如在微服务架构下，数据库以及缓存将会面临更大的压力，可以采用中间件的形式进行。以及在图像识别算法方面，可以对算法模型做出改进，以应对更多的需求与变幻莫测的疾病。

总之，在今后长期的运营服务中，系统还有很多值得优化、改进的地方。力求为用户提供高质量、便捷易用的智能诊断服务，为国家医疗事业贡献自己的一份力量。

参考文献

[1] World Health Organization.Novel Coronavirus(2019-nCoV) Situation Report-22 [EB/OL]. Coronavirus disease (COVID-19) Weekly Epidemiological Updates and Monthly Operational Updates, 2020-02-17.https://www.who.int/docs/default-source/coronaviruse/situation-reports/20200217-sitrep-28-covid-19.pdf?sfvrsn=a19cf2ad_2.

[2] 专家解读：奥密克戎变异株毒力明显减弱 重症病例多表现为基础病加重.广东市场监管.2022-12-06.https://www.thepaper.cn/newsDetail_forward_21049766.

[3] 林思瑶. 基层智能辅助诊断系统应用与发展研究[D].华中科技大学,2020.DOI:10.27157/d.cnki.ghzku.2020.006201.

[4] “新冠”肺炎CT影像AI定性辅助诊断系统助力打赢攻坚战“疫”.赵广娜.2020-02-23.https://www.cn-healthcare.com/article/20200223/content-531300.html?appfrom=jkj.

[5] 王筝,袁慧书,赵亮.PACS/RIS在大型综合医院放射科的应用[J].医疗卫生装备,2011,32(09):48-50.

[6] 左其群. 基于多模态的肺炎辅助诊断系统的设计与实现[D].江苏科技大学,2021.DOI:10.27171/d.cnki.ghdcc.2021.000426.

[7] 2017 中国AI英雄风云榜技术创新人物候选人之何恺明.网易智能.2017-11-16.https://www.163.com/tech/article/D3CMBKU30009987P.html.

[8] 唐进民.深度学习之PyTorch实战计算机视觉[M].背景：电子工业出版社，2018.6.

致　谢

行文于此，落笔为终，本科四年的生涯也即将结束。这四年来目光所及之处，皆是回忆。我度过了人生中最青春的年华，总有万般不舍，但仍心怀感激。

首先，我要感谢我的家人，感谢他们对我二十多年的悉心照顾，让我衣食无忧，快乐健康的成长。感谢他们对我每次决定给予的鼓励与支持，他们的爱我会牢记于心，砥砺前行。

其次我要感谢我的指导老师郭晓燕，诲人不倦，郭晓燕老师治学严谨，有着丰富的学科知识以及认真的工作态度，让我在学习和做人方面都受益匪浅。在整个论文的定题、开题等过程中，老师细心审查，耐心为我修正错误，提出方向。我将牢记老师的教诲，奋力拼搏，超越自己。饮其流者怀其源，学其成时念吾师。

再者，在这四年的大学生涯中，感谢每一位同学的帮助，在我失意时给予鼓励与安慰。感恩知己，愿你们前程似锦，以梦为马，不负韶华。

感谢吉林大学珠海学院对我的栽培，感谢学校里面的每一位教师，感谢这四年里老师们的倾囊相授，感谢给予我一次又一次的锻炼机会。

在论文撰写过程中，感谢各种参考文献，使我的文章内容更加充实与饱满，也因为有前人的研究，我才有幸能够站在科学巨人的肩膀上看世界。

最后，我要感谢我自己，感谢那个不放弃梦想，努力追逐的自己，在追梦路上有许多的阻碍，但是未到终局，又焉知生死，只有不放弃，坚持自己，放手拼搏，才能让自己更加靠近理想。

终有一别，但是我们来日方长。