分类号：TP391　　U D C：D10621-408-(2023)4761-0
密　级：公 开　　编 号：2018075054

# 成 都 信 息 工 程 大 学
# 学 位 论 文

## 基于视觉与文本技术的看图写话系统设计与实现

论 文 作 者 姓 名：陈衍汀
申 请 学 位 专 业：机器人工程
申 请 学 位 类 别：工学学士
指导教师姓名(职称)：王伟（讲师）
论 文 提 交 日 期：2023 年 5 月 17 日

# 基于视觉与文本技术的看图写话系统设计与实现

**摘要：**看图写话是根据图片情景写出描述语句的技能，这项技能对人类来说是比较容易的，但对机器来说具有巨大的挑战性。尽管目前的看图写话技术还处于比较早期的阶段，但是这种技术未来会有很大的应用空间。例如，能够实现看图写话的人工智能助手可以很好地协助视力困难患者了解网上图片中的信息，或者看他所拍摄的图片帮助他认识自己所处的环境，甚至能够协助医学工作者更好地理解医学图像照片。看图写话技术可以让计算机具备对现实视觉环境的认知和推理能力，也意味着计算机的理解水平将迈上全新的台阶。因此，基于以上看图写话的实际应用和对未来的人工智能发展趋势的推测，看图写话系统的设计与实现具有较大的实际意义。

本课题的主要工作是运用视觉与文本技术，识别图片并输出一段描述性语言，其主要研究图片含有一到三个意象的场景。围绕这个工作，本课题设计和开发含有三个模块的系统，分别是前端模块、后端模块以及算法模块。首先，前端模块包括首页、登录、注册、提交图片、查询结果页面。其次，后端模块包括对前端各个页面进行的业务逻辑处理，以及和算法模块进行交互，并将处理结果返回给前端。最后，算法模块需要自建微服务，将算法处理服务暴露给后端服务器，并可以随时替换不同的算法，设计为有极大的扩展性和可替换性。

本系统达到了如下的指标，设计了界面优美的页面，并且页面性能流畅，体验效果好；设计了一个响应速度在毫秒级别的后端系统，方便及时响应前端和其他系统请求；算法描述图片准确率达到了 70%以上，并且处理能力强，处理速度较块，控制在秒级左右。

**关键词：**视觉与文本技术；看图写话；视觉对话；

# A Visual and Text Technology-Based Image-To-Descriptive System: Designing and Executing.

**Abstract：** Reading pictures and writing sentences is a skill that involves writing descriptive sentences based on the context of the picture. This skill is relatively easy for humans, but it poses a huge challenge for machines. Although the current technology of reading pictures and writing words is still in its early stages, this technology will have great potential for application in the future. For example, an artificial intelligence assistant capable of reading and writing words can effectively assist visually impaired patients in understanding the information in online images, or help them understand their environment by looking at the images they take, and even assist medical workers in better understanding medical images and photos. The technology of reading pictures and writing words can enable computers to have the ability to recognize and reason about the real visual environment, and it also means that the understanding level of computers will take a new level. Therefore, based on the practical application of the above picture reading and speech writing system and the speculation of the future development trend of artificial intelligence, the design and implementation of the picture reading and speech writing system have great practical significance.

The main task of this project is to use visual and textual techniques to identify images and output a descriptive language. Its main focus is on studying scenes where images contain one to three images. Based on this work, this project designs and develops a system consisting of three modules, namely the front-end module, back-end module, and algorithm module. Firstly, the front-end module includes the homepage, login, registration, image submission, and query results page. Secondly, the backend module includes business logic processing for each page in the front-end, as well as interaction with the algorithm module, and returning the processing results to the front-end. Finally, the algorithm module needs to build its own microservices, expose the algorithm processing services to the backend server, and can replace different algorithms at any time, designed to have great scalability and replaceability.

This system has achieved the following indicators: a page with a beautiful interface has been designed, and the page performance is smooth and the experience effect is good; Designed a backend system with a response speed of milliseconds to facilitate timely response to front-end and other system requests; The algorithm achieves an accuracy rate of over 70% in describing images, and has strong processing power, relatively fast processing speed, and is controlled around the second level.

**Key words：** Visual and Text Technology；Speaking through pictures；Visual dialogue；

目录

论文总页数：49 页

# 引言

选题背景：看图写话是根据图片情景写出对应的描述语句，这项对人类来说比较容易的技能，但对于机器来说就比较有挑战性。尽管目前的看图写语技术仍处于早期阶段，但它正在迅速发展，并且具备巨大的应用潜力，可以为未来带来更多的便利。通过使用看图写话系统将图片翻译为文字，如果再通过语音技术转换为声音，视觉功能障碍的人士不仅可以轻松地阅读和理解图片上的信息，而且还可以通过拍摄身边的照片来更好地了解自己所在的外界环境;通过使用这种技术，医生们能够更准确地理解医学影像图像。

AR/VR 技术为人类带来了一种全新的交流方式，通过使用看图写话系统，在虚拟现实中的 AI 可以与用户一起进入一个全新的视觉环境，并且可以使用自然语言来交流和分享周围的信息。看图写话的系统，可以让机器在视觉方面具备了对现实世界的认知和推理能力，这也意味着人工智能的对现实世界理解能力将迈上新的台阶[1]。

国内外研究现状：Facebook 和 Instagram 提供的帮助失明和视力残疾人士的服务，可以自动以语言对画面上的图像作出说明。目前虽然已经有一些生成文本的方案，但还远达不到“信达雅”的标准，甚至还会存在一些荒谬的描述。但不可否认的是，随着技术发展和标注数据越来越丰富，模型效果相比以前已经有了比较大的提升[2]。

2014 年，谷歌在该领域的经典论文 Show and Tell: A Neural Image Caption Generator 中将一个图像作为输入，和经过训练的最大化可能的文本序列来充分描述图像。这篇论文提出了一种基于神经网络和概率的模型结构，并且详细描述了它的功能和性能。其大致内容是：“近年来统计与机器翻译领域取得了巨大进步，人们提出了一种强大的语言序列模型，它可以通过最大化输入翻译的概率来获得最佳结果，并且可以通过端到端方法来输入语言。这个方法利用了一个递归的神经网络，将可变长输入，编码到一个固定的长度向量中，并通过这个表示方法来编码从它到期望的输出语言”[3]。

2019 年，阿里 AI 凭借其卓越的性能，在全球第二届视觉交流竞赛 Visual Dialogue Challenge 上，战胜微软、首尔大学等十支强大的对手，最终获得了荣耀的胜利。由全球视觉领域的顶级国际学术会议 CVPR 所举办的全球视觉交流比赛，是目前在视觉交流领域中的最权威的国际赛事之一。AI 在观察了数万张图片之后，能够准确地回答出人们对其中任何内容的提问，这是本次比赛的一大亮点。经过一场激烈的比赛，阿里 AI 凭借 74%的精确度脱颖而出，其识别精度提高幅度超越了上一届的水平，达到了惊人的 16.82%。尽管人类的精确度达到了 64%，但是这个数字仍然远远低于其他人工智能[4]。

AI 是一个复杂的问题，AI 技术可以帮助我们理解和表达环境信息，并将其转换成人类可以理解的文本数据。然而，传统的人工智能技术在这方面的表现相对较弱，因此难以应对“这只猫旁边的男生穿了什么颜色的衣服”这样的挑战。AI“递归探索对话模型”是阿里 AI 的一项重大突破，它将图像识别、关系推理和自然语言理解三个模块完美结合，可以有效模拟人类的思维，从而更好地理解不同场景下的情况；此外，“递归探索对话模型”还可以准确识别图片中的数据，推断出图片中的事件，并且能够模拟语境，从而更好地理解人类提出的实际问题，从而更好地掌握实际的知识，更好地应对复杂的社会环境。通过深入研究，我们可以得出更加合理的结论[5]。

目前，看图写话系统前沿技术还在处于探索阶段，训练出的模型大多用于比赛，在实际应用中，专门看图写话系统软件还是非常稀少的，并没有把前沿的研究真正应用到实际中。

研究意义：针对以上所提到的社会现状和各种问题，本课题旨在设计并开发一款基于视觉与文本技术的看图写话系统，能够将用户提交到的图片翻译成一段描述性语言，再通过语音转换等工具可以让视觉障碍人士理解网络上图片的信息，或者是用来接入其他 AI 系统使其具有对图片的语言理解能力，还可以用来给想想体验人工智能的用户一个体验平台，这对方便残障人士的生活和扩展 AI 对现实世界的现实理解能力有极大的意义。

研究目的：本课题旨在通过对前端、后端、算法进行学习、理解和运用，完成看图写话系统的设计和实现，通过前端给用户提供一个可以注册和登录的平台，使其方便的体验到 AI 看图写话的功能，同时使用后端接收和处理信息，将前端界面和算法服务器紧密联合起来，使得前端提交的图片经过后端可以传递给算法服务器处理，再将算法服务器处理结果经过后端返回给前端。看图写话系统实质只有将图片转为描述性语言的功能，但其可作为中间平台，接入其他系统，使得其他系统功能更加完善。

本文旨在探讨如何利用视觉和文本技术来构建一个看图写话系统，并对其进行全面的架构设计、算法模块的搭建、网页前端页面的设计、以及后端业务逻辑的处理等方面进行深入的分析和探讨。章节安排如下：

第一章：介绍了整个系统需要完成的性能指标，以及系统的总体设计和系统的各个功能模块。

第二章：介绍了整个算法功能模块的理论知识和搭建步骤，包括算法原理、算法的使用方法以及算法处理的结果分析。

第三章：介绍了后端功能模块的理论知识和搭建步骤，包括后端架构设计和服务器的搭建过程和结果。

第四章：介绍了前端功能模块，包括前端界面效果图以及和后端的交互逻辑

第五章：介绍了整个系统的实验结果以及看图写话算法准确率分析。

# 1 系统总体方案设计

## 1.1 主要设计功能及指标

为了满足识别图片生成一段描述性文字的需求，本课题的主要目的是设计看图写话系统，此系统应该实现以下功能及指标：

（1）界面优美性能流畅的前端。可以和用户进行人机交互，并尽量优化页面效果，使得页面优美，性能流畅。并且可以在算法处理时间较长时提示用户，缓解用户焦虑的感觉。

（2）业务逻辑响应快的后端。可以处理用户发送的请求，可以提交图片到服务器上，后面优化到可以批量提交图片。提升请求性能，减小响应时间。在提交图片上达到 50 毫秒响应时间就可异步返回，而调用算法接口则需一分钟左右的处理时间。

（3）可灵活配置不同算法的算法处理模块。可以将后端传递过来的图片进行算法识别，输出一段描述性文字并返回到前端，并且描述成功准确率达到 70%以上。

## 1.2 总体技术路线

基于视觉与文本技术的看图写话系统主要有以下三个重要模块：算法模块，后端模块，前端模块，系统架构图下面如图 1.1 所示。

（1）算法模块：该模块可以灵活配置实现看图写话的算法，主要以微服务的形式暴露不同的接口来实现配置不同的算法。现在主要是两种算法，第一种是组合算法，即通过使用语义分割算法处理图片从而得到关键词，再使用关键词进行文本续写，从而得到一段描述性文字。第二种是直接使用图像字幕算法，输入一张图片直接得到图片的描述性文字。

（2）后端模块：后端模块负责接收用户的请求，同时将算法模块微服务化，按需调用。后端主要有注册登录功能，需要记录用户信息，判断用户是否有权限使用算法模块功能。在进行算法处理时会使用异步操作将提交图片和算法处理进行异步，在提交图片成功后，同时启动算法处理，但不等待算法处理完，直接返回给用户提交成功，让用户去等待查询结果，以此达到响应快速的目的。

（3）前端模块：前端模块负责用户进行交互，人性化的显示各个算法模块的处理结果，使得用户容易使用看图写话系统。前端模型需要编写首页、注册登录页面、看图写话系统界面、提交图片界面、查询结果界面。首页负责展示看图写话系统的介绍，以及导航到注册登录界面，在登陆后进入看图写话系统中，然后在提交图片页面提交图片，在查询结果页面等待算法的处理结果的返回。

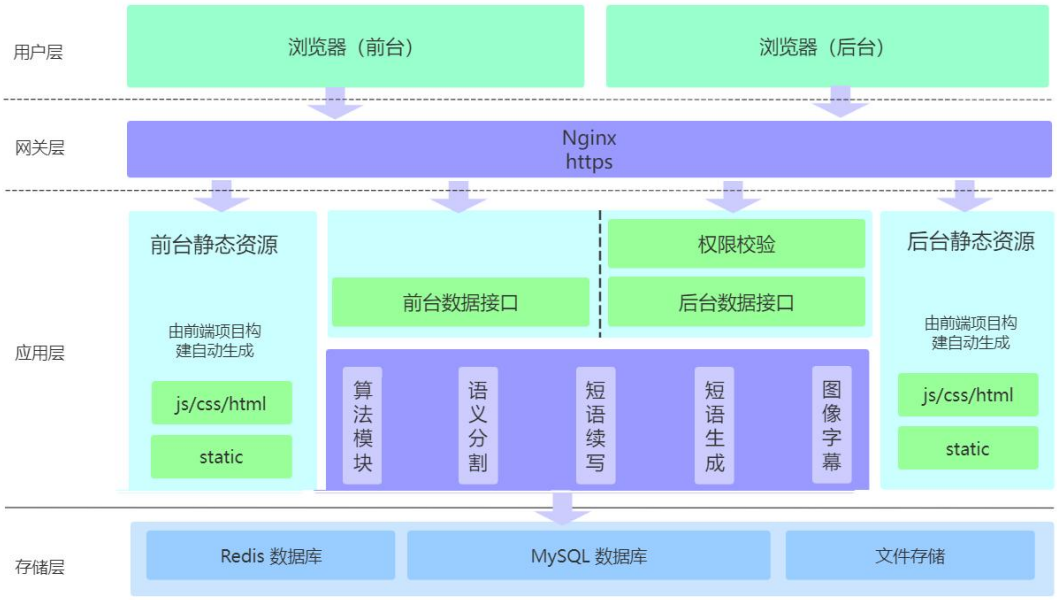



图 1.1 系统架构图

工作重点如下：

（1）算法微服务的搭建，包括四种人工智能算法，图像语义分割，关键词生成短语，短语续写以及图像字幕算法，前面三种算法组合实现看图写话功能，最后一种算法则单独实现看图写话功能，可以对比两种实现方式的优劣。将算法以为微服务的形式暴露给后端服务器，从而使得后端服务器可以按需调用各种算法功能，提高系统的可扩展性。

（2）后端服务器的搭建，后端需要使用负载均衡，服务网关等技术，需要接入数据库存储用户和算法数据，接入缓存提高系统性能。

（3）前端模块的编写，使用 UI 框架搭建出优美的页面，并实现看图写话的基本功能，可以实时显示各个算法模块的处理结果，与用户进行人机交互，方便用户微调算法执行结果，从而达到最终看图写话的效果。

## 1.3 本章小结

本章重点讲述了基于视觉与文本技术的看图写话系统完成的功能指标，以及此系统的整体框架，通过对本章节的阅读，可以对此系统的整体设计和工作流程有一个大致的了解，并且因为此系统采用了模块化设计，也能够对此系统的各部分功能模块有一个初步的了解，后续将对各部分功能模块进行详细讲解。

# 2 看图写话系统的算法部分介绍和使用

看图写话系统的算法部分包括了四大部分，可以用两种不同的方式实现看图写话。这四大部分，分别是图像语义分割，关键词生成短语，短语续写以及图像字幕算法，前面三种算法组合实现看图写话功能，最后一种算法则单独实现看图写话功能。第一种实现方案是使用语义分割提取出图片的元素，得到关键字，再通过关键词生成短语算法生成一段短语，最后通过短语续写算法把短语完善。第二种实现方案是直接使用图像字幕算法，输入图像得到一段描述性语言。

## 2.1 对图片进行语义分割提取关键词

### 2.1.1 语义分割是什么及意义

语义分离是计算机视觉学习中一项重要的任务，它要求我们将视觉系统划分为不同的语义或解释类别，以便更好地理解和解释真实世界中的信息。在当今的电脑视觉系统中，语义分割是一个非常重要的组成部分。与目标检测和识别不同，语义分割技术可以将图像分割成像素级别。这个工具可以根据图片和视频的不同特征，将其划分成多个独立的部分。像图片识别以及物体分析一样，语义分割有助于人们对图片有更为详尽的认识。这些知识对于包括自动驾驶、人工智能和图像搜索引擎等诸多方面都是十分关键的。

### 2.1.2 语义分割基本流程

语义分割有三个流程，图片预处理、分类、后处理。图片预处理的作用是优化图片使其可以更好的训练，为分类做准备。分类的作用是将图片中的各个元素归类到其属于的种类中，这样就可以得到元素的语义了。后处理的作用是优化算法效果使其分类的更准确，减少误分类和少分类的可能[6]。

（1）图片预处理：图片预处理是为了使图片更好的投入模型训练。当内容数量太大时，就会出现内存满的情况，造成训练错误，此时就需要对图片进行预处理，可以对图片的分辨率进行缩小或者将图片分割为小图像。图片提前预处理主要是为了使图片更好的去训练，所以当原本图片中出现了过于模糊等问题，那就必须作一些预处理操作。也可以通过各种图像处理库，对原图片亮度、对比度、锐度等方面进行提升。通过预先分割图像数据，我们可以提前进行测量和检验。通常来说，在代码实现阶段，我们就可以将图像分割成不同的部分。当然，如果图像数据分布表现得非常规律，我们也可以自动将它们分为检测集和验证集。

（2）分类：使用卷积神经网络分类。最近热门的语义划分框架，通常是使用卷积神经网络（CNN）给一个图像划分一种的分类标签。卷积层技术能够更高效的捕获目标图像中的部分信息，并且可以通过层级的形式将多个功能类似的单元信息嵌套到一起，这样 CNN 就能够尝试着获得更多的结构信息了。如果使用一

系列卷积技术捕获目标图像的更复杂信息，CNN 就能够将一个图像的信息编码成紧凑表征了。如果我们想要将单个图像信息转换为标签，我们就需要将 CNN 编码器扩展为一个更加复杂的编码器-解码器结构。采用折叠层和池化层的结构，编码器能够有效地压缩目标图像的大小，并将它们转换成更精细的信息。通过解码器，可以收集特征，并将其转换为矩阵，从而在执行上进行采样，从而使得特征空间的长度得到增加，从而提高特征的空间宽度。在特定状态下，解码器的中间程序也可以使用调优译码器。最后，解码器得到了一个可以表示为原始特征图像标签的数组。

（3）后处理：CNN 在许多语义分割结构中被用来最大限度地减少交叉熵损失。这个目标函数可以用来比较预期的像素概率分布与现实情况之间的差距，从而更好地掌握各种类型的特征。然而，从语义上看，交叉熵损失的效果并不尽如人意。由于交叉熵损失会导致每一个像素的值都发生变化，因此，这种情况下，它不会有利于将图形中的像素对齐。尽管交叉熵损失可以在图像中提供一定的信息，但由于无法引入更先进的机制，因此对于交叉熵最小化的标签分析结果往往缺乏完整性和准确性，而且还需要进一步的研究来证实其可靠性。CNN 的原始标识通常缺失一些重要的信息，因此，对于图片中的一些细节，可能无法准确地进行标注，也无法与周围的图像进行匹配。要想消除这种不连贯的状态，我们需要采取一种更加流畅的策略。通过对图像进行分析，我们可以确保目标的位置与其周边的每个像素的值保持一致，从而实现对目标的准确描述。为了有效地处理这些问题，一些系统采用了条件随机场化技术（CRF），以确保目标原始像素与周围图像的一致性，从而提取出 CNN 的所有标签。

下面图 2.1 是条件随机场的示意图，条件随机场是由随机变量值组成的，在这个情况下，绿色顶点 Xi 代表了给定图像的 CNN 标签，而黄色顶点 Yi 代表了给定图像的真实类别标签。通过将两种图像分别编码到边缘，我们可以用蓝色来区分它们的真实类型，并且可以探究它们之间的相互影响。CNN 的红色指针代表着它与给定图像的预期结果与实际结果之间的相互影响。通过考虑所有相互依存的因素，条件势能估计可以被描述为一类基于两个完全一致的随机变量的函数。也因此，当相邻图像的实际目标标签数量相同时，对第一类目标导向的条件势能估计效果较好。通过 CNN 算法，即使目标标签被掩盖，也能够通过机率分配获取可见的图像信息。采用 CRF 技术来提取有效的标记，需要进行多次交互式实验，以深入理解图像模型的各种参数。接下来，我们将通过训练来提高概率 P（Y 1,Y2，…X1, X2, Yn,...通过将 Xn 的值最大化，我们实现了对参数的优化。CRF 图像可以用来提取原始数据中的关键信息。事实上，CRF 图像的边完全相互联系，因此，无论它们的位置相隔多远，它们的边仍然能够被有效地共享。然而，由于这张图有数十亿条边，因此从数学角度来看，它无法被精确地处理。CRF

架构强调采用有效的近似技术来实现有效的解决方案[7]。

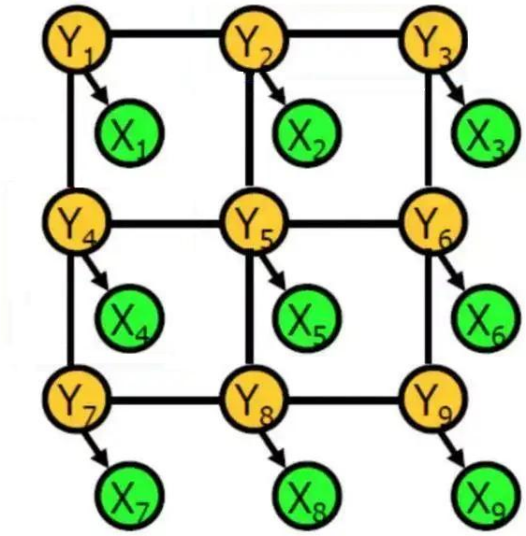



图 2.1 条件随机场

### 2.1.3 MaskFormer 模型

MaskFormer 模型基于 ADE20k 语义分割进行训练。MaskFormer 通过预测一组掩码和相应的标签以相同的范式处理实例、语义和全景分割。

现代方法通常将语义分割公式化为每像素分类任务，而实例级分割则使用另一种掩码分类来处理。随着掩码分类技术的普及，我们可以利用相似的模型、损耗和训练算法，有效地将语义与实例进行细致的区分，从而获得更优的结果[8]。

MaskFormer 即是根据这一观察提出的，图 2.2 即是 MaskFormer 模型，MaskFormer 是一个简单的掩码分类模型，它预测一组二进制掩码，每个掩码都与单个全局类标签预测相关联。总体而言，MaskFormer 所提出的基于掩码分类的方法简化了语义和全景分割任务的有效方法，并显示了优异的经验结果。

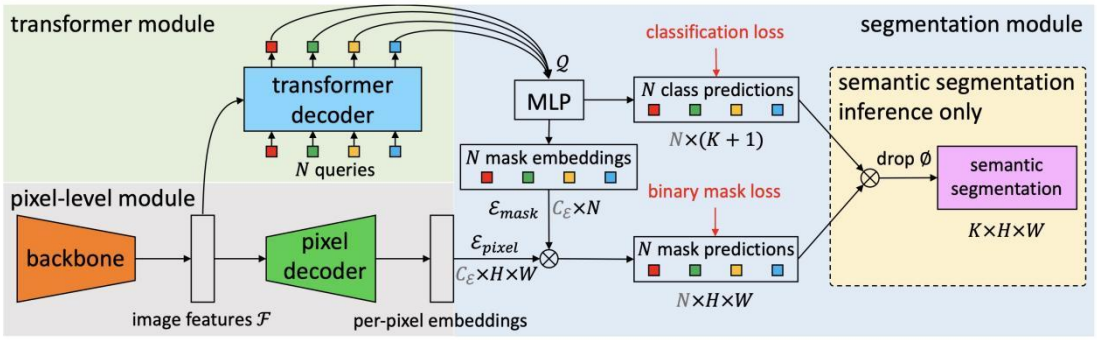



图 2.2 MaskFormer 模型

MaskFormer 模型由三个模块组成，分别是像素级模块(pixel-level module)、转换器模块(transformer module)、分段模块(segmentation module)。

像素级模块负责将原图通过骨干网络转为特征图，该特征图有两个作用，第

一个作用是进入转换器模块的转换解码器中解码，第二个作用是进入像素解码器，最后进行每像素嵌入操作。

变流器功能主要是接收从像素级模块传过来的特征图像，再经过转换后解码器上传到 MLP 层，MLP 有多个感知机，包括输入与输出层，在它们之间还可能有几个输入与输出层，所以一个单纯的 MLP 就有几个输入与输出层，是三层的结合，而多个感知机层和底层之间也是完全连通的，所以 MLP 中所有的数据也就是各个层次间的连接权重和偏置。

分段模块中 MLP 在这里主要实现掩码嵌入和分类预测。掩码嵌入的结果会反馈到每个像素嵌入操作中，将反馈后的结果进行掩码预测。将掩码预测后的结果与分类预测进行随机失活操作，最后进行语义分段推理，从而得到语义分割图。

MaskFormer 模型使用步骤如下：

（1）安装 Anaconda

（2）打开 Anaconda Prompt 终端

（3）创建新的 conda 环境

```
conda create --name chenyanting_env python=3.8
--channelhttps://mirrors.tuna.tsinghua.edu.cn/anaconda/pkgs/free/
```

使用如上命令创建了名为 chenyanting_env 的虚拟环境，python 版本为 3.8，并使用清华源加速下载

（4）搭建 cuda 和 pytorch 适配环境方便 GPU 训练

```
conda install torchvision==0.13.1  cudatoolkit=11.3 torchaudio==0.12.1
pytorch==1.12.1  -c pytorch
```

（5）pip 安装必备包

```
pip install transformers -i https://pypi.tuna.tsinghua.edu.cn/simple
```

（6）使用模型进行进行语义分割

```
from transformers import MaskFormerFeatureExtractor, MaskFormerForInstanceSegmentati
onfrom PIL import Imageimport requests

url = "https://huggingface.co/datasets/hf-internal-testing/fixtures_ade20k/resolve/m
ain/ADE_val_00000001.jpg"

image = Image.open(requests.get(url, stream=True).raw)

feature_extractor = MaskFormerFeatureExtractor.from_pretrained("facebook/maskformer-
swin-large-ade")

inputs = feature_extractor(images=image, return_tensors="pt")

model = MaskFormerForInstanceSegmentation.from_pretrained("facebook/maskformer-swin-
large-ade")

outputs = model(**inputs)# model predicts class_queries_logits of shape `(batch_size,
 num_queries)`# and masks_queries_logits of shape `(batch_size, num_queries, height,
 width)`

class_queries_logits = outputs.class_queries_logits

masks_queries_logits = outputs.masks_queries_logits

# you can pass them to feature_extractor for postprocessing# we refer to the demo not
ebooks for visualization (see "Resources" section in the MaskFormer docs)
```

```
predicted_semantic_map = feature_extractor.post_process_semantic_segmentation(output
s, target_sizes=[image.size[::-1]])[0]
```

（7）结果展示：下面图 2.3 是语义分割处理之前的原图，可以看到这张图的元素整体是一个房子，地面上有草地，左下角还有树，上半部分就是天空，也算是元素比较多的一张图。

**图 2.3　语义分割处理之前的原图**

下面图 2.4 是经过语义分割处理后的图，可以看到图片中的相同种类的部分都被同一种颜色上色，下面表 1 是分割得到的语义如下所示，有建筑物、草地、道路、树、天空以及墙，可以看出该算法模型的语义分割效果非常好，不止把我们之前的对图片的描述全部预测出来，还预测了一些我们没有想到的，但是却符合逻辑的元素，后面我们就可以通过这些元素语义得到关键词并续写成一段短语。

**表 1　分割得到的语义**

| 序号 | 标签 | 语义 |
|---|---|---|
| 1 | building | 建筑物 |
| 2 | grass | 草地 |
| 3 | road | 道路 |
| 4 | tree | 树 |

| | | |
|---|---|---|
| 5 | sky | 天空 |
| 6 | wall | 天空 |

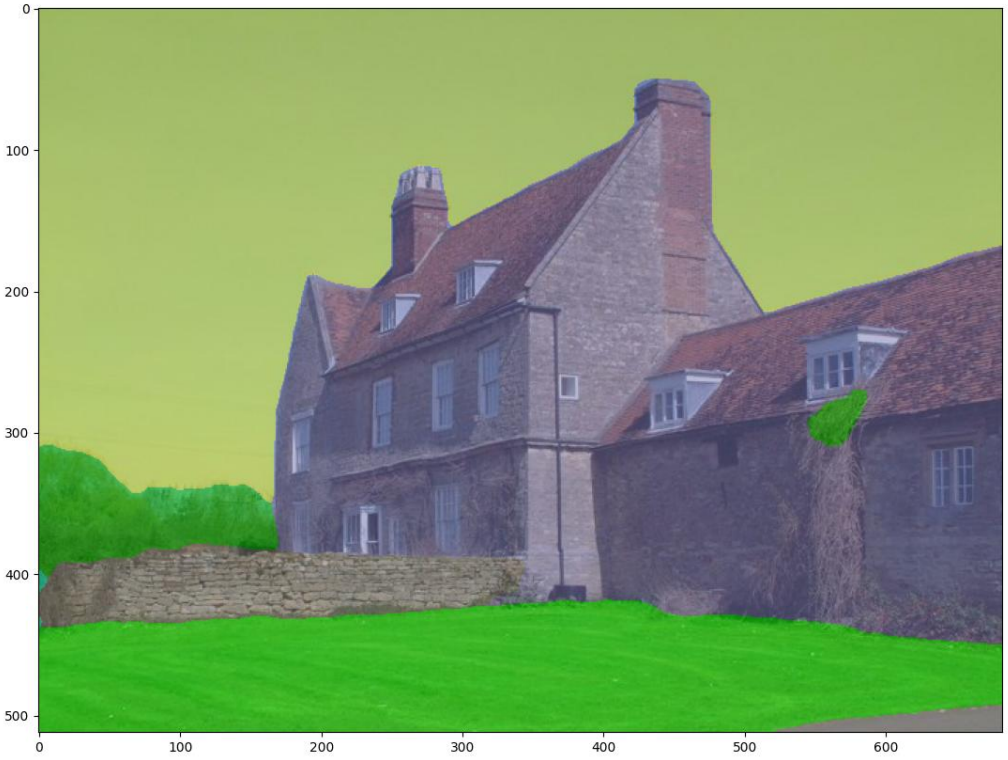



图 2.4　语义分割后的图

通过上面的使用，可以看出 MaskFormer 模型的语义分割准确率十分高，不但精准识别语义，而且能在像素级进行分割。MaskFormer 总共可以识别百余种分类，这为我们看图写话系统提供了很大的发挥空间。

## 2.2 根据关键字生成一段短语

### 2.2.1 自然语言处理

自然语言处理（Natural Language Processing, NLP）的出现极大地推动了语言学与人工智能的进步，它提供了一种全新的、极具弹性的语言处理模式，大大提高了语言处理的效率，极大地改善了语言处理的准确性与可靠性。这是一种新的技术，可以有效地将人类大脑与计算机之间的信息交流转化为自然语言，从而提升交流的效率。自然语言处理是一种有效的沟通方式，它可以帮助人们更好地理解和应用自然语言，从而更有效地实现数据的传输和共享。通过利用人类对自然语言的处理能力，使计算机能够理解其中的内容。自从人类开始研究机器翻译功能，我们就开始了对自然语言处理的深入研究。虽然自然语言处理涉及到许多

不同的领域，如语音、句子、意思和用途，但它的核心目标依旧是通过使用词典、词频学和上下文语义学等工具，来解决一些特定的问题，使它们能够被组织成一个个独立的、拥有丰富语义信息的小型单元[9]。

### 2.2.2 自然语言处理存在的问题

内容的有效界定。生活中语句间的词语往往是无法孤立出现的，必须把语言中的各种词汇加以交叉联系才可以表现出一定的意思，如果构成特定的短语，词汇之间也会产生相应的划分关系。如果缺乏合理的定义，内容就会变成模棱两可，无法加以有效的理解。比如“他就背着妈妈和妹妹悄悄地出去玩了”。若没有正确界定“和”的含义，则可能会导致母亲和妹妹之间毫无关联，甚至连母亲本身也无法确认他们是否与其他小朋友一起出游。

消歧和模糊性。词汇或者句式在各种情景下的使用常常具有多种意义，很容易形成模糊的定义甚至是不同的观点，比如高山流水这种词汇具有多重意义，既能够描述大自然，又能够表示二者之间的关联，或者是表达音乐的美妙，但是自然语言处理必须根据前后的内容加以划分，从中去掉歧义或者模糊性，表现出真正的含义[10]。

语言行为与计划。句子的含义通常不仅仅局限于字面意义；“你能把盐递过来吗”中，一个优秀的回答应该是将盐递给对方；在大多数情况下，“能”可能会被视为不合适的回应，尽管“太远了我拿不到”的观点也许能够被接受。再比如“去年，由于该门课程未能开设，因此有多少学生未能通过考试？回答“去年没开这门课”的效果要好于回答“没人没通过”的效果。

### 2.2.3 GENIUS 模型

GENIUS 模型，是一种 CLM(conditional text generation)预训练模型，能根据你给定的一个草稿，包含你要表达的关键信息，可以是词、短语、短句，来生成一段完整流畅的文本。

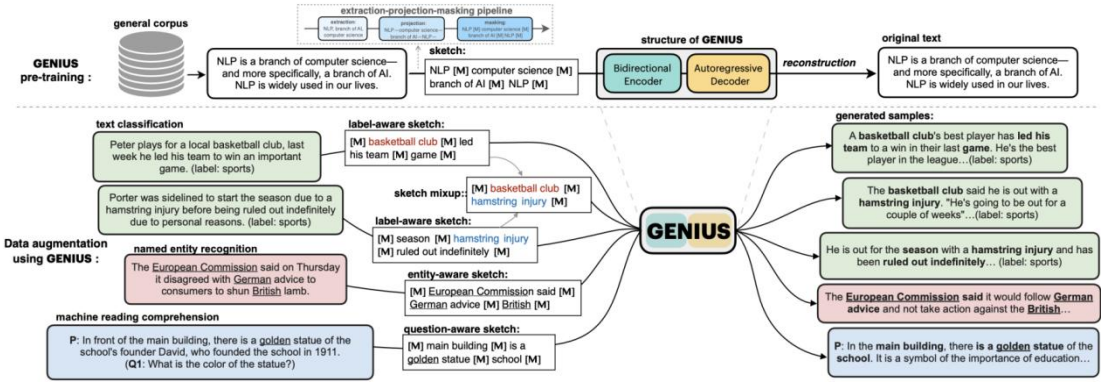



图 2.5 GENIUS 模型处理过程

图 2.5 就是 GENIUS 模型处理过程，GENIUS 模型使用一种由草稿到全文的重构的任务进行预训练，其中采用了一种极端的选择性掩盖的策略，因为相比同

类算法如 BERT、BART 等，GENIUS 抛弃的信息更多。

构造草稿的大致过程是对于一个文本，会抽取其关键信息，然后对剩余部分全部丢弃，对每一段连续丢弃的语句，使用一个掩码标记进行填补。具体步骤分为三步：提取、预测、掩盖。

其中，提取步骤，使用无监督抽取工具 YAKE 来抽取最大为三段的关键短语，在预测步骤中，会允许一个词出现多次，并按照原始的顺序，最后在掩盖步骤中，使用掩码标记进行对要丢弃的部分进行替换。构造好短语之后，会将短语输入改模型，然后输出原始的文本，即根据草稿对原文进行重构。

GENIUS 采用的是一种选择性掩盖的方式，只对不重要的部分进行掩盖，而 BERT、BART 等一系列方法，均采用随机掩盖的方式，并且 GENIUS 的掩盖比例可以高达 80%，堪称极端的掩盖，而 BERT 系列方法只使用 15%的掩盖比例，BART 更大一点，但也只有 30%。这两点不同，使得 GENIUS 可以在仅仅根据几个关键词，或者短语，就重构出大段的文本，这是 BERT、BART 等模型所不具备的能力[11]。

GENIUS 模型使用步骤如下：

（1）安装 Anaconda

（2）打开 Anaconda Prompt 终端

（3）创建新的 conda 环境

```
conda create --name chenyanting_env python=3.8
--channelhttps://mirrors.tuna.tsinghua.edu.cn/anaconda/pkgs/free/
```

使用如上命令创建了名为 chenyanting_env 的虚拟环境，python 版本为 3.8，并使用清华源加速下载

（4）搭建 cuda 和 pytorch 适配环境方便 GPU 训练

```
conda install torchvision==0.13.1  pytorch==1.12.1  cudatoolkit=11.3
torchaudio==0.12.1  -c pytorch
```

（5）pip 安装必备包

```
pip install transformers -i https://pypi.tuna.tsinghua.edu.cn/simple
```

（6）使用模型进行关键词生成短语

```
from transformers import MaskFormerFeatureExtractor,
MaskFormerForInstanceSegmentationfrom PIL import Imageimport requests
from transformers import BertTokenizer, BartForConditionalGeneration, Text2TextGener
ationPipeline
checkpoint = 'beyond/genius-base-chinese'
tokenizer = BertTokenizer.from_pretrained(checkpoint)
genius_model = BartForConditionalGeneration.from_pretrained(checkpoint)
genius_generator = Text2TextGenerationPipeline(genius_model, tokenizer, device=0)
genius_generator
sketchs = [
```

```
    "今天[MASK]篮球[MASK]学校[MASK]",
    "自然语言处理[MASK]谷歌[MASK]通用人工智能[MASK]",
    "我昨天做了一个梦[MASK]又遇见了她[MASK]曾经那段时光让人怀恋[MASK]",
    "[MASK]疫情[MASK]公园[MASK]散步[MASK]",
    "[MASK]酸菜鱼火锅[MASK]很美味，味道绝了[MASK]周末真开心[MASK]"
    ""
]
for sketch in sketchs:
    print('input sketch:\n>>> ', sketch)
    print('genius-chinese output:\n>>> ',genius_generator(sketch, max_length=200, do
_sample=True, num_beams=3)[0]['generated_text'].replace(' ',''),'\n')
```

（7）结果展示

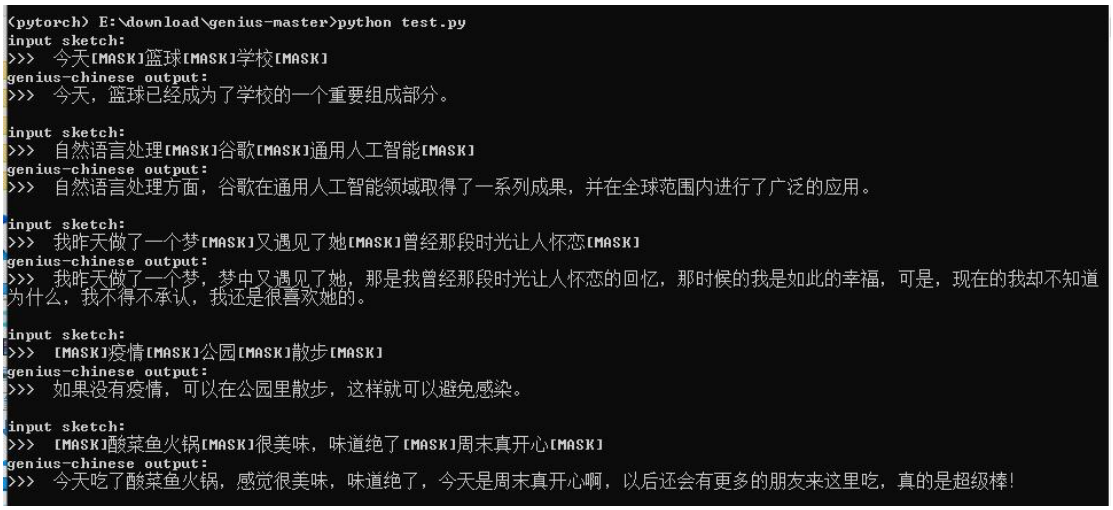



图 2.6 GENIUS 模型处理结果

图 2.6 是 GENIUS 模型处理结果，可以看到 GENIUS 可以在使用【MASK】标记的位置自动填充符合上下文语义的内容，并且填充幅度比较大，所以它是可以将关键词生成短语的算法。

通过上面的使用，可以看到 GENIUS 模型，可以将关键词和掩码标记[MASK]组成的句子，生成一句短语，会使用其他语句来填充 MASK，并且当关键词的顺序和掩码标记的位置不同时，生成的语句也会不同，这为后面的短语续写提供了非常大的发挥空间。

## 2.3 根据短语续写文章

### 2.3.1 预训练技术

通过预练习方法，模型变量不再是随机的，而是通过建立模板来进行训练，从而提高模型的准确性。然而，目前，解决 NLP 挑战的最佳策略，仍然是通过

提前训练和精细调整来实现。NLP 的发展离不开预训练语言的作用，它的应用范围已经从最初的 Word2vec、Glove 等技术扩展至现在的 ULMFiT、EMLo 等众多文本处理技术，为人工智能的发展提供了强大的支持。Transformer 模型是当前最出色的预训练语言模型之一，它具有出色的性能。基于 Self-Attention 理论构建的这种模式，是当今 NLP 技术中最具潜力的图像信息获取器，它不仅可以自动捕捉图像，而且还可以远距离捕捉依赖信息[12]。

BERT 是一种基于 Transformer 的双向深度学习技术，它在当前的语言学领域中发挥着重要作用。"BERT“是一种复杂的多头解码器系统，其中标准版拥有 12 个多头注意力层，而其他两个版本则拥有 768 个隐藏层，以满足不同的应用需求；24 层增强版的 Transformer 具备 24 个独立的注意力层，其中最高可达 1024 个，这样可以显著改善系统的运行效率，并且极大地提高系统的整体表现。显然，深度较窄的模型具有更优越的性能。BERT 已经取得了惊人的成就，它在机器翻译、文本识别、语义匹配、翻译理解等多个领域都取得了出色的表现。（1）通过使用特定的语言结构来构建 2 种不同的 BERT 模型。（2）使用预测技术来确定句子的下一个部分。

经过采用上述两种技术，我们可以建立一个普遍适用的模型，并且可以利用这些模型来实现更多的任务，包括文本分析、机器翻译等。与传统的预训练模型相比，BERT 能够更准确地捕捉双向上下文的语义信息。尽管 BERT 具有许多优势，但它也存在一些不足之处，例如，大量使用会降低模型的准确性，而且每批次的标记数量仅为 15%，这使得它在训练过程中的收敛速度变得缓慢。由于 BERT 的预处理过程缺乏统一性，导致它在自动化语音处理方面的表现较差，并且它无法完成文本处理的 NLP 任务，仅能用于处理句子或段落。

### 2.3.2 BERT 模型

Bidirectional Encoder Representations from Transformers(BERT)是一种先进的双向编码技术，可以有效地提取和传递一个词语的语境信息，使得读者可以更加清晰、准确地理解其语义，同时也可以通过对比不同的语境信息，灵活调节编码器的位置，以便更加精准地传递信息[13]。

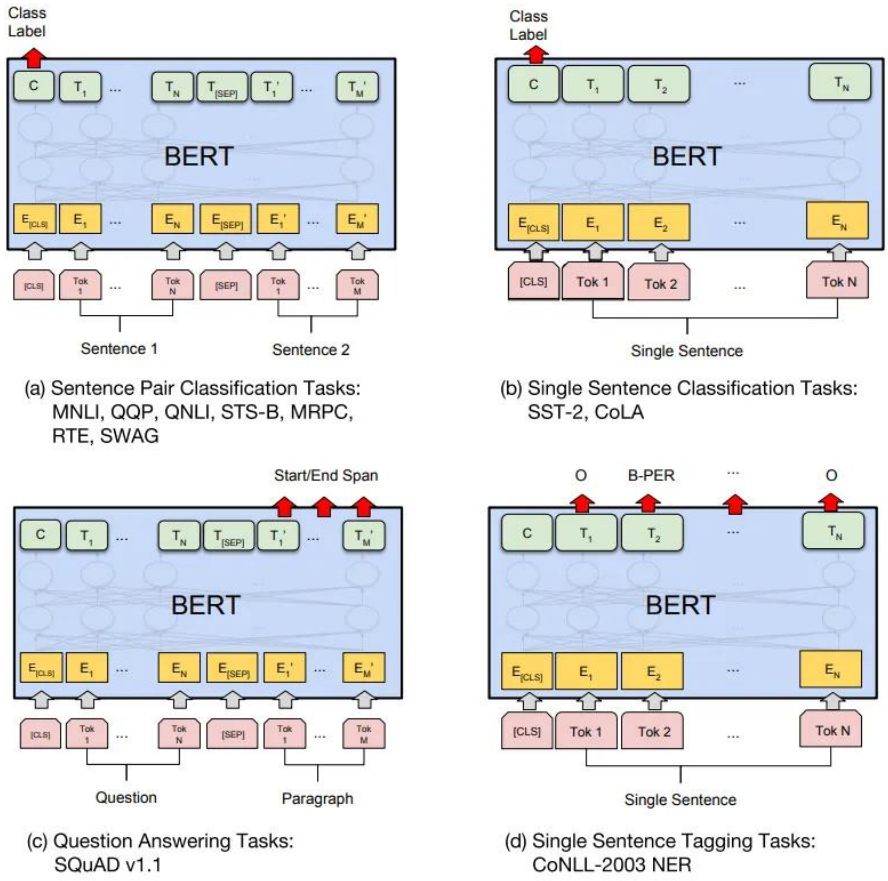



图 2.7 BERT 模型的组成部分

图 2.7 是 BERT 模型的组成部分，分为四大任务，分别是多语句分类任务、单语句分类任务、回答问题任务、单语句打标签任务。这四个任务相互配合共同完成任务，首先单语句打标签任务为语句打标签，然后多语句分类任务、单语句分类任务对语句进行分类，最后由回答问题任务将前面的处理结果总结并输出最终结果。

BERT 为了进行深度的双向表达，使得双向的作用使得同一个词可以同时在多个语境中间接的看到它。选择使用 MLM(Masked Language Model)方法，随机屏蔽掉部分的句子，接着再去检测一些被屏蔽掉的句子，并且在检测时从句子的视角来思考问题，同时还可以在语料库中得到句子。

通过 BERT 预训练，我们可以构建一个更加复杂的真实双向语言模型，而不仅仅局限于传统的马尔可夫链编码，这种新的编码方式可以有效地抑制噪声，提高模型的效率。采用完形填充技术能够有效地克服传统语法结构中的编码限制。采用完形填空策略，可以有效地将 15% 的单词隐藏起来，并利用编码器的最终状态，以及一层归一化指数函数，来实现对完形填空的准确预测。但是这种策略会有二个缺点，第一个缺陷就是不匹配，由于掩盖的标记永远不会出现在微调阶段，造成了预训练阶段与微调阶段之间形成了不匹配。可以采用缓解方式，针对

随机选择的百分之十五待掩盖单词，不是直接将它替换为遮蔽标记，而是再作一次随机，以百分之八十的概率将这个单词替换为遮蔽标记，以百分之几十的概率将这个单词替换为下一次随机的单词，以百分之几十的概率不替换，但是因为 Transformer 编码器还不了解那些单词被要求事先作预测，又或是那些单词已经被随意地替换掉了，所以针对每个新输入的单词，它都必须保持上下文嵌入；并且，现在这种策略下随机替换掉的单词也只是百分之一点五，所以基本不会影响模型的语言建模能力。第二个缺点也就是现在使用的 MLM 模型中，由于每批次大约有百分之十五的单词被错误预测，使得收敛速度的确减慢了，但是对效果所产生的改善却不少。

通过 NLP（Natural Language Processing），许多句子之间存在着密切的相互关系，因此，为了进一步提升任务二的预测准确度，我们将它视为一个二分类任务，采取多任务学习的策略，以更好地发挥任务一的 MLM 功能。替换 50% 的句子为随机的句子，我们可以构建一个负样本，从而更深入地探索语言的本质特性。

### 2.3.3 GPT 模型

GPT(Generative Pre-Trained Transformer)就是通过 Transformer 为基础模型，使用预训练技术得到通用的文本模型。

GPT 训练方法可以分为两种：一种是不受外界干扰的自动化学习，另一种则是基于外部信号的自动化学习。GPT 的自动化学习可以通过一个没有标记的序列 g={g1，……,gn}来实现,从而更好地理解基因的表达特征。通过优化，我们的目标是尽可能地提高下列几个似然值的精度。

$$L_1(g) = \sum_i \log P(g_i \mid g_{i-k},...,g_{i-1};\theta) \quad (2.1)$$

公式（2.1）里面的 k 是滑动窗口的尺寸，P 是条件概率值，$\theta$ 是模型的参数。

GPT 采用了 12 个 transformer 模块作为解码器，它们具有多头的自注意力机制，并且能够根据整个链路的可能性分布来估算输出。

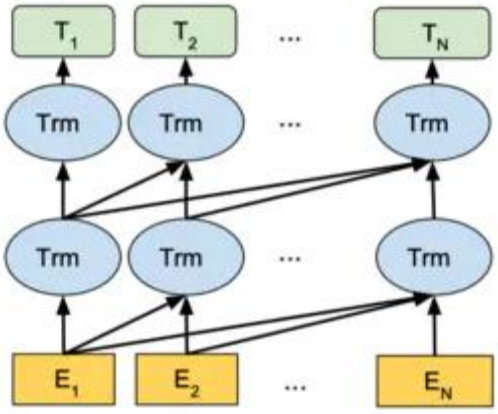



图 2.8 GPT 模型的组成部分

图 2.8 是 GPT 模型的组成部分，可以看到多个 transformer 模块直接相互通信，其不同层 transformer 之间的通信公式如下：

$$h_0 = UW_e + W_p \quad (2.2)$$

$$h_l = \text{transformer\_block}(h_{l-1}) \forall i \in [1, n] \quad (2.3)$$

$$P(u) = \text{softmax}(h_n W_e^T) \quad (2.4)$$

在公式（2.2）中，U=（U−k,...,U−1）代表当前时间段的前后关系，n 则指代层次，We 则指代动词嵌入矩阵，而 Wp 则指代位置嵌入矩阵，公式（2.3）是对 h0 进行转换迭代，公式（2.4）则是进行最大池化操作。

GPT 的有监督微调技术允许在没有外部干扰的情况下，通过对模型进行训练，并将其相关的参数应用于需要外部干扰的情况下，来达到有效的微调效果。在一个具有标记的数据集合 C 中，每个实例都包含 m 个输入词{x1，…，xm}以及一系列其他输入词是一个由一个特定的符号 y 构成的集合。通过将该语句插入之前的预处理模型，我们可以获得一系列有效的特征向量。经过一个完整的连接层，我们可以获得 y 的精确信息。

$$P(y|x^1, \ldots, x^m) = \text{softmax}(h_l^m W_y) \quad (2.5)$$

公式（2.5）中 Wy 是全连接层的参数而监督的效果则是最大化上式的数值:

$$L_2(C) = \sum logP(y|x^1, \ldots, x^m) \quad (2.6)$$

公式（2.6）表明当进行所有监督微调的时候，也可只使用输出层的 $W_y$ 值或者分隔符的嵌入值。

GPT 模型使用步骤如下：

（1）安装 Anaconda

（2）打开 Anaconda Prompt 终端

（3）创建新的 conda 环境

```
conda create --name chenyanting_env python=3.8
--channelhttps://mirrors.tuna.tsinghua.edu.cn/anaconda/pkgs/free/
```

使用如上命令创建了名为 chenyanting_env 的虚拟环境，python 版本为 3.8，并使用清华源加速下载

（4）搭建 cuda 和 paddlepaddle 适配环境方便 GPU 训练

```
conda install  paddlepaddle-gpu==2.0.0   cudatoolkit=10.1  -c paddle
```

（5）pip 安装必备包

```
pip install paddlehub==2.0.0rc0 cudatoolkit=11.3 -c pytorch
pip install sentencepiece==0.1.92
```

（6）使用模型进行关键词生成短语

```
# 导入 PaddleHub 库（运行一次即可）
import paddlehub as hub
# 加载 CPM-LM 模型（运行一次即可）
```

```
# 加载需耗时 1 分钟左右，请耐心等待
model = hub.Module(directory='CPM_LM')
# 设置输入文本
inputs = '''方平带众人骑马出了城，残雪点缀原本泛黄的大地。他一身黑衣在一群铁甲士兵中尤其显眼
'''
# 使用 CPM-LM 模型进行文本续写
outputs = model.sample(
    inputs, # 输入文本
    max_len=128, # 最大生成文本的长度
    end_word=None, # 终止符号
    repitition_penalty=1.0, # 重复度抑制
    temperature=1.0, # 温度
    top_k=3000, # 取前 k 个最大输出再进行采样
    top_p=1.0 # 抑制概率低于 top_p 的输出再进行采样
)
# 打印输出
print(outputs)
```

（7）结果展示：下面图 2.9 是短语续写的结果，可以将短语续写成一段比较长的话。可以看出续写出的语句比较通顺，并且可以续写出比较长的语句，当对图片的描述有比较长的字数要求时可以优先选择该算法。并且选用的这个模型比较适合续写文学小说，可以扩展看图写话功能，变为看图联想写一篇文学的文章，如小说等文章。

```
[notice] To update, run: pip install --upgrade pip
aistudio@jupyter-3124264-5406254:~$ python test.py
/opt/conda/envs/python35-paddle120-env/lib/python3.7/site-packages/paddle/fluid/layers/utils.py:26: DeprecationWarning: `np.int` is a de
precated alias for the builtin `int`. To silence this warning, use `int` by itself. Doing this will not modify any behavior and is safe.
 When replacing `np.int`, you may wish to use e.g. `np.int64` or `np.int32` to specify the precision. If you wish to review your current
 use, check the release note link for additional information.
Deprecated in NumPy 1.20; for more details and guidance: https://numpy.org/devdocs/release/1.20.0-notes.html#deprecations
  def convert_to_list(value, n, name, dtype=np.int):
/opt/conda/envs/python35-paddle120-env/lib/python3.7/site-packages/matplotlib/__init__.py:107: DeprecationWarning: Using or importing th
e ABCs from 'collections' instead of from 'collections.abc' is deprecated, and in 3.8 it will stop working
  from collections import MutableMapping
/opt/conda/envs/python35-paddle120-env/lib/python3.7/site-packages/matplotlib/rcsetup.py:20: DeprecationWarning: Using or importing the
ABCs from 'collections' instead of from 'collections.abc' is deprecated, and in 3.8 it will stop working
  from collections import Iterable, Mapping
/opt/conda/envs/python35-paddle120-env/lib/python3.7/site-packages/matplotlib/colors.py:53: DeprecationWarning: Using or importing the A
BCs from 'collections' instead of from 'collections.abc' is deprecated, and in 3.8 it will stop working
  from collections import Sized
W0119 11:43:44.703116   487 device_context.cc:362] Please NOTE: device: 0, GPU Compute Capability: 7.0, Driver API Version: 11.2, Runtim
e API Version: 10.1
W0119 11:43:44.709244   487 device_context.cc:372] device: 0, cuDNN Version: 7.6.


Building prefix dict from the default dictionary ...
2023-01-19 11:44:52,072 - DEBUG - Building prefix dict from the default dictionary ...
Dumping model to file cache /tmp/jieba.cache
2023-01-19 11:44:52,786 - DEBUG - Dumping model to file cache /tmp/jieba.cache
Loading model cost 0.773 seconds.
2023-01-19 11:44:52,845 - DEBUG - Loading model cost 0.773 seconds.
Prefix dict has been built successfully.
2023-01-19 11:44:52,845 - DEBUG - Prefix dict has been built successfully.
方平带众人骑马出了城,残雪点缀原本泛黄的大地。他一身黑衣在一群铁甲士兵中尤其显眼。“你们怎么看?”“白墨报仇了,进去就是了。”禁卫军原本就对
白墨兄弟怨恨不已,闻言激烈赞同。殇山河一带确实很少有人知晓白墨,白墨和凰火率领的自由骑士团那么出名,亡魂骑士更是纵横自由位面数十年不倒,短短
半年多的时间福地第二军,殇山河这样的小地方就什么都能打听出来,凭白墨与
```

图 2.9 短语续写结果

通过上面的使用，可以看出 GPT 模型续写的文本语句通顺，容易理解，且富有含义，但是由于内存限制最多续写文字只有 128 字。

## 2.4 视觉编码器-解码器模型

### 2.4.1 概念

视觉编码器-解码器模型可以来初始化从视频到文本模型，包括将所有预训练的基于 Transformer 的视觉模型作为解码器，还有所有预训练的模型都用作解码器。在读图写话时，解码器模块用来对图象进行解码，然后自回归语言建模，而解码器模块则用于合成描述性语句。

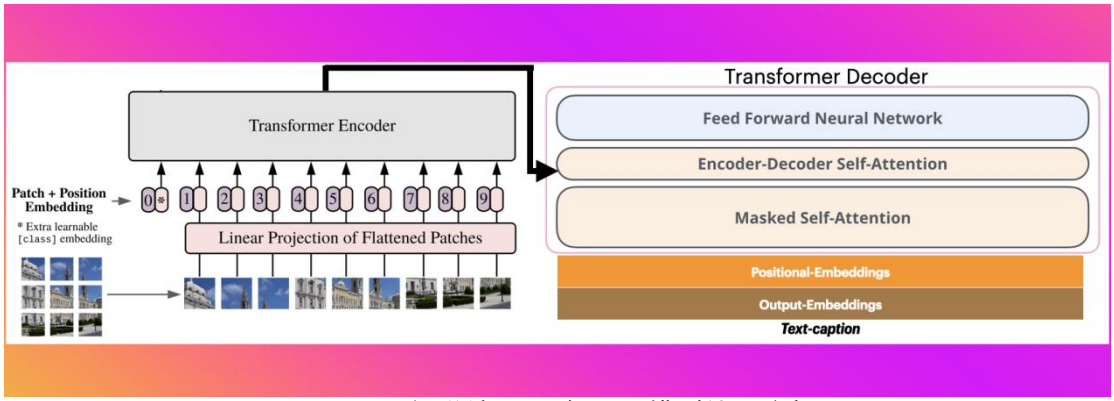



图 2.10　视觉编码器-解码器模型处理流程

上面图 2.10 是视觉编码器-解码器模型处理流程，其大致步骤是将一张图片切分为 9 块扁平块，并通过线性投影投入 Transformer 编码器。线性投影模型如下：

$$y = x'\beta + e \tag{2.7}$$

公式（2.7）是进行线性投影转换，在图片进行完线性投影转换后，会送入 Transformer 解码器中，在 Transformer 译码器中再进行前馈神经网络处理，由前馈神经网络将各神经元层级排列，每个神经元只和上一级的神经元连接。接受上级的数据，再传递到下层，各层之间没有传递。然后通过编码器-解码器自注意力机制来让机器注意到整个输入中不同部分之间的相关性。然后通过掩码标记的自注意力机制，关注到被掩盖的字符之间的相关性。最后通过位置嵌入和输出嵌入得到一段描述性语言。

### 2.4.2 视觉编码器-解码器模型的使用

（1）安装 Anaconda

（2）打开 Anaconda Prompt 终端

（3）创建新的 conda 环境

```
conda create --name chenyanting_env python=3.8
--channelhttps://mirrors.tuna.tsinghua.edu.cn/anaconda/pkgs/free/
```

使用如上命令创建了名为 chenyanting_env 的虚拟环境，python 版本为 3.8，并使用清华源加速下载

（4）搭建 cuda 和 paddlepaddle 适配环境方便 GPU 训练

```
conda install  cudatoolkit=10.1  paddlepaddle-gpu==2.0.0 -c paddle
```

（5）pip 安装必备包

```
pip install paddlehub==2.0.0rc0 cudatoolkit=11.3 -c pytorch
pip install sentencepiece==0.1.92
```

（6）使用模型进行关键词生成短语

```
# 导入 PaddleHub 库（运行一次即可）
from transformers import VisionEncoderDecoderModel, ViTImageProcessor, AutoTokenizer
import torch
from PIL import Image
# 加载 CPM-LM 模型（运行一次即可）
# 加载需耗时 1 分钟左右，请耐心等待
model = VisionEncoderDecoderModel.from_pretrained("nlpconnect/vit-gpt2-image-captioning")
feature_extractor = ViTImageProcessor.from_pretrained("nlpconnect/vit-gpt2-image-captioning")
tokenizer = AutoTokenizer.from_pretrained("nlpconnect/vit-gpt2-image-captioning")
device = torch.device("cuda" if torch.cuda.is_available() else "cpu")
model.to(device)
max_length = 16
num_beams = 4
gen_kwargs = {"max_length": max_length, "num_beams": num_beams}
def predict_step(image_paths):
  images = []
  for image_path in image_paths:
    i_image = Image.open(image_path)
    if i_image.mode != "RGB":
      i_image = i_image.convert(mode="RGB")
images.append(i_image)
  pixel_values = feature_extractor(images=images, return_tensors="pt").pixel_values
  pixel_values = pixel_values.to(device)
  output_ids = model.generate(pixel_values, **gen_kwargs)
  preds = tokenizer.batch_decode(output_ids, skip_special_tokens=True)
  preds = [pred.strip() for pred in preds]
  return preds
 # ['a woman in a hospital bed with a woman in a hospital bed']
#进行预测
print(predict_step(['1.jpg']))
```

（7）结果展示：下面图 2.11 是输入的待描述图片，目标是描述出该图片里面的元素的行为，可以看到该狗在草丛里呆坐着，可能正盯着摄像头。

图 2.11 输入的待描述图片

下面图 2.12 是视觉编码器-解码器模型加载后进行预测的过程，最后输出结果：a black and white dog is looking at the camera，表明的是一个黑白的狗正在看着摄像头，说明该视觉编码器-解码器模型还具有推理能力，可以推理出图片中不存在却合理的描述。

```
You have 18 unapplied migration(s). Your project may not work properly until you apply the migrations for app(s): admin,
 auth, contenttypes, sessions.
Run 'python manage.py migrate' to apply them.
April 23, 2023 - 19:15:15
Django version 3.2.18, using settings 'imageCaptioning.settings'
Starting development server at http://127.0.0.1:8000/
Quit the server with CTRL-BREAK.
b'E:/nginx-1.8.0/html/temp/static/result/2a7f0a9c-751a-4484-b6c0-15e00086fdf5.jpg'
[23/Apr/2023 19:38:38] "POST /api/imageCaptionHandle HTTP/1.1" 200 12
获取到的图片路径参数：E:/nginx-1.8.0/html/temp/static/result/2a7f0a9c-751a-4484-b6c0-15e00086fdf5.jpg
替换后的图片路径参数：E:\nginx-1.8.0\html\temp\static\result\2a7f0a9c-751a-4484-b6c0-15e00086fdf5.jpg
开始加载模型
开始预测
预测结果：
['a black and white dog is looking at the camera']
写入文件完成路径为：E:\nginx-1.8.0\html\temp\static\result\data.json
```

图 2.12 视觉编码器-解码器模型处理过程和处理结果

### 2.4.3 视觉编码器-解码器模型总结

通过上面的使用，视觉编码器-解码器模型可以直接处理图片得到一段描述性语言，可以一步从图片翻译成描述性语言，且描述的准确度也比较高，预测结果也比较符合逻辑，是一个很不错的图像字幕算法模型。

# 3 看图写话系统的后端部分设计

本章主要讲的是看图写话系统后端部分设计，会先给出系统的整体架构，再逐一讲解架构中每一个模块的作用和意义，讲述后端一些架构的基本原理，以及设计缘由。

## 3.1 系统架构

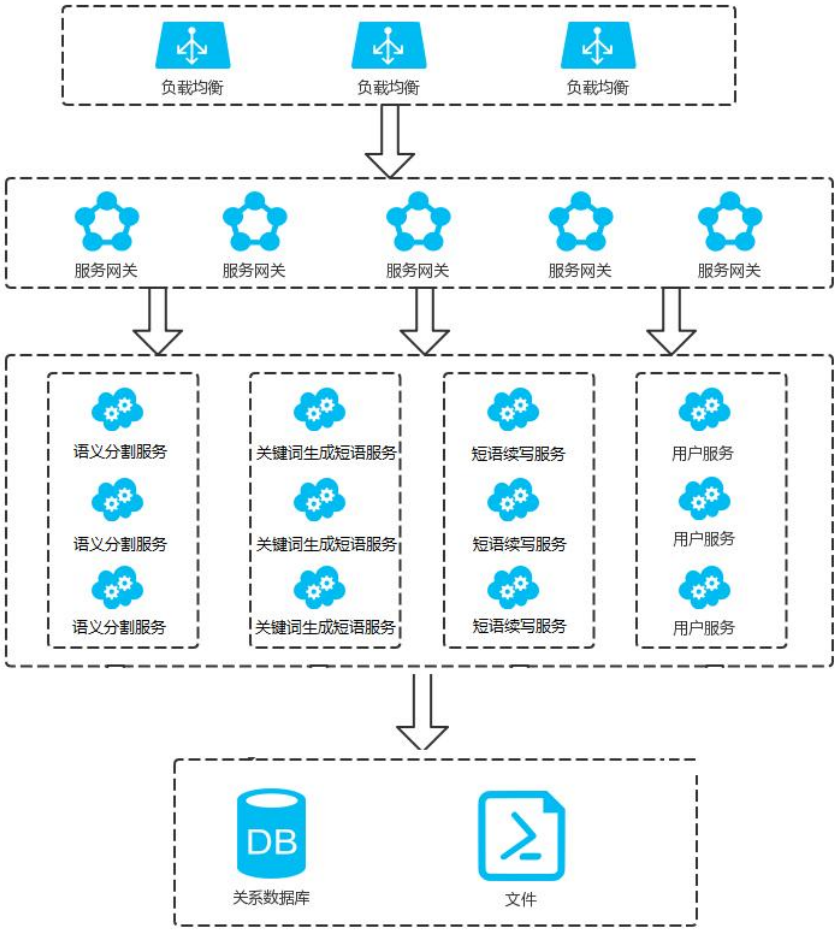



图 3.1 后端服务器架构

上面图 3.1 是后端服务器架构，后端采用是分层架构，以实现快速响应以及承载多请求的功能，在最顶层是负载均衡用来均分压力，负载均衡是使用将请求均匀分给多台服务器，让每台服务器都处理一部分请求，从而达到缓解服务器压力和增大并大量的功能。再下一层是服务网关用来将请求路由到响应的算法微服务器上，在负载均衡后的请求需要知道自己需要去哪台算法微服务器上，且服务网关还可以充当权限校验的功能，拦截非法和未知的请求。再下一层是算法微服务器层，将算法以微服务的方式暴露接口给服务网关，这样设计的目的是提升算法模块的可扩展性，可以随时替换实现看图写话的算法，以比较不同算法之间

的效果和准确率。最底层是存储层，存储层由数据库和文件组成，用来存放后端和算法服务器的处理数据，文件则是用户上传的图片之类的文件。

通过不同层之间的合理分工和紧密配合，就可以搭建好一个高性能，可靠且响应速度快的后端平台，可以持续且稳定的为前端用户提供看图写话系统的服务，提高用户体验感。

## 3.2 负载均衡

负载均衡是一种有效的技术，它可以将复杂的任务分配给多个独立的、协作性强的工作单元，以实现更高效的任务完成，提高工作效率和质量。负载均衡技术是一种高效的网络架构，它不仅可以有效地优化网络设施和服务器的带宽，还能够显著增强系统的吞吐量，并且还可以有效地提高系统的灵活性、可靠性，从而实现节省成本、提高运行效率的双重目标。

### 3.2.1 负载均衡算法

（1）轮询法：通过轮询，我们可以按照顺序向服务器提交用户的请求，这样就可以一次性收集大量的数据。虽然这种计算方法相对简单，而且具有绝对均衡的优势，但由于其耗费的时间和资源都相当可观，因此它不能完全确保任务的合理分配，也不能仅凭借服务器的能力来实现任务的有效分配。

（2）随机法：采用随机法，可以从众多服务器中精确挑选出最合适的任务。通过采用这种方式，我们能够将服务请求尽可能地分散，以达到公正和平等。它不再受到状态的限制，也不需要重复上次的选择和平衡条件[5]。随着任务量的增加，轮询算法可能会遇到一些挑战，这些挑战可能会影响最终的结果。

（3）最小连接法：最小链接算法将任务指派给在此时具有最小连接数的节点，由于这是动态负载平衡方法。在某个节点接收到下一个任务时连接数将被设计为 1，但在节点失败后就会将节点的最大连接权数设置为 0，从而不再给节点分配任务。所以最小链接算法适用于当各个节点接收的任务性质一致时。通常，任务分配单元会将任务分配给服务器，以确保任务的性质和可靠性。当服务器的性能存在显著的差异时，任务的完成可能会受到严重的影响。由于最大连接数无法准确反映服务器的处理能力，因此，即使连接数较小，但其功能也可能不及拥有更高连接数且功能更优的服务器。然而，由于某些原因，任务无法被有效地分配给具有更高处理性能的设备，从而影响了整体效率。

（4）源地址哈希法：通过哈希函数，我们可以获取服务器的 IP 地址，并利用这些信息来确定服务器的大小。这样，我们就能为客户端提供唯一的地址编号。使用原地址哈希算法，我们能够在客户端列表保持不变的情况下，将所有的资源映射到同一台后台服务器，从而达到负载均衡的目的，并且能够更好地利用这些资源。

### 3.2.2 负载均衡技术

（1）基于 DNS 的负载均衡：因为在 DNS 网络上，可以给几个不同的地址分配同样的名称，而搜索这个名称的客户端会在分析这个名称后获取其中某个位置，所以这个代理方法就是利用 DNS 网络上的随机名字和域名的 IP 从而达到负载平衡。

（2）反向代理负载均衡：反向代理模式与传统的标准代理模式有着本质的区别，前者是指几个用户共同使用一个 Web 客户端，而后者则是指几个用户分别使用不同的客户端，从而实现代理服务的交互。

（3）基于 NAT（Network Address Translation）的负载均衡网络:利用位置转换技术，我们可以将外部设备的位置精确地映射到内部服务器，这样就可以保证内部服务器和外部设备之间的高效沟通，并且达到负荷平衡的目的。通过将外部地址与内部地址进行映射，我们能够在两者之间建立一个桥梁，从而让外部客户端能够轻松访问内部服务器，并将数据传输至指定的目的地。这样，即使是一个未知的目的地，也能够有效地传输。

### 3.2.3 负载均衡技术的使用

我们也可以使用 Nginx 来进行高负载平衡，Nginx 作为一个轻量级的网络客户端、反向代理服务器，由于它的内存占用比较小，并且运行速度快，高并发功能强，常在互联网开发中应用。

（1）安装 nginx

（2）配置负载均衡

修改 nginx 配置文件 nginx.conf，在其中加入 upstream 关键词，用来指定负载均衡的算法和进行负载均衡的主机 IP 地址端口

```
#轮询
upstream chenyanting{
      server 192.168.117.101:9000
      server 192.168.117.102:9000 down;
      server 192.168.117.103:9000 backup;}
```

当前的 server 已经不再参与负载，//down 表示它已经停止运行.

当其他或全部机器处于非 backup 状态时，如果它们请求了 backup，那么这台机器的负荷将会大大降低，从而提高系统的效率和可靠性。

上面使用的 nginx 的轮询算法，表示在 ip 地址分别为 192.168.117.101、192.168.117.102、192.168.117.103 的三台主机进行轮流请求。

```
#加权轮询
upstream chenyanting{
      server 192.168.117.101:9000 weight=2;
```

```
      server 192.168.117.102:9000 weight=3;
      server 192.168.117.103:9000 weight=5;
}
```

加上 weight 关键词进行加权轮询，值越大则权重越大，则被选中进行请求处理的概率越大。

```
#源地址哈希
upstream chenyanting{
      ip_hash;
      server 192.168.117.101:9000;
      server 192.168.117.102:9000;
      server 192.168.117.103:9000;
}
```

加上 ip_hash 关键词使用源地址哈希算法来选择主机。通过哈希算法，我们可以从服务器的 IP 地址中提取一个特定的值，并使用它来确定服务器的位置，从而实现对服务器的哈希，从而更好地了解服务器的状态。采用哈希计算技术，可以在保持 IP 地址不变的情况下，将所有的网络资源映射到一个相同的后端服务器上，从而达到负荷均衡的效果。

```
#源地址哈希
server {
        listen       80;
        server_name  localhost;
         location /api/ {
            proxy_pass http://chenyanting;
        }
}
```

上面的配置是开启一个服务监听在本地的端口 80 处，则可以通过访问 http://localhost:90 来访问该服务，并通过访问 http://localhost:90/api 来访问后端接口，并使用 proxy_pass 转发进行负载均衡操作，转发到的主机正是我们之前在 upstream 中定义的 ip 地址和端口

## 3.3 微服务

“微服务”是一种新型的现代软件开发技术，它通过将复杂的 AP 拆解成更小的服务块来实现对整个系统的支持。这种方法能够让不同的业务部门能够有效地协同工作，并且能够为用户带来更好的体验。所有的服务都被安装在自己的系统中，而且可以使用低成本的通信协议（通常是基于 HTTP 的 RESTful API）来

实现业务和服务之间的交流。所有的服务都是基于特定的业务需求来构建的，可以轻松地应用于各种产品和类别的环境。

在这里我们将系统分为前端服务、后端服务和算法服务，而算法服务又细分为语义分割服务、关键词生成短语服务、短语续写服务、用户服务。

### 3.3.1 服务网关

通过使用网关，我们可以为多个节点提供代理服务。通过使用服务网关，我们能够实现跨企业之间的各种服务交互、路由调度以及公共管理。服务网关模块是单一调解，用于处理多个服务使用者和提供者的请求。

任何服务网关都有如下四个典型步骤：

（1）常用处理。一旦网关接收到消息，就对所有消息执行常用处理，例如添加协议级的头或者记录该消息。

（2）服务标识。必须将网关所处理的消息标识为特定服务类型。例如，查询消息以确定它是针对服务提供者 A、B 还是 C。

（3）端点路由。当它确定某消息将传递到特定服务提供者时，它将映射到网络可寻址端点，以便可以将该消息转发到服务提供者。

（4）特定服务的处理。执行特定目标服务所需的任何处理。

### 3.3.2 服务网关的使用

我们可以使用 nginx 来充当服务网关，nginx 可以通过对不同路径进行不同的转发，从而达到网关的代理效果，可以用来处理多个服务使用者和提供者的请求。可以进行如下配置

```
#源地址哈希
server {
        listen       80;
        server_name  localhost;
         location /SemanticSegmentation/ {
            proxy_pass http://www.chenyanting.com:9000/server;
        }
         location /KeywordContinuation/ {
            proxy_pass http://www.chenyanting.com:9001/server;
        }
         location /PhraseContinuation/ {
            proxy_pass http://www.chenyanting.com:9002/server;
        }
         location /User/ {
            proxy_pass http://www.chenyanting.com:9003/server;
        }
}
```

以上配置表示，当想访问语义分割服务时只需访问链接 http://localhost:90/SemanticSegmentation，当想访问关键词生成短语服务时只需访问链接 http://localhost:90/KeywordContinuation，当想访问短语续写服务时只需访问链接 http://localhost:90/PhraseContinuation，当想访问用户服务时只需访问链接 http://localhost:90/User，而 nginx 会自动帮我转发请求到想要请求的服务上去，这样设计还能支持异构服务，可以支持不同语言搭建的服务器，这里我打算使用 Java 语言进行后端服务器的搭建，使用 Python 语句进行算法服务器的搭建，可以通过 nginx 将它们整合在一起。

### 3.3.3 搭建算法服务器

由于算法是由 Python 实现的，所以我们需要使用 Django 来搭建一个 Python 服务器,从而实现语义分割微服务的搭建。

Django 框架采用 MVC 设计理念，为开发者提供一种全新的解决方案。MVC 由三个关键概念组成：Model、View 和 Controller，它们共同构成了一个完整的模型，用于实现可视化和控制。Django 框架可以被视为一种改进的 MTV 设计方法。MTV 的结构由三个部分组成：Model、Template 和 View，它们共同构成了 mtv 的三个基本要素：模型、模板和视觉效果。在 Django 框架中，控制器系统提交的所有信息都已经被精心处理，而最重要的是模型、模板和视图，这些组件构成了一个完整的系统，被称为 MTV。其各自的主要特点如表 2 所示：

**表 2　MTV 层次特点表**

| 层次 | 职责 |
| --- | --- |
| 模型（Model），即数据存取层 | 处理与数据相关的所有事务： 如何存取、如何验证有效性、包含哪些行为以及数据之间的关系等。 |
| 模板(Template)，即表现层 | 处理与表现相关的决定： 如何在页面或其他类型文档中进行显示。 |
| 视图（View），即业务逻辑层 | 存取模型及调取恰当模板的相关逻辑。模型与模板的桥梁。 |

根据 Django 的框架视图，它不仅仅是为了处理用户的输入，还能够为用户提供一个清晰的展示界面，从而使得用户能够更加轻松地获取所需的信息。Django 的模板不仅可以用来展示框架视图中的特定内容，而且还可以更好地满足用户的需求。通过 Django，用户可以将 MVC 中的视图细分为框架视图和模板，并且可以根据自身需求自由调整模板大小，从而更有效地展示数据。

MVC 控制器的实现，由 Django 架构中的 URLconf 来完成，以满足系统的

需求。通过采用 URLconf 机制，我们可以利用正则表达式的 URL，以及精心设计的 Python 语言来实现高效的传输。URLconf 提供了一种灵活的 URL 模式，无论您想要保持传统、RESTful、还是独特的风格，都能够轻松地实现。架构中的每个控件层都已经封装好了，并且每个可以无缝直接和数据通信的层都有对数据库表的读出，输入，删除，以及修改的接口。通过采用适当的方法，你可以在编制过程中大大提高效率，并且可以获得更好的结果。Django 架构已经将程序员的控制权转移到了它的自动化系统中。通过编写简单的程序，我们可以完成大量的任务。然而，与 MVC 框架相比，它更加全面地解决了问题，因为现在的程序员们都在控制层的更小的范围内进行工作。把这些工作交给了它，就只需花极少的时间调用程序，从而大大提高了工作效率。

（1）安装 Django

```
pip install django -i https://pypi.tuna.tsinghua.edu.cn/simple
```

（2）创建 Django 项目

```
django-admin startproject SemanticSegmentation
```

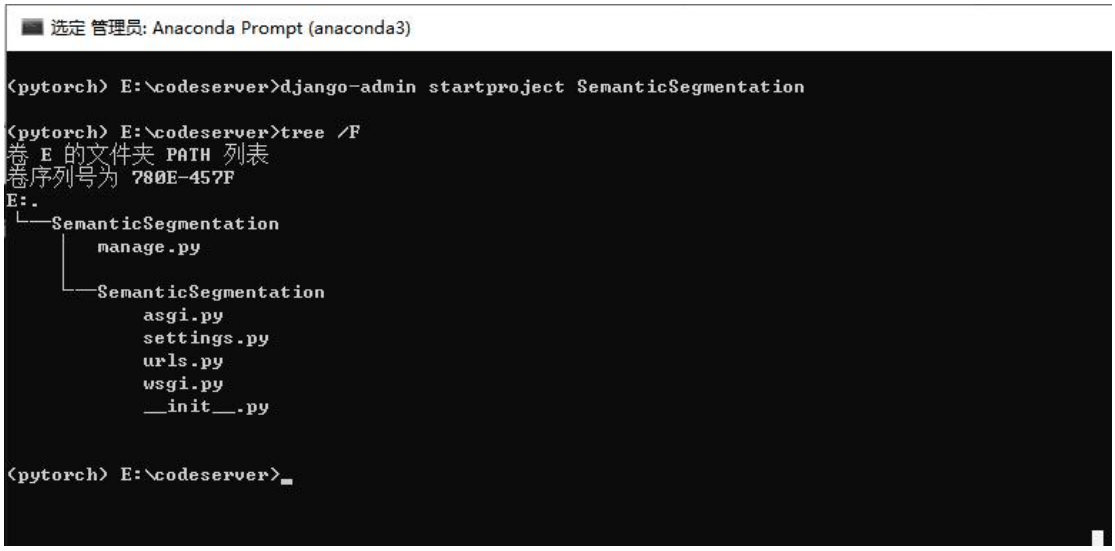



图 3.2 Django 项目目录

图 3.2 是 Django 项目目录，通过查看 Django 项目的目录，我们发现__init__.py 是一个未知的文件夹，它提示我们将 Python 包装在此，从而实现 Pythons 的功能。Settings.py 是此 Django 框架项目的配置。URLs.py 代表了 Django 框架的标识符，它也是该框架的网站的根目录。asgi.Py 提供了一个便捷的 web 服务，以便用户可以访问和使用 asgi 的项目。Wsgi.py 是一款兼容于 Wsgi 的 web 浏览器，它可以为您的项目提供一个便捷的入口。

（3）运行 Django 项目

```
python manage.py runserver
```

下面图 3.3 是启动 Django 项目过程，可以看到 Django 服务器启动在 ip 地址为 127.0.0.1，端口为 8000 的路径上，通过访问 http://127.0.0.1:8000/ 路径可

以访问

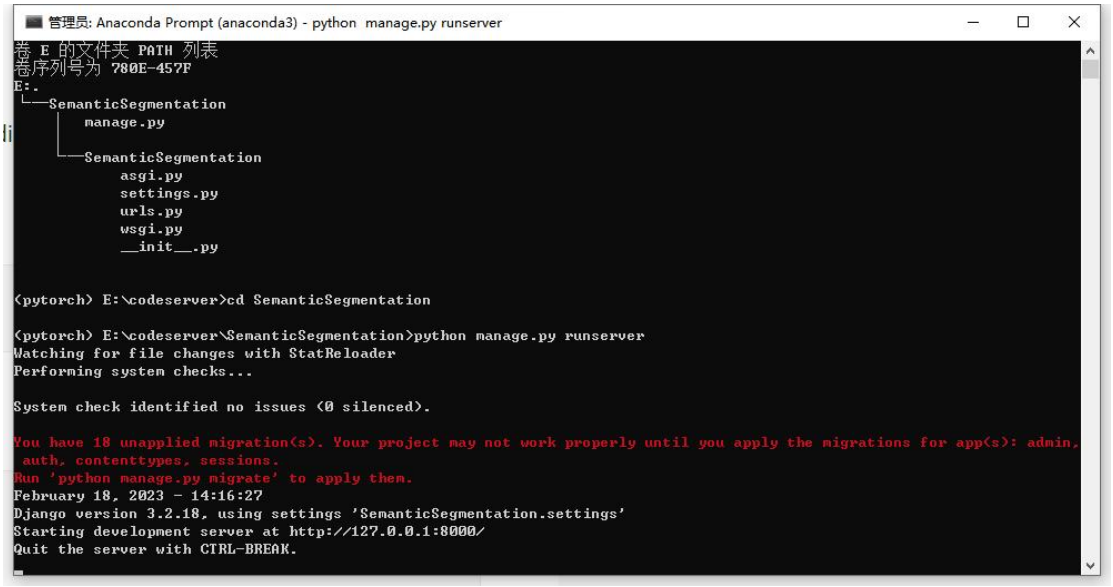



图 3.3 启动 Django 项目过程

下面图 3.4 是启动的 Django 服务器首页，我们需要将其改造成语义分割服务器。

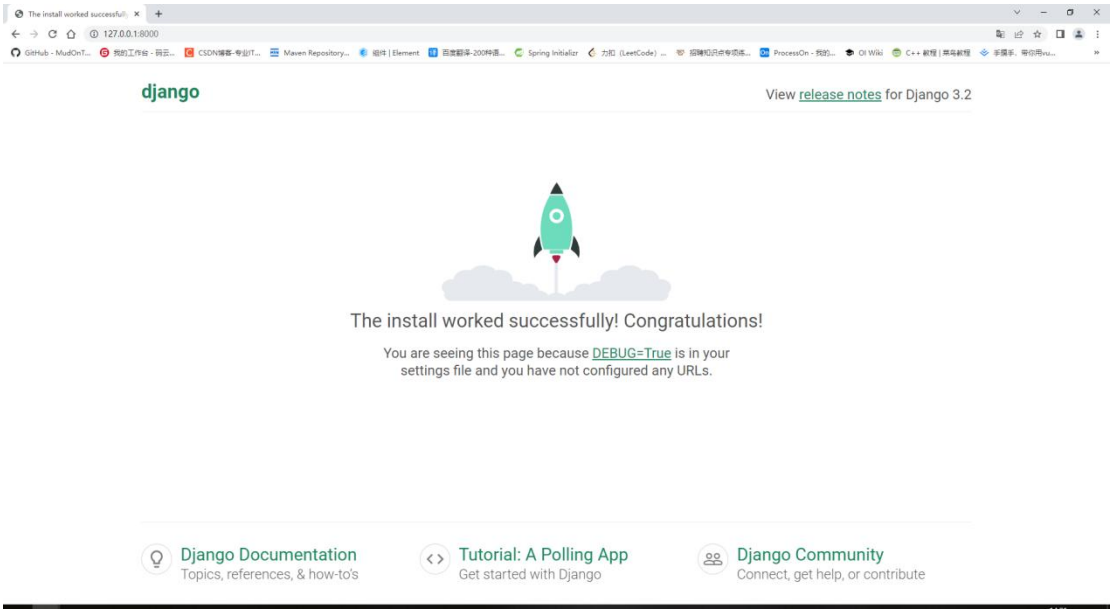



图 3.4 启动 Django 的 Django 服务器首页

（4）改造 Django 项目

由于算法操作费时比较久，所以采用的是异步操作，Django 服务器负责接收图片，上传图片到指定文件夹，并启动算法操作，然后直接返回，并启动定时任务周期性检测任务是否完成。算法操作完成后，会将处理后的图片和处理后的结果写入到指定位置。当写入完后，Django 服务器负责将结果反馈给用户。

# 4 看图写话系统的前端部分设计

本章主要讲的是看图写话系统前端部分设计，包括首页、注册页面、登录页面、提交图片页面、查询结果页面。其中每个页面都会以交互图的形式展示界面和后台交互流程。负责的页面如注册页面、登录页面还会以流程图的形式展示业务逻辑。

## 4.1 前端界面展示

### 4.1.1 首页


图 4.1　首页用户交互设计图

图 4.1 是首页用户交互设计图，当用户访问首页时，会在浏览器向 nginx 静态资源服务器请求其所需的资源，使用的 http 协议，使用 URL 来指定要请求的资源，http 协议实质上是一个文本协议，里面会有 get 请求类型，当 nginx 收到 get 请求类型，会解析 URL 从而找到要请求的资源，再通过 http 底层的 TCP 连接通道将资源发回去给前端。网页前端会解析 HTML 文件渲染界面。

下面图 4.2 是首页入场动画，有注册、登录和下拉按钮。首页入场动画是一个动态的背景动画，给用户一种比较好的体验。注册、登录按钮可以导航到注册页面、登录页面，使用超链接标签封装，当点击时会自动向 nginx 服务器发送 http 请求，就会返回相应的页面。下拉按钮会导航到下拉界面。


图 4.2　首页入场动画

下面图 4.3 是首页下拉后页面，首页下拉后，是看图写话系统简介，有轮播图和看图说话系统简介。轮播图中会有看图写话的示例，使用图文介绍的方式可以更好的介绍看图写话系统，给用户一种想要了解和使用该系统的冲动。

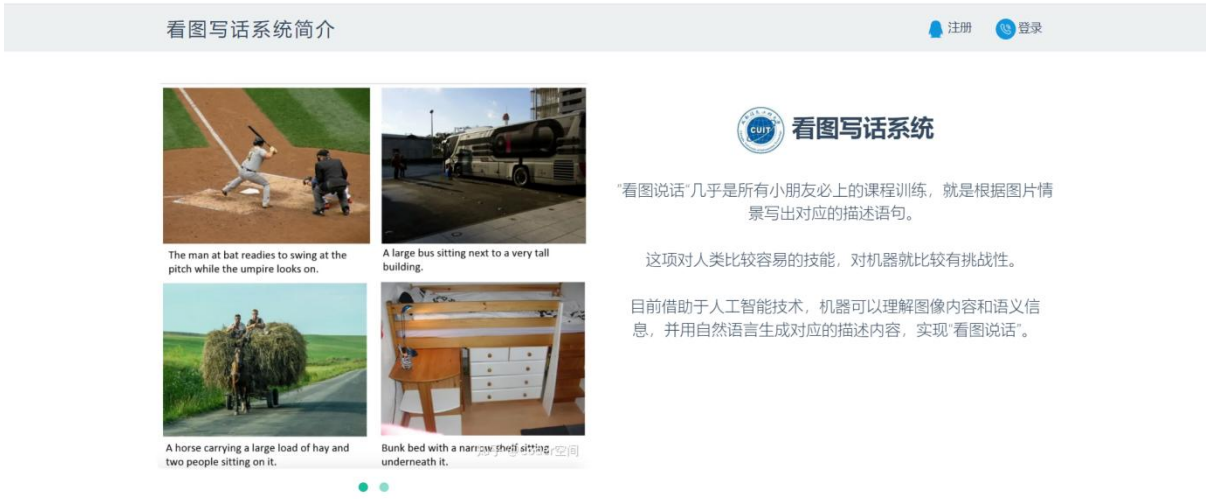



图 4.3　首页下拉后页面

### 4.1.2 注册页面

下面图 4.4 展示了一种注册交互图设计，用户可以先向 nginx 服务器发出注册请求，然后填写注册所需的信息，经过前端数据校验后，再将请求发送给后端服务器，后端服务器会根据校验结果，将注册信息存储到数据库中，最终返回注册成功的结果。


图 4.4　注册交互设计图

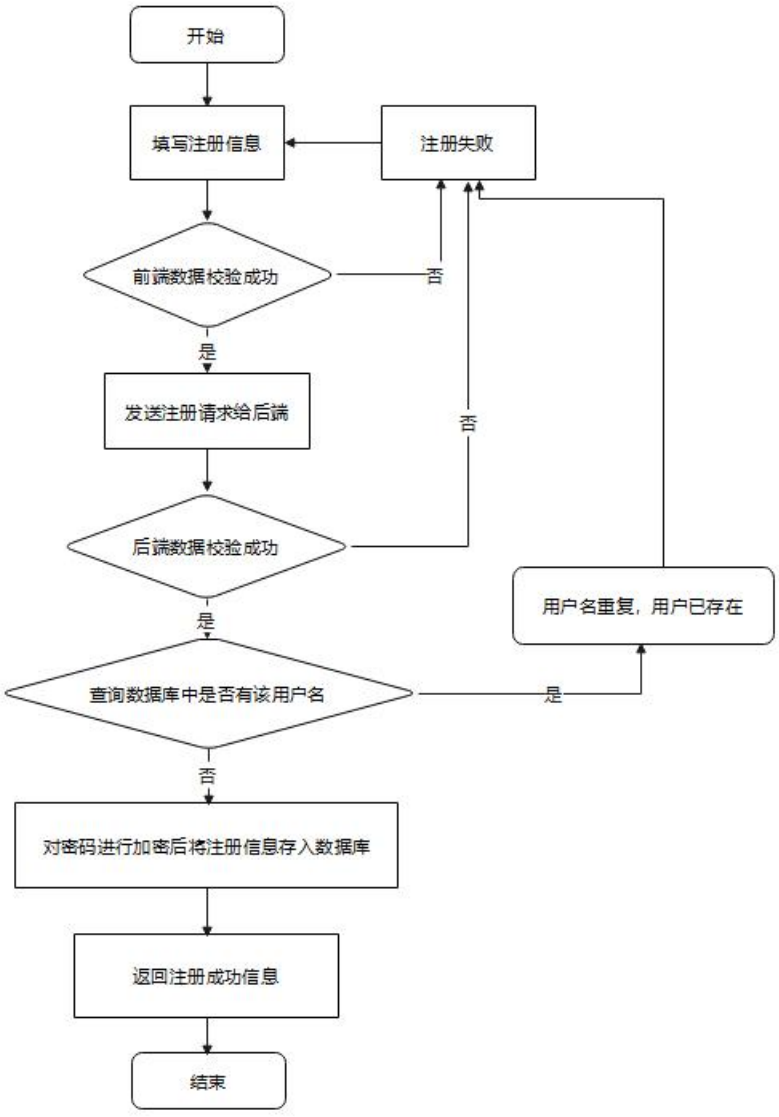



图 4.5 注册流程图

上面图 4.5 是注册流程图，当用户填写了信息后会先在前端进行数据校验，如果校验失败，直接返回注册失败，提示用户重新填写信息。当前端数据校验成功后则发送注册请求到后端，后端会再一次进行数据校验，防止特别的人直接绕过前端通过接口请求后端。当后端数据校验失败后直接返回注册失败，当后端数据校验成功后会去数据库查询该用户名是否已存在，如果存在则需提示用户名重复，用户已存在，并返回注册失败给用户。当数据库无该用户名，说明可以进行注册，会先对密码进行加密再存储，而不存储明文密码，防止密码泄露，最后把注册信息存储在数据库完成后，就返回注册成功信息给用户。

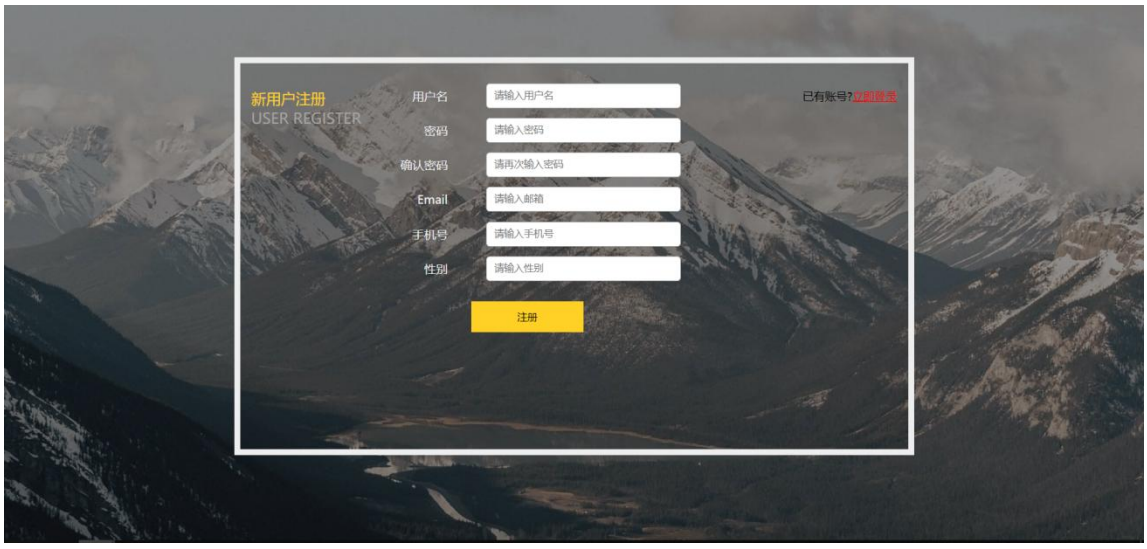



图 4.6 注册界面

上面图 4.6 是注册界面，用于注册用户。可以看到需要填写的信息有用户名、密码、确认密码、Email、手机号以及性别。

### 4.1.3 登录页面


图 4.7 登录交互设计图

图 4.7 是登录交互图设计，用户会先向 nginx 服务器请求登录页面，然后填写登录所需的信息，并发送请求给 nginx 服务器，nginx 充当网关将注册请求转发给后端服务器，后端服务器会查询是否有该用户并比对密码后，返回登录成功信息。

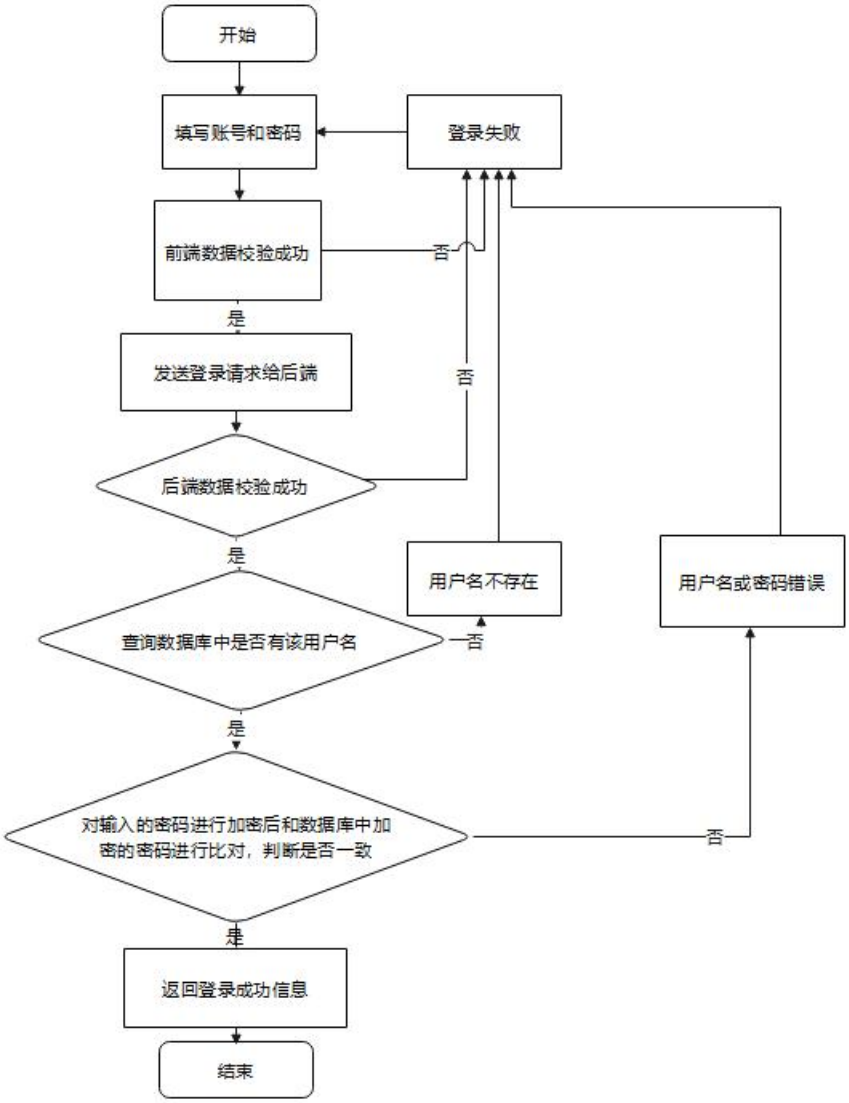



图 4.8 登录流程图

上面图 4.8 是注登录流程图，当用户填写了信息后会先在前端进行数据校验，如果校验失败，直接返回登录失败，提示用户重新填写信息。当前端数据校验成功后则发送登录请求到后端，后端会再一次进行数据校验，防止特别的人直接绕过前端通过接口请求后端。当后端数据校验失败后直接返回登录失败，当后端数据校验成功后会去数据库查询该用户名是否存在，如果不存在则需提示用户名不存在，并返回登录失败给用户。一旦用户名被存储在数据库中，系统就会对输入的密码进行加密，并且将其与数据库中的加密密码进行比对，如果比对结果正确，就会返回登录成功的消息。


图 4.9　登录界面

图 4.9 是登录界面，用于登录看图写话系统，用账号、密码输入框和登录按钮，后面是背景图，有动态特效和精美设计，给用户一种美好的登录体验。

### 4.1.4 提交图片页面


图 4.10　提交图片交互设计图

图 4.10 是提交图片交互图设计，用户首先向 nginx 服务器请求提交图片页面，选择图片后点击上传按钮，向 nginx 发送提交请求，nginx 转发该请求给后端服务器，后端服务器会先保存该图片，然后异步调用算服务器接口对图片进行处理和预测，然后直接返回，不等待算法处理完成。这样就将提交图片和算法处理两个过程异步开，使得用户体验得到极大的改善。

下面图 4.11 是提交图片页面，负责提交图片到后端服务器，这里的图片提交可以拖拽上传。拖拽图片到下面的框中，浏览器将会自动检索出文件的名称、类型、大小以及上一次更新的时间等细节，并将这些信息展示给用户。可以拖拽多张图片并展示。当点击上传后，就会向后端发送请求，最后会弹出提交成功的提示框。


图 4.11　提交图片页面

### 4.1.5 查询结果页面

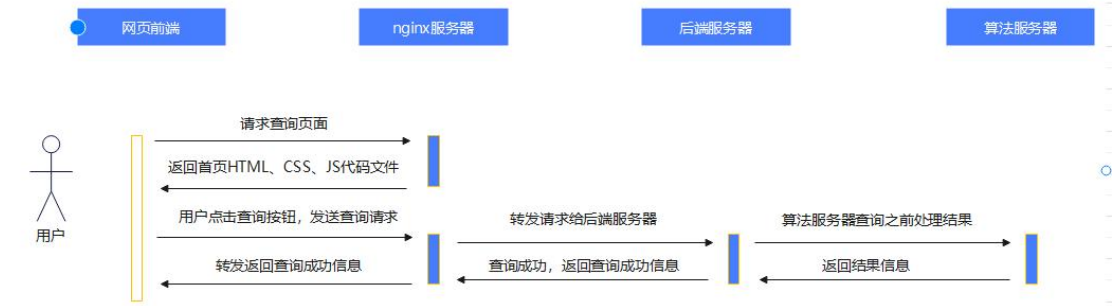



图 4.12　查询结果页面交互设计图

图 4.12 是查询结果页面交互设计图，用户首先会请求查询结果页面，当其点击查询按钮后，后端会去询问算法服务器之前处理的图片的处理信息，当算法服务器返回信息后则查询成功，会将查询到的信息渲染到前端界面。

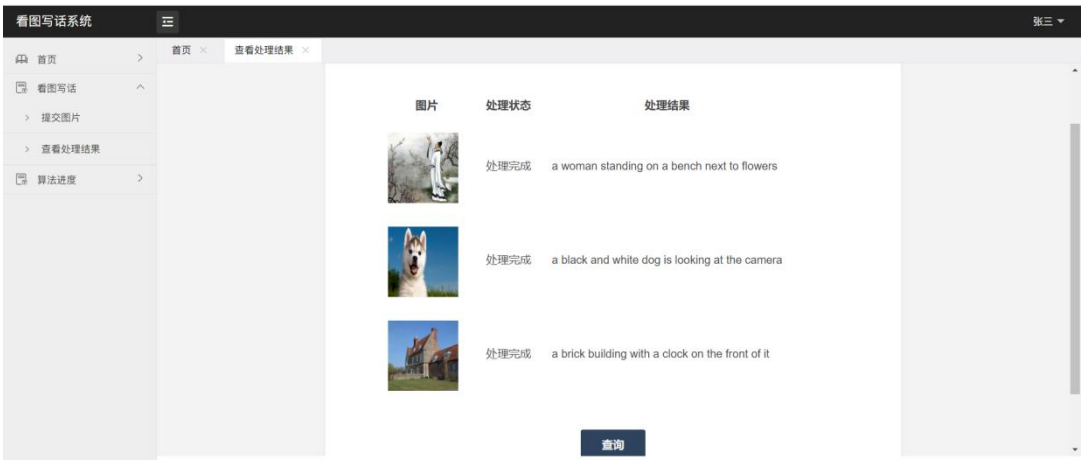



图 4.13　查询结果页面

图 4.13 是查询结果页面，查询提交的图片的处理状态和处理结果，当正在处理中会有倒计时缓解用户焦虑。

# 5 实验与测试

本章主要讲的是看图写话系统的实验与测试，主要分为三部分，测试系统流程、测试算法模型描述准确率、测试系统接口性能。测试系统流程的方法是通过点击前端页面，查看系统各个功能是否开发完备，流程是否通畅。测试算法模型的方法是选用三种不同难度的测试集输入算法模型中，得到对应的描述性语言，再统计描述准确率。测试系统接口性能的方法是使用专业的并发测试工具，测试系统在高并发的环境下，真实的处理性能。

## 5.1 测试系统流程

### 5.1.1 注册账户

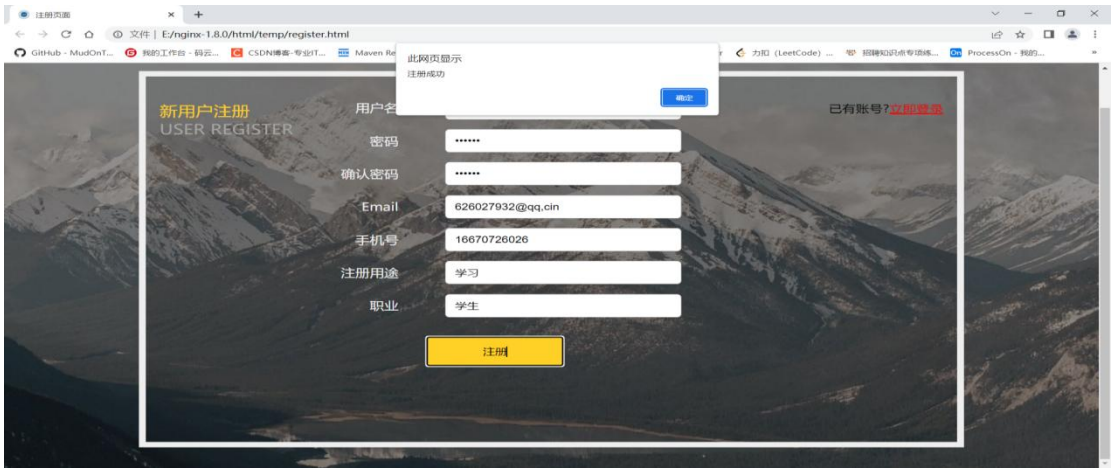



图 5.1 注册成功效果

图 5.1 是注册成功效果，点击注册按钮后，可以发现注册成功，注册的用户名为陈衍汀，填入 Email、手机号和注册用途、职业就可以完成注册了。

### 5.1.2 提交图片

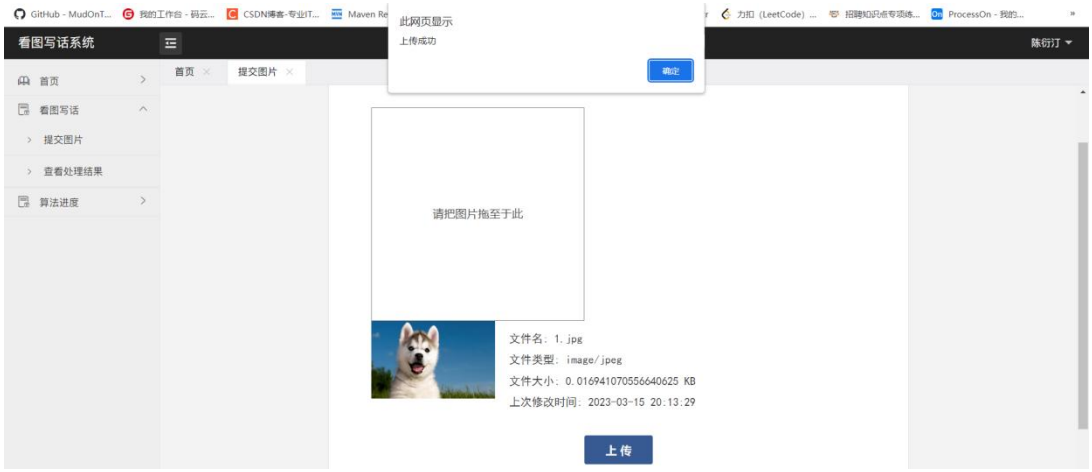



图 5.2 拖拽上传提交图片

上面图 5.2 是拖拽上传图片。这里我们拖拽了一张小狗坐在草地上的图片，点击提交按钮可以看到提交成功。

### 5.1.3 查询处理结果

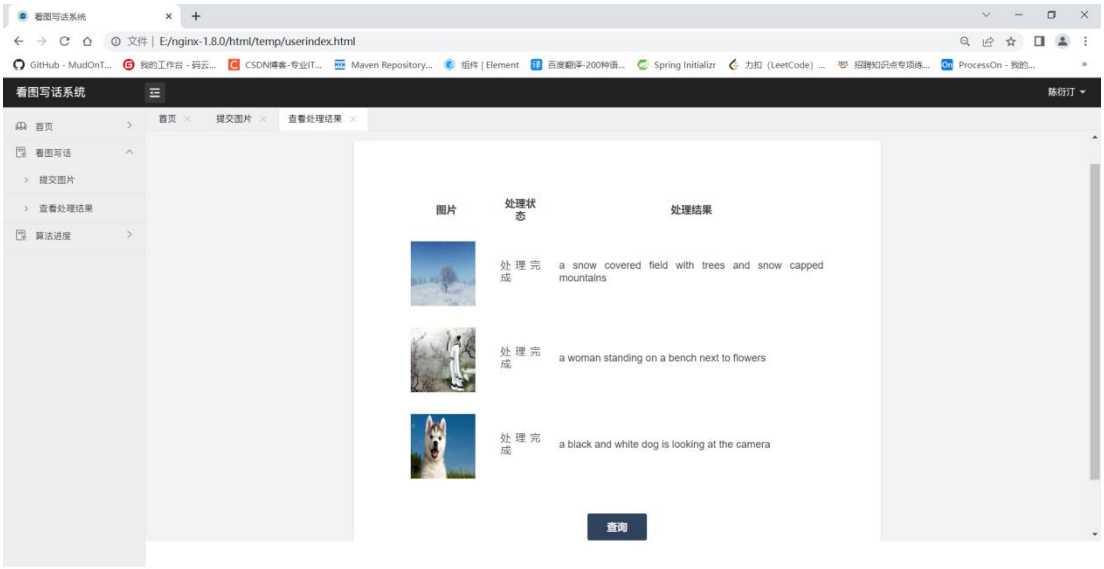



图 5.3 查询处理结果

图 5.3 是查询处理结果，点击查询按钮，可以看到刚刚我们提交的图片已经处理完成了，并显示了处理结果：a black and white dog is looking at camera，该预测结果是符合图片描述的，甚至预测到了图片中没有的但是却符合逻辑的行为。

## 5.2 测试算法准确率

### 5.2.1 简单图片测试集

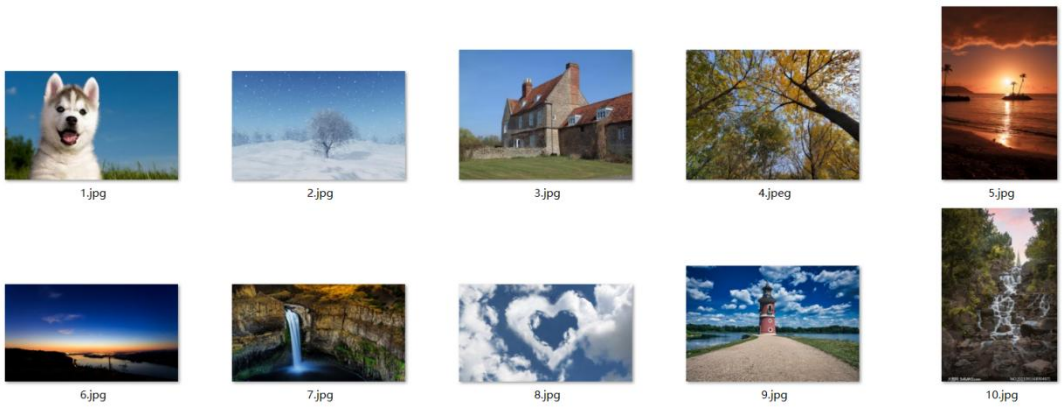
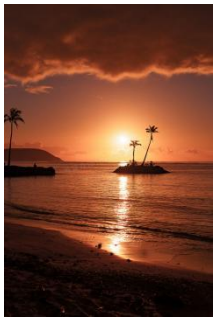



图 5.4 简单图片测试集

图 5.4 是简单图片测试集，选用的都是一些容易理解的图片，图片比较简单，只有一两个对象，对象之间的行为也是比较简单的。

表 3　简单图片测试集处理结果

| 图片 | 英文描述 | 中文描述 | 是否准确 |
| --- | --- | --- | --- |
|  | a black and white dog is looking at the camera | 一只黑白相间的狗正在看着摄像机。 | 准确 |
|  | a snow covered field with trees and snow capped mountains | 白雪皑皑的田野，有树木和白雪覆盖的山脉 | 准确 |
|  | a brick building with a clock on the front of it | 前面有钟的砖砌建筑 | 准确 |
|  | a tree filled with lots of leaves in the middle of a forest | 森林中央长满树叶的树 | 准确 |
|  | a sunset on a beach with a boat in the distance | 海滩上的日落，远处有一艘船 | 准确 |
|  | a sunset in the distance of a harbor with a lighthouse | 远处有灯塔的港口的日落 | 准确 |
|  | a waterfall that is in the middle of a waterfall | 一缕瀑布在这整个瀑布的中央 | 准确 |
|  | an airplane flying through a cloudy sky | 在多云的天空中飞行的飞机 | 不准确 |
|  | a tall tower with a clock on top of it | 塔顶有钟的高塔 | 准确 |

| | | | |
|---|---|---|---|
|  | a river filled with lots of water and trees | 一条布满树木的河流 | 准确 |

表 3 是简单图片测试集处理结果，可以看到图像字幕算法模型在处理简单图片时准确率十分高，可以达到百分之九十，不但可以识别其中元素还可以描述清楚其的简单行为。

### 5.2.2 中等图片测试集

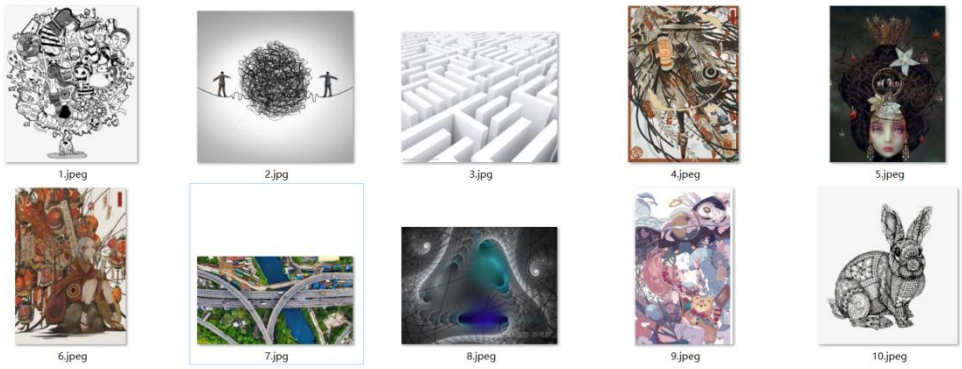



图 5.5　中等图片测试集

上面图 5.5 是中等图片测试集，中等图片选用的都是些看着元素不算多，但是具有迷惑性的图片，可以迷惑算法的分类和识别。

表 4 中等图片测试结果

| 图片 | 英文描述 | 中文描述 | 是否准确 |
|---|---|---|---|
|  | a figurine of a giraffe on top of a table | 桌子上的长颈鹿雕像。 | 不准确 |
|  | a man standing on top of a white object | 一个男人站在一个白色物体上 | 准确 |

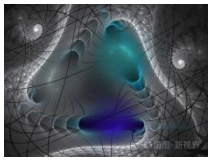
| | | | |
|---|---|---|---|
|  | a row of black and white pictures on a wall | 墙上的一排黑白画 | 不准确 |
|  | a figurine of a bird sitting on top of a table | 坐在桌子上的鸟的雕像 | 不准确 |
|  | a woman in a dress with a flower in her hair | 一个穿着裙子、头发上插着花的女人 | 准确 |
|  | a statue of an elephant with a basket on its head | 头上戴着篮子的大象雕像 | 不准确 |
|  | a city street filled with lots of traffic | 车水马龙的城市街道 | 准确 |
|  | a blue sky with a blue sky background | 以蓝天为背景的白云 | 不准确 |
|  | a collage of photos of people and animals | 人和动物的照片拼贴 | 准确 |
|  | a teddy bear sitting on top of a table | 坐在桌子顶上的泰迪熊 | 不准确 |

表 4 是中等图片测试结果，可以看到图像字幕算法模型在处理中等难度图片时，对于这种元素不算太多，但具有迷惑性的图片时效果不好。

### 5.2.3 复杂图片测试集

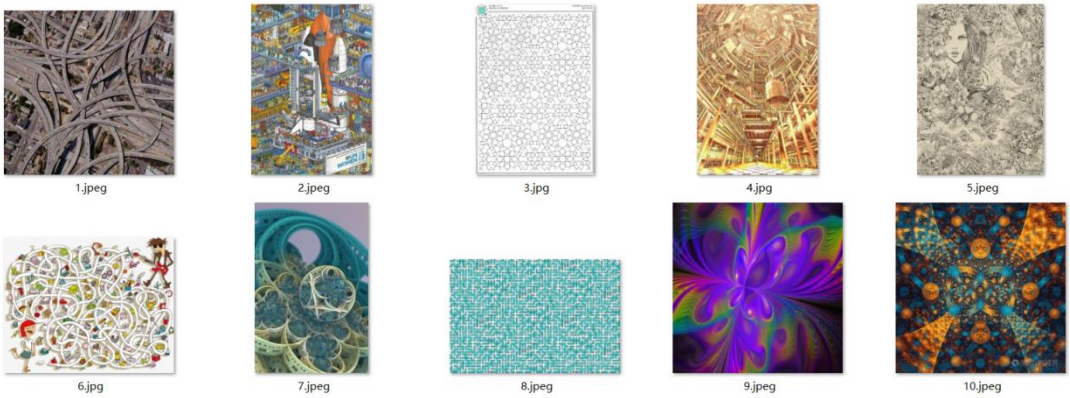
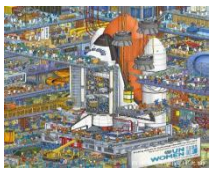
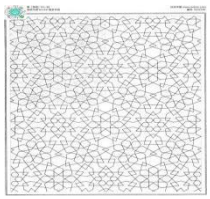



图 5.6　复杂图片测试集

图 5.6 是复杂图片测试集，复杂图片选用的都是些图片元素十分复杂，连人都不太好描述的图片，属于抽象类的图片。

表 5 复杂图片测试集处理结果

| 图片 | 英文描述 | 中文描述 | 是否准确 |
|---|---|---|---|
|  | a city street filled with lots of traffic | 车水马龙的城市街道 | 准确 |
|  | a model of a train on display in a museum | 博物馆里展出的火车模型 | 不准确 |
|  | a photo taken from the inside of a wall | 从墙内侧拍摄的照片 | 不准确 |
|  | a room filled with lots of different types of objects | 房间里摆满了许多不同类型的物品 | 准确 |

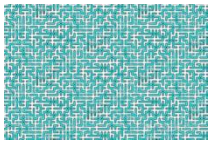
| | | | |
|---|---|---|---|
|  | a young child laying in a pile of dead animals | 躺在一堆动物尸体中的小孩 | 准确 |
|  | a collage of many different types of christmas decorations | 许多不同类型的圣诞装饰品的拼贴画 | 准确 |
|  | a blue vase filled with colorful flowers on top of a table<br>1 | 桌子上放着一个蓝色的花瓶，花瓶里摆满了五颜六色的花 | 准确 |
|  | a series of photos showing a variety of different colored flowers | 一系列展示各种不同颜色花朵的照片 | 不准确 |
|  | a blurry photo of a painting of a purple flower | 一幅紫色花朵的模糊照片 | 准确 |
|  | a blue and white patterned patterned rug on a tile floor | 瓷砖地板上的蓝白花纹地毯 | 准确 |

表 5 是复杂图片测试集处理结果，看到图像字幕算法在处理复杂图片这种图内元素十分复杂，且行为混乱且抽象的图片时，不能很好的描绘图片。

## 5.3 测试系统性能

（1）测试注册接口

| Label | # Samples | Average | Min | Max | Std. Dev. | Error % | Throughput | Received KB/sec | Sent KB/sec | Avg. Bytes |
|---|---|---|---|---|---|---|---|---|---|---|
| Thread Group:... | 60000 | 285 | 5 | 3944 | 264.05 | 0.00% | 691.6/sec | 169.99 | 104.68 | 251.7 |
| TOTAL | 60000 | 285 | 5 | 3944 | 264.05 | 0.00% | 691.6/sec | 169.99 | 104.68 | 251.7 |

**图 5.7 注册接口测试数据**

图 5.7 是注册接口测试数据，由图可知其吞吐量是是 681.6 个事务每秒，其平均响应时间是 285 毫秒，由于注册需要写入数据库，并进行加密等操作，所以耗时较长，但响应也十分的快只需 0.3 秒即可注册成功，达到系统性能标准。

（2）测试登录接口

| Label | # Samples | Average | Min | Max | Std. Dev. | Error % | Throughput | Received KB/sec | Sent KB/sec | Avg. Bytes |
|---|---|---|---|---|---|---|---|---|---|---|
| HTTP Request | 307955 | 30 | 0 | 1142 | 72.14 | 0.06% | 6380.1/sec | 718.50 | 741.00 | 115.3 |
| TOTAL | 307955 | 30 | 0 | 1142 | 72.14 | 0.06% | 6380.1/sec | 718.50 | 741.00 | 115.3 |

图 5.8　登录接口测试数据

图 5.8 是登录接口测试数据，由图可知其吞吐量是 6380.1 个事务每秒，其平均响应时间是 30 毫秒，登录操作由于只需查询数据库，并且使用了缓存加速，所以比较快，响应十分快只需 0.03 秒，达到系统性能标准。

（3）测试提交图片接口

| Label | # Samples | Average | Min | Max | Std. Dev. | Error % | Throughput | Received KB/sec | Sent KB/sec | Avg. Bytes |
|---|---|---|---|---|---|---|---|---|---|---|
| HTTP Request | 54404 | 711 | 0 | 2344 | 666.15 | 5.63% | 280.5/sec | 69.75 | 30.76 | 254.6 |
| TOTAL | 54404 | 711 | 0 | 2344 | 666.15 | 5.63% | 280.5/sec | 69.75 | 30.76 | 254.6 |

图 5.9 提交图片接口测试数据

图 5.9 是提交图片接口测试数据，由图可知其吞吐量是 280.5 个事务每秒，其平均响应时间是 711 毫秒，由于提交图片需要将图片上传到服务器，提交图片的时间取决于图片的大小，所以吞吐量不会很高，所以该吞吐量达到系统性能标准。

（4）测试查询结果接口

| Label | # Samples | Average | Min | Max | Std. Dev. | Error % | Throughput | Received KB/sec | Sent KB/sec | Avg. Bytes |
|---|---|---|---|---|---|---|---|---|---|---|
| HTTP Request | 1119278 | 54 | 0 | 2395 | 163.58 | 0.05% | 3686.0/sec | 365.07 | 428.12 | 101.4 |
| TOTAL | 1119278 | 54 | 0 | 2395 | 163.58 | 0.05% | 3686.0/sec | 365.07 | 428.12 | 101.4 |

图 5.10 查询结接口测试数据

图 5.10 是注册接口测试数据，由图可知其吞吐量是是 3686 个事务每秒，其平均响应时间是 54 毫秒，查询结果操作由于只需查询数据库，并且使用了缓存加速，所以比较快，响应十分快只需 0.05 秒，达到系统性能标准。

## 5.4 本章小结

完成了看图写话系统整体功能的测试，前端界面达到了界面优美、性能流程的指标，后端服务器达到了响应速度快，响应时间短，除个别接口大部分接口响应时间都在 50 毫秒左右，算法在识别相对简单的图片也达到了百分之九十的识别准确率，综上所述，看图写话系统完成了课题最初设定的性能指标要求，并且实现了所有功能。但依然发现了系统中存在的一些不足之处，比如算法模型在描述复杂图片时准确率较低，后端系统在注册和提交图片接口的吞吐量较低，处理耗时较长，前端系统在提示用户如何使用仍然欠缺，这些都是这个系统可以进行后续跟进与发展的地方。

# 结　论

本课题完成了看图写话系统，包括前端界面的设计和开发，后端服务器的架构和业务代码的编写，以及算法模型的选用和搭建算法微服务器，最终实现了看图写话系统，其主要功能是可以在网页方便的提交图片，并获取算法的处理结果即图片的描述信息，该系统作为一个中间平台，也暴露了使用接口，方便各类人士的各类需求，AI 初学者可以来该平台体验该算法功能，开发人员也能通过调用暴露的接口来给自己的平台外接看图写话功能，甚至可以用来幼儿教育，幼师们可以让孩子们登录在该平台中和 AI 根据图片进行相互交流，让孩子们对 AI 产生兴趣为以后的人生规划打下扎实基础。

本研究的主要贡献：

此系统的功能实现部分采用模块化设置，分模块设计编写的程序降低了程序复杂度、方便单个模块功能调试，模块化的设计让后期的调试工作轻松了不少。由于实现该系统的 python、java、HTML 都具有跨平台的功能，所以此系统的可移植性高，能够在多个操作系统上进行实现。该系统采用的接口化设计可以方便外部系统调用，具有易扩展，可扩展的能力。该系统界面优美，操作简单，可以让更多人的对 AI 产生兴趣，从而去学习和开发 AI，为新时代注入新鲜的血液。

课题展望：

本研究实现了课题要求，但碍于本人自身的知识储备和硬件的性能限制，在算法部分不能实现真正的对任意图片进行准确的描述，以及算法模型的处理时间也比较长，后续可以增加模型的训练样本，改进模型的处理方式来提高算法模型的描述准确率和处理效率，降低处理功耗。在前端部分上未设计使用指导流程，没有降低看图写话系统的使用难度。在后端部分上系统业务逻辑仍然存在可以优化的空间，可以降低业务逻辑的处理时间，并加快响应速度。

综上，本文所论述的基于视觉与文本技术的看图写话系统设计与实现虽然完成了课题设计要求，但仍存在许多不足，有待改进。由于时间紧张，再加上个人能力有限，不足的地方在所难免，但在日后的工作与学习中，本人将会继续学习并完善。

# 参考文献


[1] Cho K,Van Merriënboer B,Bahdanau D,et al．On the properties of neural machine translation: Encoder-decoder approaches[J]：arXiv preprint，2014

[2] 李德毅，于剑，中国人工智能学会，中国科协新一代信息技术系列丛书人工智能导论[M]，中国科学技术出版社，2018.08,168

[3] Oriol Vinyals, Alexander Toshev, Samy Bengio, Dumitru Erhan.Show and Tell: A Neural Image Caption Generator[J];

[4] 赵京胜,宋梦雪,高祥.自然语言处理发展及应用综述[J].信息技术与信息化,2019(07):142-145.

[5] Milan Sonka,Vadav Hhvac,Roger Boyle£.Image Processing,Amdy-sis,and Machine Vision[M].人民邮电出版社。2003

[6] 魏伟波，芮筱亭．图像分割方法研究[J]．计算机工程与应用。2002.24(6):91-94

[7] 薛景浩，章毓晋，林行刚等．基于特征散布的图像 FCM 聚类分割[J].模式识别与人工智能。1998，11（4）：462-467

[8] 李长云，王志兵,智能感知技术及在电气工程中的应用[J],电子科技大学出版社,2017.05,163

[9] 赵园丁.谈人工智能时代背景下自然语言处理技术的发展应用[J].办公自动化,2019,24(10):63-64.

[10] 李雪晴，王石，王朱君，朱俊武．自然语言生成综述[J]．计算机应用. 2018

[11] Sutskever I, Vinyals O, Le Q V．Sequence to sequence learning with neural networks[C]：Advances in neural information processing systems，2014

[12] Lisa Anne Hendricks, Subhashini Venugopalan, Marcus Rohrbach, Raymond Mooney, Kate Saenko, Trevor Darrell.Deep Compositional Captioning: Describing Novel Object Categories without Paired Training Data[J].2019

[13] SubhashiniVenugopalan, LisaAnne Hendricks, Marcus Rohrbach, Raymond Mooney, Trevor Darrell, Kate Saenko.Captioning Images with Diverse Objects[J].

[14] Gehring J, Auli M, Grangier D, et al．Convolutional sequence to sequence learning[C]：Proceedings of the 34th International Conference on Machine Learning，2017

[15] Vaswani A, Shazeer N, Parmar N, et al．Attention is all you need[C]：Advances in neural information processing systems，2020

[16] 罗枭.基于深度学习的自然语言处理研究综述[J].智能计算机与应用,2020,10(04):133-137.

[17] 王飞,陈立,易绵竹,谭新,张兴华.新技术驱动的自然语言处理进展[J].武汉大学学报(工学版),2018,51(08):669-678.

[18] 刘维.负载均衡技术在电子商务网站中的应用研究[D].湖南:湖南大学,2018.

[19] 刘立斌.基于负载均衡技术的研究及展望[J].市场周刊,2017,(10):140-142.

[20] Kenji, Suzuki,Kunio, Doi.How can a massive training artificial neural network (MTANN) be trained with a small number of cases in the distinction between nodules and vessels in thoracic CT?[J].Academic radiology.2005,12(10).1333-41.

[21] Pascal Vincent,Hugo Larochelle,Yoshua Bengio.Extracting and Composing Robust Features with Denoising Autoencoders[C].2019.

[22] Yann LeCun,Marco Scoffier,Koray Kavukcuoglu, 等 .Learning Long-Range Vision for Autonomous Off-Road Driving[J].Journal of Field Robotics..2009,26(2).
[23] Simon Osindero,Yee-Whye Teh,Geoffrey E. Hinton.A Fast Learning Algorithm for Deep Belief Nets[J].Neural Computation.2006,18(7).1527-1554.
[24] P.Haffner,L. Bottou,Y. Bengio, 等 .Gradient-based learning applied to document recognition[J].Proceedings of the IEEE.1998,86(11).2278-2324.
[25] LeCun, Y,Boser, B,Denker, J, 等 .Backpropagation Applied to Handwritten Zip Code Recognition[J].Neural Computation.1989,1(4).541-551.DOI:10.1162/neco.1989.1.4.541.
[26] 王俊杰.基于加速推断策略的深度学习算法[J].武汉科技大学.2015.
[27] 郭丽丽.基于深度学习的图像分类方法研究[J].中国矿业大学.2016.
[28] 马金荣.复杂环境下的机器人视觉图像增强算法的研究[J].燕山大学.2017.
[29] 于立强.基于机器视觉的手写数字识别与目标追踪技术研究[J].燕山大学.2019.
[30] 陈诚.PCA-PSO-FCM 在短文本聚类中的研究与应用[D].2020.

# 致 谢

四年的学习生涯转眼间就要结束，我深深地感激那些曾经为我提供支持、指导和鼓舞的老师、朋友们，他们也是我最珍贵的家人。我对这四年来给予我们指导和帮助的老师们表示感谢。他们为我们提供了宝贵的知识和指导。

特别是王伟老师对我的指导和支持让我受益匪浅，他的精神和智慧深深地影响了我，使我在学术和人生上取得了巨大的成就，这是一种永恒的回忆。

在我的大学生涯中，您是我遇见的最好的老师之一。每一堂课上您总是站在讲台前，无论是讲解还是提问都是非常有条理和深度的。更为重要的是，您注重理论与实践相结合，在教学中很少使用空洞的例子，每一个问题都是经过严谨的思考和深入的研究。这让我在学习过程中感受到了学习的快乐和扩宽思路的重要性。

除了在课堂上进行教学，您还会给我们讲解一些个人的学术成果和做学问的理念。您的渊博知识和深入调查的作风，以及您对学术事业的坚定执着，都极大地鼓舞了我下定决心放手去做学习新技术。这也是成功的第一步，谢谢您。

除了学术方面的指导，您还在我人生的关键时刻提供了帮助和支持。我曾经遇到了一些困难并因此感到迷茫和失落，但是您的鼓励和建议让我重新找到了信心和前行的方向。不仅如此，您还鼓励我积极参加实习公司项目和多学习新技术，以后好为国家添砖加瓦，这种精神，我深深地在心中体会到了，谢谢您。

最后，我要再次感谢您在我的大学生涯中所给予我众多的帮助和鼓励。您的言传身教对于我的成长影响深远，永远铭刻在我的心里。今天我即将走出学习并成为一名优秀的职场人员，是您给予我的关心和信任，让我得以不断地提升自己，直到达到今天这个水平。作为一个学生，我不知道如何回报您的教诲和帮助，唯有向您深情致谢，并向您发出珍贵感恩的思念之情。再次感谢您，祝福您能够健康快乐地度过每一天。

作者简介：

姓　　名：陈衍汀　　性别：男

出生年月：2000 年 9 月　　民族：汉族

Email：626027932@qq. com

# 声 明

本论文的工作是2022年12月至2023年5月在成都信息工程大学自动化学院完成的。文中除了特别加以标注的地方外，不包含他人已经发表或撰写过的研究成果，也不包含为获得成都信息工程大学或其他教学机构的学位或证书而使用过的材料。

关于学位论文使用权和研究成果知识产权的说明：

本人完全了解成都信息工程大学有关保管使用学位论文的规定，其中包括：

（1）学校有权保管并向有关部门递交学位论文的原件与复印件。

（2）学校可以采用影印、缩印或其他复制方式保存学位论文。

（3）学校可以学术交流为目的复制、赠送和交换学位论文。

（4）学校可允许学位论文被查阅或借阅。

（5）学校可以公布学位论文的全部或部分内容（保密学位论文在解密后遵守此规定）。

除非另有科研合同和其他法律文书的制约，本论文的科研成果属于成都信息工程大学。

特此声明！

作者签名：陈衍汀

2023年05月14日