湖南大學

HUNAN UNIVERSITY

本科生毕业设计(论文)

设计(论文)题目：应用深度学习算法的工业锅炉智能温度控制系统

学生姓名：陈芷安

学生学号：201808010805

专业班级：智能 1802 班

学院名称：信息科学与工程学院

指导老师：佘兢克

学院院长：李肯立

2022 年 5 月 日

湖 南 大 学

毕业设计（论文）原创性声明

本人郑重声明：所呈交的设计（论文）是本人在导师的指导下独立进行研究所取得的研究成果。除了文中特别加以标注引用的内容外，本论文不包含任何其他个人或集体已经发表或撰写的成果作品。对本文的研究做出重要贡献的个人和集体，均已在文中以明确方式标明。本人完全意识到本声明的法律后果由本人承担。

学生签名：　　日期：2022 年 5 月 日

毕业设计（论文）版权使用授权书

本毕业设计（论文）作者完全了解学校有关保留、使用设计（论文）的规定，同意学校保留并向国家有关部门或机构送交设计（论文）的复印件和电子版，允许设计（论文）被查阅和借阅。本人授权湖南大学可以将本设计（论文）的全部或部分内容编入有关数据库进行检索，可以采用影印、缩印或扫描等复制手段保存和汇编本设计（论文）。

本设计（论文）属于

1、保　密□，在_____年解密后适用本授权书。

2、不保密□。

（请在以上相应方框内打“√”）

学生签名：　　日期：2022　年 5 月　　日

导师签名：　　日期：2022　年 5 月　　日

应用深度学习算法的工业锅炉智能温度控制系统

摘要

锅炉在工业生产中有广泛的应用，而在工业锅炉相关研究中，温度控制是一项重要的课题。已有的一些传统控制办法，如经典 PID 控制，具有精度不高、实时性不好的缺点，鲁棒性不强，不能适应复杂工况。在这种情况下，基于模糊控制的模糊控制器对此作了一些优化；在现阶段，采用对计算能力要求较高的实时的深度学习神经网络预测的方法进行的辅助 PID 控制已经成为可能。

本设计首先在 MATLAB Simulink 平台上借助平台提供的功能模块搭建了一个 PID 控制系统；同时，使用 MATLAB 程序设计语言实现了 Simulink 平台的 TCP/IP 客户端通信功能，能够将控制误差传递给本机 Python 程序，同时接收远 Python 程序预测模块部分传回的预测结果，用于 PID 控制。

其次，本设计完成了一个基于历史温度误差数据进行预测的 TensorFlow LSTM 预测模块和 Python Socket TCP/IP 通信模块，实时地接收 Simulink 仿真系统传来地温度误差，执行预测后，将预测结果送回 Simulink PID 进行优化控制。

本设计在基于数学推导的前提下，以锅炉温度控制为情景，证实了采用误差预处理函数对控制系统原始误差进行一定的补偿和处理能够优化 PID 的控制结果。优化后本设计的 PID 控制系统的上升时间 t_r、峰值时间 t_p 以及调节时间t_s 缩短为原始值的约 40%，同时具备了更小的超调量，而不需要动态地调节 PID 参数，以应对实际工况下 PID 整定完成后难以调节的问题。同时，初步探索了误差优化函数的可能形式，给出了被控对象传递函数较为简单的情形下的有效的误差优化函数设计方式，为后续可能的进一步研究打下了基础。

关键词：PID　锅炉温度控制　神经网络　LSTM　误差处理

Intelligent Industrial Boiler Temperature Control System using Deep Learning Algorithm

Abstract

Boilers are widely used in industrial production, and temperature control is an important subject in the related research of industrial boilers. Some existing traditional control methods such as classical PID control have the shortcomings of low precision, poor real-time performance, and low robustness, so they cannot adapt to complex working conditions. In this case, the fuzzy controller which based on fuzzy algorithm has made some optimizations; at this stage, auxiliary PID control has become possible by using the real-time deep learning neural network prediction method which requires high computing power.

In this design, a PID control system is first built on the MATLAB Simulink platform with the help of the functional modules provided by the platform. at the same time, the MATLAB programming language is used to implement the TCP/IP client communication function of the Simulink platform, which can transmit the control error to the native Python code. The program also receives the prediction results returned by the prediction module part of the remote Python program for PID control.

Secondly, this design completes a TensorFlow LSTM prediction module and Python Socket TCP/IP communication module for prediction based on historical temperature error data, which receives the temperature error from the Simulink simulation system in real time, and sends the prediction results back to Simulink after the prediction is performed. PID for optimal control.

Based on the premise of mathematical derivation, this design takes boiler temperature control as a scenario, and it is confirmed that using the error preprocessing function to compensate and process the original error of the control system can optimize the PID control results. After optimization, the rise time t_r, peak time t_p and adjustment time t_s of the PID control system of this design are shortened to about 40%

of the original value, and at the same time have a smaller overshoot, without the need to dynamically adjust the PID parameters to deal with the problem that it is difficult to adjust after PID tuning is completed under actual working conditions. At the same time, the possible forms of the error optimization function are preliminarily explored, and an effective error optimization function design method under the condition that the transfer function of the controlled object is relatively simple is given, which lays a foundation for the possible further research in the future.

Key words: PID boiler temperature control Neural network LSTM error processing

目录

毕业论文原创性声明和毕业论文版权使用授权书 I
摘要 II
Abstract III
1 绪论 1
1.1 课题学术背景及意义 1
1.2 国内外相关研究进展 2
1.3 研究内容 4
1.4 文章组织结构 5
2 关键技术背景知识介绍 6
2.1 神经网络 6
2.1.1 RNN 与 LSTM 6
2.1.2 TensorFlow Keras 9
2.2 MATLAB 与 Simulink 11
2.2.1 MATLAB 语言 12
2.2.2 Simulink 平台 12
2.2.3 在 Simulink 中使用 MATLAB 语言 13
2.3 本章小结 14
3 系统架构与优化控制系统实现 15
3.1 系统需求分析 15
3.2 模块实时通信框架 15
3.3 可行性分析 16
3.3.1 TCP/IP 连接建立 16
3.3.2 数据类型与数据转换 17
3.3.3 LSTM 数据集 18
3.4 PID 优化控制系统结构 18
3.5 简单工况下传递函数的推导 20
3.6 PID TCP/IP 通信模块实现 22
3.7 本章小结 23

4 LSTM 相关模块的设计与实现.....24
4.1 LSTM 预测模块结构.....24
4.2 LSTM 数据集相关探究.....25
4.2.1 传统 PID 的不足与改进对策.....25
4.2.2 数据集获取.....27
4.3 LSTM 训练.....28
4.4 Python socket TCP/IP 通信模块.....30
4.5 本章小结.....31
5 设计验证.....32
5.1 仿真.....32
5.2 数据分析.....33
5.3 本章小结.....36
6 总结与展望.....37
参考文献.....39
致谢.....41

1 绪论

1.1 课题学术背景及意义

锅炉在工业生产中有广泛的应用，而在工业锅炉相关研究中，温度控制是一项重要的课题[1]。传统的控制办法，存在精度不高、实时性不好的缺点，控制系统鲁棒性不太强，不能适应复杂工况。在这种情况下，基于模糊控制的模糊控制器对传统控制方式作了一些优化；在现阶段，采用对计算能力要求较高的实时的深度学习神经网络预测的方法进行的辅助 PID 控制已经成为可能。

我国工业锅炉目前主要以燃煤、燃油和电热锅炉为主[2]。电热锅炉本身使用电能，不会对环境造成直接的污染，但目前我国火力发电依然是主流[2]，电锅炉提高了能量转换效率，但燃煤产生的有毒有害气体排放对环境依然造成了不小的负担[3]。进入新发展时代，低碳发展、绿色发展的理念深入人心，相关研究也一直没有停止。关于锅炉如何节约燃料或者电能，一个有效的办法就是更好地控制锅炉温度，减少锅炉不必要的能量消耗，从而达到节能减排的目的。

目前，国内对工业电热锅炉温度控制的基础策略是经典控制理论（如 PID 控制）结合模糊控制（fuzzy control）等手段进行改进，而传统的工业锅炉控制主要通过人工完成，具有自动化程度低、人力消耗大、控制不稳定的缺点[5]，锅炉温度数字控制逐渐成为主流。经典控制理论适合解决线性定常系统的控制问题，并有较好的成效。但是复杂工况下的工业锅炉属于非线性时变系统，传统的方法控制效果不佳[4]。近十多年来，计算机技术快速发展催生了以程序控制为主的现代控制理论[5]，现代控制理论基于状态量描述，在解决线性或者非线性定常或者时变的多输入多输出系统问题中得到了广泛的应用。然而，传统控制和现代控制都要求建立被控对象的精确数学模型，定量地分析被控对象的各种指标进行系统设计。然而对于锅炉这一个干扰因素较多、变化复杂的系统来说，建立一个精确的数学模型是非常困难的[4]，一个固定的数学模型不可能满足千变万化的生产环境条件。因此，传统或者基础现代控制理论要想在锅炉温度控制中获得较好的控制效果几乎不可能。

本设计以传统 PID 模型为基础，在 MATLAB Simulink 平台上建立了 PID 控制仿真程序，并通过 TCP/IP 连接与运行在同一主机上的 Python TensorFlow LSTM 深度学习程序进行实时通信，PID 从深度学习模块实时地接收预测误差并反馈控制数据，在构建了一个行之有效的通信框架的基础上，通过深度学习预测的方法，让锅炉温度模拟控制系统取得了同工况下的更优良的控制效果。通过仿真验证，证实了深度学习优化可以让 PID 锅炉温度控制取得更好的动态性能[5]，即更小的调节时间 t_s 和超调量 $\sigma\%$ 等。

1.2 国内外相关研究进展

传统的锅炉温度控制主要采用 PID 控制方法，其中数字 PID 控制在生产过程中是一种最普遍采用的控制方法，在机电，冶金、机械等行业得到了广泛的应用[6]，其原理是将偏差的比例（Proportional, P）、积分（Integral, I）和微分（Derivative, D）通过线性组合构成控制量[7]，可以通过调整这三个单元的增益K_p、K_i 和 K_d 来调节 PID 控制量特性，称为 PID 控制器。因此，PID 控制器是线性控制器的一种，它根据给定值 $y_d(t)$ 与实际输出值 $\mathrm{y}(t)$ 构成控制偏差 error，误差反馈给 PID 控制器进行控制调节。

PID 控制要想达到良好的效果，必须对参数进行较好的整定；目前普遍采用的 PID 整定办法包括人工调试、齐格勒-尼科尔斯方法（Ziegler–Nichols method）和 Cohen-Coon 离线自整定办法等[8]。人工调试方法不需要精确的数学计算，可以在线调节控制器参数并观察效果，缺点是其是基于经验的一种整定办法，往往不能达到很好的效果[9]，需要由有经验的工程师完成；齐格勒-尼科尔斯方法是 20 世纪 40 年代早期由泰勒仪器公司的两位工程师提出的，该方法也就由两位工程师的名字命名[4]，其调试办法为，先将积分和微分增益置 0，将比例增益从 0 开始增加直至极限增益 K_u，此时控制器输出将在恒定值振荡。根据 K_u 和振荡周期 T_u 的不同类型，通过下表中的方式来设置比例、积分和微分增益[10]：

Ziegler-Nichols 方法			
控制类型	K_p	K_i	K_d
比例	$K_u/2$	-	-
比例-积分	$K_u/2.2$	$1.2K_p/T_u$	-
经典比例-积分-微分（PID）	$0.66K_u$	$2K_p/T_u$	$K_pT_u/8$
Pessen Integral Rule	$0.7K_u$	$2.5K_p/T_u$	$0.15K_pT_u$

部分过冲量	$0.33K_u$	$2K_p/T_u$	$K_pT_u/3$
无过冲量	$0.2K_u$	$2K_p/T_u$	$K_pT_u/3$

表 1-1 Ziegler-Nichols 方法

齐格勒-尼科尔斯方法具有 $1/4$ 振幅衰减的特性（系统第二次超调过冲量是第一次的 $1/4$），这样的特性在某些应用场合是适宜的，但并非所有的控制场合都符合该方法，齐格勒-尼科尔斯方法是一种基于经验的控制办法。Cohen-Conn 方法具有较好的程序模型，但是它需要一些数学方法，需要离线的调试，只对一阶系统具有较好的效果。

综合上述几种整定办法来看，PID 控制系统能够处理很多控制问题，但是其问题在于 PID 的控制参数是定值，且控制器结构简单，在调试前 PID 并不能知道被控对象的信息，实际工况下，经常改变 PID 控制器参数也非常困难且不切实际，因此 PID 经过参数整定后的整体控制性能是一种妥协的结果[11]。针对锅炉温度控制，目前主要的改进思路有两种，一种是通过模糊控制等手段辅助传统的 PID 控制模型，另一种则是改用新的控制方法结合优化策略来回避传统温度控制器无法较好应对非线性系统控制的问题[12]。

模糊控制理论的思路是让计算机模拟人的决策判断行为，形成一条条模糊控制规则，从而取代人对系统对象进行控制。1974 年英国的马尔丹尼设计出了模糊控制器，从一开始便用于锅炉和蒸汽机的控制并获得了成功。王丽娟在文[4]中设计了一种以单片机为核心的智能化电热锅炉温度控制系统，将模糊控制算法引入了控制策略中，对传统 PID 控制进行优化，经实际运行证明，该控制系统具有良好的控制效果，能够满足实际需要。其优点是模糊控制模拟的是人的控制经验而不依赖被控对象的数学模型，具有很强的鲁棒性，可以实现自适应[13]，然而，模糊控制依然属于总结专家经验来进行控制，模糊规则由大量离线的经验法则即模糊控制语句组成，实际的工况的被控对象往往非常复杂，需要的模糊规则的数量也相应地增多，想要完整地构建一个性能良好的模糊控制器是非常有难度的。

程龙等人[14]针对锅炉温度控制系统中存在比较典型的延时、非线性控制等的问题，提出了基于遗传算法的滑模控制系统（Sliding Mode Control，SMC）以取代传统的控制器对锅炉温度进行控制。滑模控制也称变结构控制（Variable Structure Control，VSC）[15]，它能够根据系统的动态过程调节系统使其达到设定的切换面，并且不需要被控系统结构固定，属于特殊的非线性控制。滑模控制不

涉及被控对象参数，因此设计上较为简单，但是滑模控制系统缺乏有效的、易操作的设计方法，在面临非线性系统设计过程中，最大的难点在于将以微积分为基础的分析方法应用于其算法约束条件下，由于其数据不可微，导致参数难以优化。而遗传算法则为参数寻优提供了更好的途径，避免了上述的缺点。此种方式的局限性在于遗传算法寻找最优值的收敛速度慢、局部搜索能力差、控制变量较多等，使得系统设计在另外的层面上变得复杂。并且，不采用传统的 PID 控制也放弃了 PID 控制系统本身具有的控制方式简单的优点。

李盛伟等人[16]提出了一种基于串级 PID 温度控制技术，使用 PID 控制循环风机风量以控制锅炉出水温度，其主控制器采用 PID 控制，副控制器采用 P 控制，通过级联，配合主副风温变送器，以控制器串联的形式来进行控制，可以稳定地控制锅炉出水温度，并且系统具有一定的抗干扰能力。

1.3 研究内容

本设计在分析了传统控制方式在锅炉温度控制的优势和不足，了解了已有的一些被证实有效的优化方式的基础上，设计并实现了一个基于 LSTM 的锅炉温度预测系统，并通过 TCP/IP 实时地将 Simulink 仿真的结果传入预测程序进行误差预测，将预测结果返回给 Simulink 中的 PID 控制模型。LSTM 非常适合进行时间序列预测，具有极强的系统鲁棒性，能够很好地应对多种工况下需求不同的温度控制问题，温度曲线能够较快地收敛到并维持稳态。设计的内容主要分为以下几个部分：

1. 在 Simulink 平台基于平台提供的模块实现了一个模拟的 PID 锅炉温度控制系统，通过参数整定方法设计了合理的 PID 控制器参数，同时加入了随机扰动以模拟多种工况下的温度控制；
2. 使用 MATLAB FUNCTION 编写了能够在 Simulink 模型内运行的通信功能模块（TCP/IP 通信客户端），将 PID 控制器的数据计算得到误差发送给 LSTM 预测程序，并接收 LSTM 预测程序回传的下一个时间点的误差预测值；
3. 使用 Python 深度学习开源库 Tensorflow 编写了 LSTM 时间序列预测基础模块，根据需要设计了 LSTM 结构；

4. 使用 Python Socket 编程实现了 Python TCP/IP 通信服务器端，响应来自 Simulink 的 PID 控制器输出，将 PID 控制器输出作为 LSTM 预测输入，将 LSTM 预测结果传回 Simulink PID 控制器。

1.4 文章组织结构

本文的文章组织结构依照论述内容，划分如下：

第 1 章为绪论，介绍了锅炉在工业生产中的重要应用，介绍了锅炉温度控制问题的背景和已有的解决方案并进行了分析，指出了各自的优势以及不足之处。最后对设计主要研究内容和文章编排结构进行了介绍。

第 2 章为关键技术背景知识介绍。首先介绍了关于神经网络 LSTM 的工作原理以及 TensorFlow Keras 深度学习模块的基础应用，介绍了研究中使用的仿真平台 Simulink 以及使用到的主要功能模块。其次介绍了 Simulink 仿真平台的一般构建流程，以及 Simulink 提供的 MATLAB 函数接口。

第 3 章为优化控制系统架构。讲解了 PID 控制模块部分和 LSTM 预测模块和相关 TCP/IP 通信模块的设计流程。对功能的可行性进行了分析，主要对 Simulink 平台和 Python TCP/IP 通信相关支持、数据类型和数据转换以及数据集特点做了介绍，得出系统设计思路可行的结论。其次，介绍了传统 PID 的结构，以及在此基础上进行的优化 PID 设计系统的结构。最后，介绍了 Simulink TCP/IP 通信模块的实现。

第 4 章为 LSTM 相关模块的设计与实现。首先对本设计采用的 LSTM 结构做了介绍，并详细介绍了本实验的数据集获取思路。最后介绍了 LSTM 结构和 Python socket TCP/IP 通信服务器端的实现流程。

第 5 章为仿真测试。首先设计了一个仿真实例，其次，对 PID 参数进行了基于经验的单一变量（控制变量）方法整定。最后，进行仿真，定性地分析了优化前后仿真曲线动态性能指标的差异，验证了优化的有效性。

第 6 章：总结与展望。回顾了本设计所作的工作，针对本设计所解决的问题做了系统的概括。重点分析了本设计中存在的不足和问题，并对后续可能的工作方向进行的初步的讨论。

2 关键技术背景知识介绍

在本设计实际进行之前，有必要对项目涉及到的关键技术背景知识做一定介绍。PID 基础知识和相关研究在绪论中已有涉及到，本节主要介绍神经网络和设计中使用的仿真平台。

2.1 神经网络

人工神经网络（Artificial Neural Network, ANN），简称神经网络（Neural Network, NN）或类神经网络，它是模仿生物的中枢神经系统的结构和功能的数学计算模型。生物的神经元胞体、树突和轴突组成，神经网络中的节点模拟了这种连接结构，图 2-1 展示了一个三输入的节点结构：

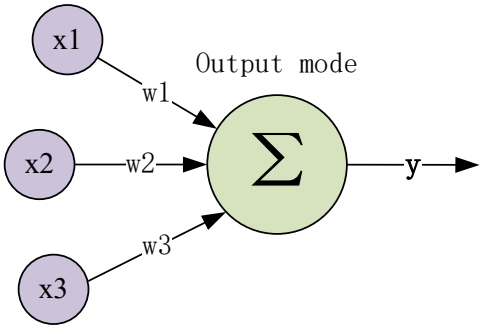

图 2-1 神经网络节点结构

神经网络使用模型，可以对函数进行近似。多个神经元连接形成计算结构，是一种自适应的系统，能够处理多种传统编程所不能解决的问题，如图像识别、语音识别、数据预测等。而神经网络同样适用于本设计中的温度误差预测任务，对 RNN，以及 RNN 的改进版本 LSTM 的基础有一定了解是非常有必要的。

2.1.1 RNN 与 LSTM

相比传统的神经网络，时间循环神经网络（Recurrent Neural Network, RNN）引入了时间循环，这使得 RNN 具有保存信息的能力。图 2-2 展示了 RNN 的基本工作原理，A 是一个神经网络，其接受输入 x_t，得到输出 h_t。A 具有的循环允许信息在神经网络中逐层传递并得到保存。

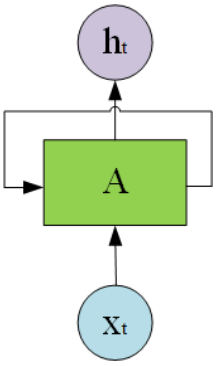

图 2-2 RNN 神经网络

实际上，RNN 可以被视为许多相同的神经网络单元的连接，如果展开 RNN，会得到如图 2-3 的级联结构：

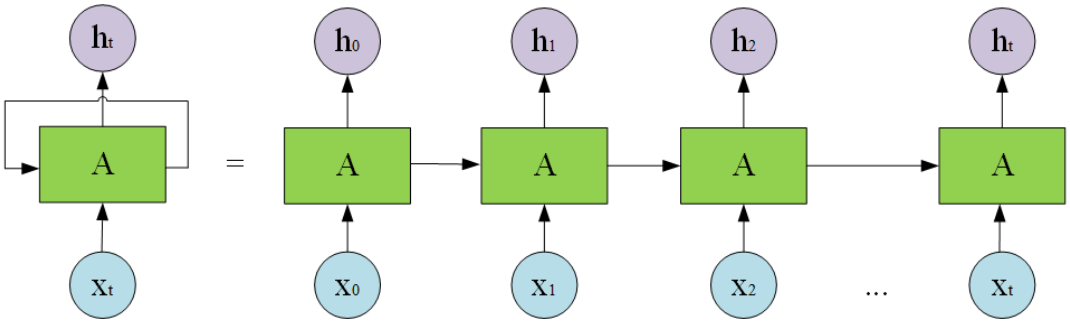

图 2-3 RNN 展开图示

RNN 的结构很自然地能够让人们想到序列或者列表，在过去的几十年 RNN 被用于许多有一定连续性的问题的求解并获得了成功，如语音识别、语言建模、机器翻译和图像标记等。RNN 的优势在于引入了信息保存的机制，但如想要让保存的信息辅助当前的工作需要满足一定的条件。如果当前的预测需要的信息是不久之前的，RNN 会非常有效；但是，如果接下来的预测需要的信息在更长之前出现过，RNN 无法有效处理这种长期的依赖。以语言建模中的词汇预测为例：

Father and mother, I love ___.

如预测 love 之后的一个单词，我们只需要 “Father and mother” 这一条关键信息，就能够很好地预测接下来的一个单词是 “you”，这是非长期依赖的情况；接下来的一个例子展示了一个长期依赖的情况：

I am Chinese, I live in China. … I can speak ___.

如果需要预测 speak 之后的词，关键之处在文章开头部分的自我介绍，但有效信息之间间隔太远，RNN 很难有效地学习长期的信息之间的联系，因此在这

个预测场景下 RNN 不能得到很好的预测结果。在实际应用中，RNN 的显著缺陷之一正是不能处理长期依赖的问题，而 LSTM 则能够应对更为广泛的时序案例。

长短期记忆（Long Short-Term Memory, LSTM）是一种特殊的 RNN，由 Hochreiter 和 Schmidhuber 于 1997 年提出。LSTM 针对 RNN 具有的长期依赖问题进行了特殊设计，弥补了传统 RNN 的缺点[17]。

标准的 RNN 中，重复的神经网络单元 A 具有简单的单层结构，图 2-4 展示了一个基础的 tanh 激活函数的 RNN 网络结构：

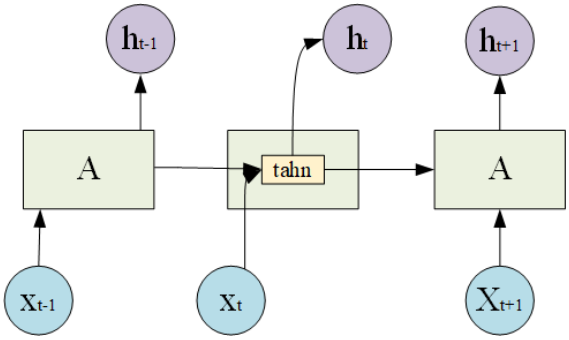

图 2-4 RNN 神经网络单元单层结构

LSTM 的链式结构单元则具有不同的结构，神经元内拥有 4 层交互结构。LSTM 单元具有单元状态（cell state），通过门（gate）来向单元状态增添或删除信息。遗忘门（forget gate）用于删除信息，输入门（input gate）用于添加信息，输出门（output gate）用于输出信息。LSTM 的计算过程如图 2-5 所示：

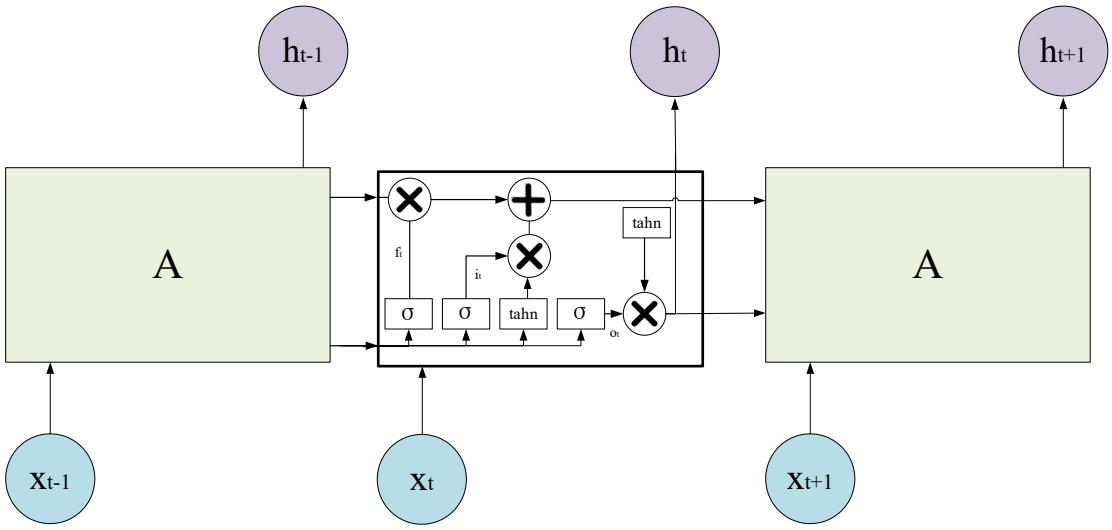

图 2-5 LSTM 计算过程

第一步，LSTM 遗忘门会根据 h_{t-1} 和 x_t 计算得到状态 C_{t-1} 中的每个数。1 代表为完全保留，0 代表完全抛弃。

$$f_t = \sigma\left(W_f \cdot [h_{t-1}, x_t] + b_f\right) \quad (2-1)$$

接下来需要决定需要在单元状态中保存的信息。这主要分为两个步骤：首先，输入门会决定哪些值需要更新，计算出 t 时刻输入门的状态 i_t，之后，tanh 会生成一个包含新的数值的向量 $\tilde{C}_t$ 加入到状态中。

$$i_t = \sigma(W_i \cdot [h_{t-1}, x_t] + b_i) \quad (2-2)$$

$$\widetilde{C_t} = \tanh(W_c \cdot [h_{t-1}, x_t] + b_C) \quad (2-3)$$

之后，新状态 C_t 将会取代旧的状态 C_{t-1}。f_t 表示遗忘门，i_t 代表输入门，C_t 与 C_{t-1} 之间的关系为：

$$C_t = f_t \star C_{t-1} + i_t \star \tilde{C}_t \quad (2-4)$$

最后，需要决定 LSTM 的输出。输出以单元状态为基础。首先会计算 t 时刻输出门的状态 o_t 决定哪些部分需要输出，接下来经过 tanh 层将单元状态 C_t 整定到-1 与 1 之间，最后得到的隐藏状态（输出）h_t 可由式 2-5 得到：

$$h_t = o_t \star \tanh(C_t) \quad (2-5)$$

2.1.2 TensorFlow Keras

本小节介绍在设计中用于实现基于 LSTM 的深度学习神经网络模型的 Python 库 TensorFlow 以及其下的深度学习模块 Keras。TensorFlow 最初是由谷歌大脑团队开发维护的机器学习开源软件仓库，于 2015 年 11 月 9 日在 Apache 2.0 许可证下发布。而 Keras 则是用于构建和训练深度学习模型的 TensorFlow 的高阶 API。利用此 API 可以快速地实现原型设计、先进的研究和生产。Keras 具有三大优势：

- 方便用户使用

  Keras 具有针对常见用例做出优化的简单而一致的界面，它可以针对用户错误提供切实可行的清晰反馈。

- 模块化和可组合

  将可配置的构造快组合在一起就可以构建 Keras 模型，并且几乎不受限制。

- 易于拓展

可以编写自定义构造块；还可以创建新的层、标以及损失函数并开发先进的模型。

Keras 主要可以划分为以下几个部分：

① Engine

Engine 包括了模型的定义并负责模型的执行。Engine 包括基础 Layer 的定义并涵盖了 Layer 的 DAG（Directed Acyclic Graph，有向无环图）结构，即基础网络。其次，在 traning.py 包括了模型 Model 以及对应的训练和估值循环。其还提供了 Sequential 顺序模型，本质上，顺序模型是多个 Layer 的线性堆叠。

② Layers

该部分包括了 Layer 的许多可用的子类。

③ Losses, Metrics

包含了基础的损失函数和评价函数及对应子类。

④ Callbacks

包含了基础回调及对应子类。

⑤ Optimizer

包含基础优化函数及对应子类。

⑥ Regularizers, Constraints

包含了输出的正则化函数以及权值约束函数等。

Layer 是 Keras 中的重要抽象之一。Keras 中的所有内容皆基于 Layer，或者与 Layer 有紧密的交互。Layer 的功能是接受一批输入，计算得到一批输出。Layer 可以在 TensorFlow 提供的命令式编程环境下工作（Eager Execution），也支持图形化的执行（Graphic Execution）。同时，Layer 提供了遮盖功能（masking），这在时间序列和特征缺失的情况下很有用。

本设计中使用的是顺序模型。Keras Sequential 模型的使用步骤包括输入数据的指定、模型编译以及模型的训练。

Sequential 构造器可以接收 Dense 全连接层并为每层指定不同的激活函数。为了得知输入数据的尺寸，顺序模型中的第一层（且只需要第一层，后续层可以

自动推断尺寸）需要接收关于输入尺寸的信息，可以通过 input_shape 参数传递表示尺寸的元组做到。

模型编译通过 compile 方法完成。compile 接收三个参数：

• 优化器 optimizer。优化器可以是现有的优化器的字符串标识符，也可以是 Optimizer 类的一个实例。

• 损失函数 loss。损失函数可以评价一个模型的好坏，一个好的模型需要具有尽可能小的 loss 值。loss 值可以是现有的损失函数字符串标识符，也可以由用户自行设计损失目标函数并传入。

• 评估标准 metrics。评估标准可以是现有的标准的字符串标识符，也可以是自定义的评估标准函数。

模型训练常用到 fit 函数。fit 函数常用的参数为：

• x：训练用到的 Numpy 数组。

• y：标签 Numpy 数组。

•batch_size：可以是整数或者 None。为每次梯度更新的样本数，默认为 32。

• epochs：整数，表示训练模型的迭代次数。

• verbose：可以为 0 或者 1 或者 2，表征模型训练日志显示的模式。0 = 安静模式，1 = 进度条，2 = 每轮一行。

模型训练好后，可以调用对象方法 save 进行保存，并通过 keras.models 提供的 load_model 方法在之后的预测任务中加载已经训练好的模型。至此完成了模型训练的过程并得到了可重用的模型。一般而言，模型的好坏取决于训练所用的数据集的质量，包括数据的准确度以及数据的量。

2.2 MATLAB 与 Simulink

MATLAB 是由 Mathworks 公司推出的面向工程师和科学家的编程和数值计算平台，主要用于数据分析、算法开发以及模型构建。Simulink 是一个模块图环境，用于多域仿真以及基于模型的设计。它支持系统级设计、仿真、自动代码生成以及嵌入式系统的连续测试和验证。Simulink 提供图形编辑器、可自定义的模块库以及求解器，能够进行动态系统建模和仿真。Simulink 与 MATLAB 相集成，

因此用户不仅能够在 Simulink 中将 MATLAB 算法融入模型，还能将仿真结果导出至 MATLAB 做进一步分析。

2.2.1 MATLAB 语言

MATLAB 是 Matrix Laboratory（矩阵实验室）的缩写。MATLAB 语言是以线性代数软件包 LINPACK 和特征值计算软件包 EISPACK 中的子程序为基础发展起来的一种开放式程序设计语言，是一种高性能的工程计算语言，其基本的数据单位是没有维数限制的矩阵。[18] MATLAB 提供了一个实时 MATLAB 语言解释环境，可以实时地得到输入的命令的结果。MATLAB 同时支持将命令保存为脚本文件，可以满足代码的复用。

本设计主要涉及 MATLAB 函数。函数是 MATLAB 语言提供的高级特性之一，与其他高级语言如 C++、Python 类似，MATLAB 函数提供了显式的参数和返回值声明，MATLAB 函数的基本形式如下：

function [返回值列表] = {函数名}(参数列表)

function 关键字需要有与之匹配的 end 关键字以划定函数体范围。在函数体内部，如有必要，需要为返回值进行赋值（如无返回值则可省略不写）。MATLAB 可以通过 reporter 来查看代码中潜在的错误，但是其提示较为有限。同时，MATLAB 语言是一种非自由、不开源的设计语言，这对工程师理解 MATLAB 语言内部逻辑以检查错误和自定义也造成了一定的困难，但 MATLAB 已经提供了大量封装好且性能良好的 API 供直接使用。MATLAB 适合用于敏捷开发一些重要的系统功能。

2.2.2 Simulink 平台

Simulink 提供了一个直观的系统构建界面，用户可以通过拖放多种功能包提供的模块经过连线形成完整的系统，并在 Simulink 中实时仿真。Mathworks 为 Simulink 提供了包括蓝牙、自动驾驶、硬件开发和深度学习在内的许多工具箱，涵盖了常见的工程开发中需要用到的功能模块，非常适合搭建模拟仿真系统。

Simulink 随 MATLAB 一同发布，并可与 MATLAB 配合使用。Simulink 不同版本间向下兼容，保证了设计的连贯性。在 Simulink 创建空白模型后软件界面如图 2-6 所示。

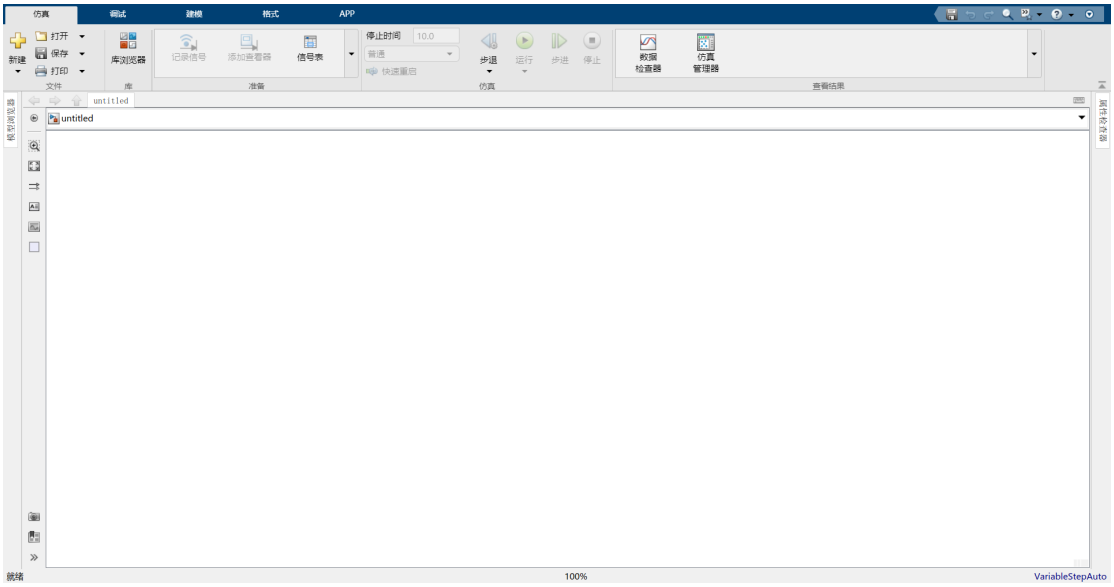

图 2-6 Simulink 空白模型界面

通过库浏览器可以浏览目前拥有的功能模块，同时，可以通过 MATLAB 平台提供的附加功能管理器在线地从 Mathworks 服务器获取新的工具箱。Mathworks 的工具箱已经涵盖了大多数功能，也可以选择下载由 Mathworks 社区用户打包上传的工具箱并使用。有经验的用户也可以通过 MATLAB Toolstrip 制作自己的 MATLAB App。

2.2.3 在 Simulink 中使用 MATLAB 语言

Simulink 提供了诸多的功能模块，提供了可视化的仿真环境搭建流程，但如用户希望自己控制仿真的逻辑，Simulink 也允许用户使用由 MATLAB 编写的功能模块并使用。

Simulink 支持 MATLAB 语言主要通过 S 函数和 MATLAB Function Block 组件。其中 S 函数是 Simulink 提供的高级特性，分为不同的等级，支持使用 C++、Python 和 MATLAB 编写 S 函数。Simulink 会自动地识别 S 函数的参数和返回值，因此不同等级的 S 函数均需要遵循一定的规范，函数初始化、返回值的设定都有严格的规定。

此外，Simulink 提供了 MATLAB Function 功能模块，允许用户编写 MATLAB 函数并在 Simulink 中使用，限制较 S 函数要少一些。MATLAB Function 会在仿真时执行，并生成 Simulink Coder 对象代码。通常，Simulink Coder 会生成可执行的 C/C++模型代码，一些 MATLAB 函数可能无法生成对应代码，可以在 MATLAB Function 中通过外部函数声明，指定一部分函数为外部函数，经由 MATLAB Engine 执行函数调用而不生成对象代码。

外部函数声明的语法结构是：

coder.extrinsic(function1, ... ,functionN)

外部函数同时具有一些限制。外部函数调用需要制作传入的数据的副本和输出数据的副本并将数据传回 MEX 函数环境，这会带来一定的性能开销；同时，代码生成器不支持使用 coder.extrinsic 来调用位于私有文件夹下的函数，也不支持使用 coder.extrinsic 调用本地函数。

本设计需要借助 MATLAB 语言实现 TCP/IP 的客户端通信流程，综合考虑以上两种方式，选用 MATLAB Function 作为 Simulink 与 MATLAB 代码之间的桥梁，在 Simulink 中使用了 MATLAB 函数。经过实际测试，这种方式能够满足设计的性能需求，并具有较好的控制精细度。

2.3 本章小结

本章首先介绍了 RNN 和 LSTM 的概念，分析了 RNN 的优势和不足，针对 RNN 不能解决的问题引入了 LSTM，对 LSTM 的输出计算过程进行了分析。之后，介绍了设计中使用的 MATLAB 和 Simulink 平台，阐述了它们在工程设计上的显著优势。最后，分析了两种在 Simulink 平台下使用 MATLAB 进行模型构建的方式，分析了各自的优劣，给出了本设计选择 MATLAB Function 作为实现手段的理由。

3 系统架构与优化控制系统实现

本设计系统采用 MATLAB Simulink 作为温度控制系统仿真工具，使用 TensorFlow 搭建基于 LSTM 的控制预测模块。系统的重要功能是实现两个模块间的数据通信。优化 PID 控制系统的具体设计包含 PID 控制系统优化结构与 Simulink 平台 TCP/IP 数据通信两个部分。

3.1 系统需求分析

本设计的目标是设计一个深度学习预测模块与 Simulink 温度控制仿真模块协同工作的系统，验证 LSTM 预测对控制结果有优化作用。

LSTM 预测模块会接受控制系统计算的原始误差，并根据前 50 个时间点的误差对下一个时间点的误差进行预测，输出预测误差值。Simulink 控制系统会接收这个误差值，输出控制结果，并反馈结果用于原始误差计算。LSTM 对长期数据具有很好的记忆和处理能力，因此，使用前 50 个时间点而不是前 1 个时间点的数据用于预测能够让 LSTM 的预测表现更好。因为系统采样时间定为 0.01s，因此从 LSTM 接受误差进行预测到 PID 控制器收到 LSTM 预测误差结果的时间消耗不能大于 0.01s，否则最后得到的优化控制曲线将丧失实时性。、

3.2 模块实时通信框架

在整个设计环节中，有必要考虑如何实现神经网络预测模块与仿真平台之间的通信。Simulink 仿真平台十分方便，提供了多种工具箱，能满足各种工程系统的构建需求。但 Simulink 自身提供的神经网络工具箱功能较少，能力有限，无法满足本设计中 LSTM 预测的需求。本设计将 LSTM 部分与 Simulink 控制系统分割开来，实现了一个实时通信框架，解决了设计中两个不同程序实体之间问题，使得 LSTM 能够在线地进行预测并输出预测结果。

实践证明，TCP/IP 通信是有效的方式。具有 Python TCP/IP 通信模块的 PID 控制系统结构如图 3-1 所示：

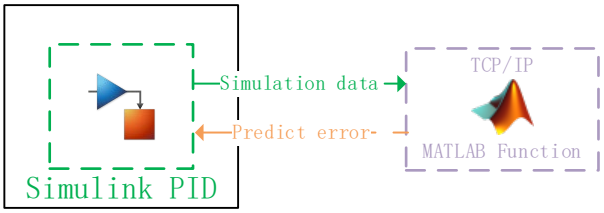

图 3-1 具有 Python TCP/IP 通信模块的 PID 控制系统结构

PID 负责对被控对象进行具体的控制，给出控制结果（当前温度）并反馈到 LSTM 联络模块中；LSTM 预测模块获取到 PID 的控制结果后，根据既有温度控制结果记录，给出下一个误差输入预测值，传给 PID 控制器。相较于传统单 PID 控制结构，本设计仅在 PID 控制器前加入误差预测调节模块，保留了 PID 控制系统结构简单的优点，便于后续改进。

3.3 可行性分析

3.3.1 TCP/IP 连接建立

首先需要探讨的是设计中两个不同功能体之间的通信问题。考察 Python 和 MATLAB 的库函数，可以知道它们都有涉及到 TCP/IP 通信的部分。Python 提供了完善的 socket 通信库，可以简单地建立可靠、即时的 TCP/IP 连接，这对本设计非常有益；MATLAB 提供了 tcpclient 封装函数，可以创建 tcpclient 对象，执行一些 TCP/IP 通信操作。tcpclient 可以代表 MATLAB 程序与远程主机的一个 TCP/IP 连接，远程主机可以是网络服务器，也可以是支持 TCP/IP 的其他硬件，但它必须已经存在。tcpclient 支持向缓冲区读、写数据。

此外，还需要注意的是 tcpclient 支持几种函数重载，t = tcpclient(address,port) 创建了一个 TCP 客户端，它与地址为 address 的服务器建立了连接，并绑定了端口号为 port 的端口。如果制定了无效的地址或者端口，或者 TCP/IP 服务器没有运行，将不能建立连接，tcpclient 对象无法建立，MATLAB 将抛出异常。

第二种重载，t = tcpclient(address,port,Name,Value)，它会建立一个和之前相同的连接，但可以附加其他的一些属性，以 <Name, Value> 的形式给出。tcpclient 支持的构造参数包括：

Address，表征远程主机名或 IP 地址，可以是字符向量或者字符串标量；

Port，远程主机端口，必须是 1~65535 之间（包含 1 和 65535）的数值；

Timeout，允许一次操作的时长，必须为数值，表征一次读或写允许的最长时间，以秒为单位。同时，Timeout 也是 tcpclient 对象的可访问属性，支持根据实际需要，动态修改；

ConnectTimeout，允许建立连接的时长，必须是数值，表征从发起连接到收连接成功或失败允许的最长时间，以秒为单位。

EnableTransferDelay，是否允许来自服务器的延迟确认。值可以是 true 或者 false，表征 Nagle 算法的启用或禁用。当属性设置为真，则客户端会收集小段的未完成的数据片，并在来自服务器的确认 (ACK) 到达时将它们以单个小数据包的形式发送。 如果用户希望立即发送数据到网络，属性应当设置为 false。 如果网络速度较慢，可以通过启用传输延迟来提高其性能。 但是，如果网络连接情况较好，确认会很快到达，启用或禁用传输延迟之间的差异可以忽略不计。

本设计中，两个功能实体在本机运行，因此只需要本机的 IP 地址和端口号就能建立通信链路。实践证明，tcpilent 使用第二种重载不是必要的，这是因为 PID 控制结构中存在数据绑定关系——只有收到了来自 Python LSTM 预测模块的误差，PID 被控对象才能给出控制结果；有了控制结果，才能计算得到误差，之后反馈给 Python LSTM 预测模块进行优化误差预测。任何一个环节如出现延迟，整个系统都将无法运行。这种绑定关系能够让 TCP/IP 连接变得简单，不需要在 TCP/IP 连接考虑某个单一实体因为异常而无响应的状况。

3.3.2 数据类型与数据转换

MATLAB 与 Python 的数据类型定义部分相似，但依然存在类型的映射关系，在通信过程中需要予以关注。因此，事先了解关于数据类型的知识是必要的，特别是 MATLAB 的数据类型相关知识。

MATLAB 的主要数据类型包括数字、字符串、向量、矩阵、单元数据和结构数据。矩阵是 MATLAB 中最基本的数据类型，例如数字可以看成许多数字组成的矩阵。字符串可以看作是字符组成的矩阵。

MATLAB 不要求在使用变量之前显示地做出声明，也不需要指定数据类型。它能够根据用户赋给变量的值或者对变量的操作分析推导变量的类型。且 MATLAB 针对变量命名有如下要求：

- 变量必须以字母开头，之后可接字母、数字或下划线；
- 变量大小写敏感；
- 变量名的长度不得超过 31 个字符，第 31 个字符之后的字符会被忽略。

MATLAB 中的变量如无特别说明，皆为局部变量，作用域最多只限于本文件（.m）中；如果需要声明全局变量，需要在变量名前增加 global 关键字。

同时，MATLAB 预定义了一些变量，习惯上称它们为常量。常见的一些常量包括 ans（默认变量名）、pi（圆周率）、eps（浮点运算相对精度）、inf（无穷大）和 NaN（非数值，或不定值）等。

在本设计中，需要考虑 MATLAB 的数据类型为 double 类型，它占 8 个字节，与 Python 中的 double 双精度浮点类型是对应的。可以通过一次读入缓冲区内的 8 个字节，获取 PID 传来的控制结果；Python 可以遵循 MATLAB 的数据要求，将预测结果（double）封装为包含 8 个 bytes 的数组，传回 Simulink。Simulink 中的 MATLAB 函数模块可以通过 read、write 函数对 TCP/IP 连接缓冲区进行读写。

3.3.3 LSTM 数据集

本设计中的 LSTM 数据集并不容易获得：Simulink 最开始只有一个 PID 控制器，一个被控对象（锅炉温度系统）。如何获取 LSTM 需要的数据集成为了难题，但有一个大致的探寻方向是：LSTM 需要对误差进行某种处理，输出的处理后的误差再传入 PID 控制器，使得 PID 控制器的控制效果能够更好，基于此思路寻找基础的数据集获得了成功，相关流程会在后续实现部分介绍。

3.4 PID 优化控制系统结构

传统 PID 控制结构如图 3-2 所示：

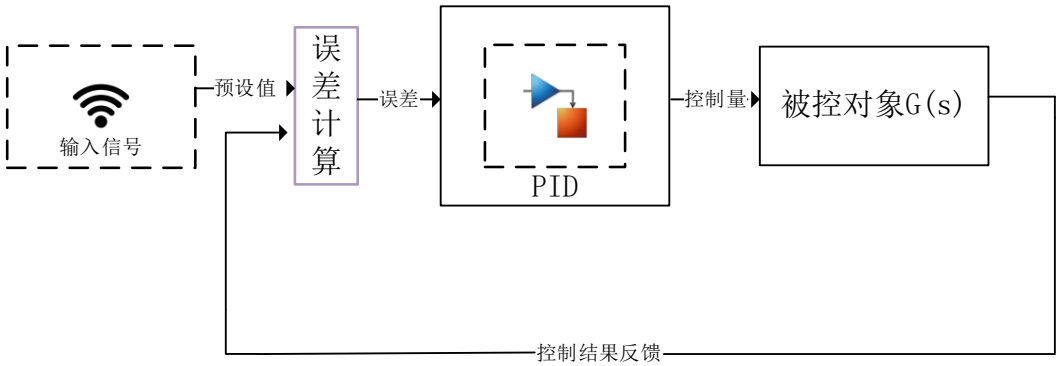

图 3-2 传统 PID 控制结构

基础的 PID 控制系统由输入信号、误差计算模块、PID 控制结构、被控对象（传递函数）4 个部分组成，如图 3-3 所示。

输入信号为常量信号，初始值 0，终止值 5。其表征调节预期温度。

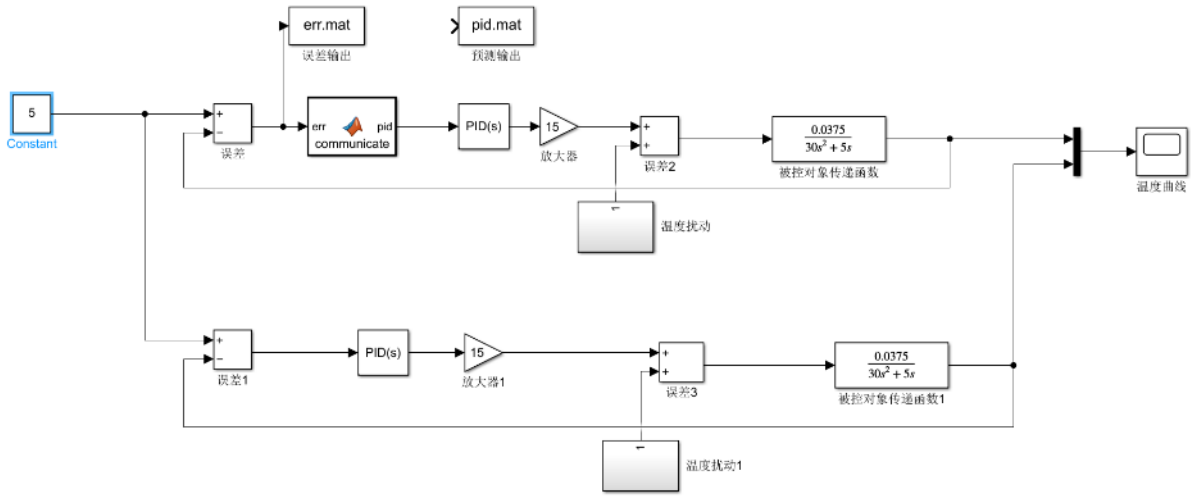

图 3-3 基础 PID 结构

在既有结构的基础上，为了符合设计需求，增改基础结构如图 3-3 所示。

增加放大器。放大器具有信号增益作用，遵循：

$$y_{out} = K \cdot U \qquad (4-1)$$

其中 U 是输入信号值，K 是信号增益，用于放大信号不稳定和超调，贴合实际复杂系统中调节扰动大、收敛慢的工况。

为了增加系统的不稳定和不确定性，引入脉冲温度扰动：振幅-1200（负号表示与原输入相减得到结果），周期 50 秒，脉冲宽度占周期 10%，相位延迟 0 秒，这意味着以 50 秒为一个周期，在起始的 5 秒内系统将受到扰动信号影响。如图 3-4 所示。

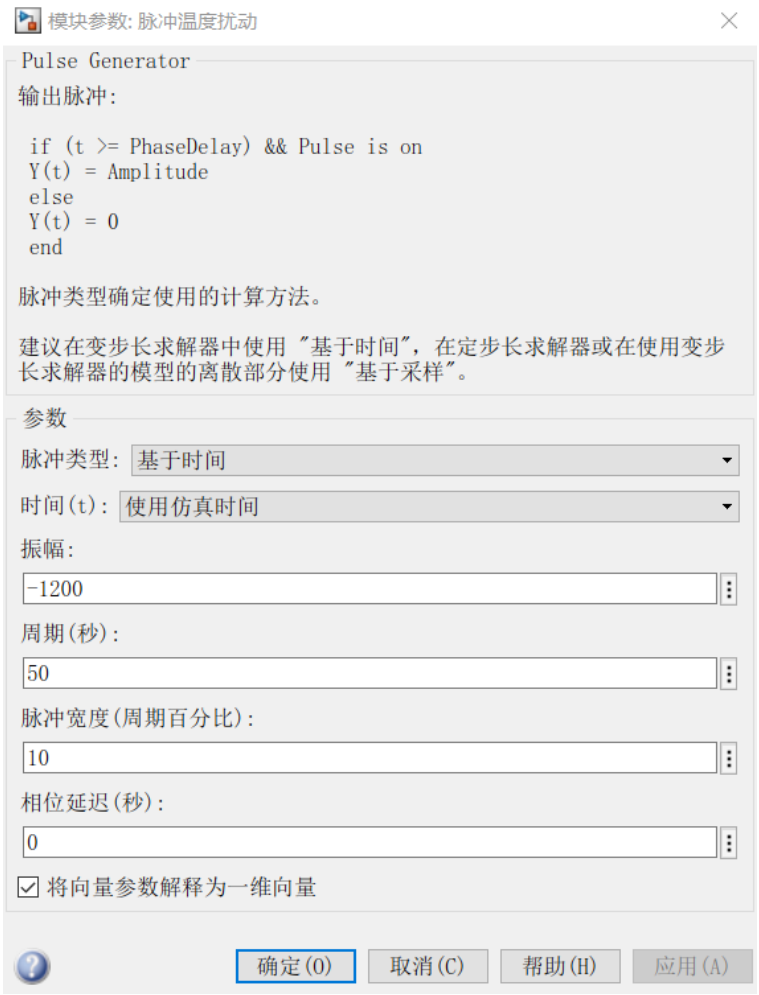

图 3-4 脉冲温度扰动设置

PID 的输出是控制量，控制量输入锅炉温度传递函数，得到锅炉温度输出（当前温度）。

为了进行对比，仿照原始结构，加入引入了 LSTM 预测优化后的 PID 控制链，将两路输出信号通过多路复用器 MUX 连接并接入示波器，仿真后观察波形。

3.5 简单工况下传递函数的推导

设置一个合理的传递函数对仿真至关重要。一个真正的锅炉温度控制系统很难通过数学推导得到精确的传递函数，或者传递函数的形式非常复杂，对于复杂系统，大多数时候都需要分块建模，最后整合[19]。但考虑较为简单的、干扰较少情况对后续的工作依然有理论上的意义。本设计采用的传递函数为常见的二阶传递函数，本节希望通过考虑一个简单的实际场景，推导出基础的传递函数的形式，由此说明将锅炉温度控制系统传递函数设置为二阶的合理性。

考虑电热锅炉，传统工业电热锅炉通过热传导效应工作，电阻丝（加热装置）升温后，将热能传递给高压水蒸气，从而得到携带了热能的高温水蒸汽或高温水。假设加热使用的电阻丝两端施加电压为 U_0（单位为伏特，记为 V），热传导系数（热传导系数的物理定义为两侧物质的表面温度差为 $1K$ 时在 $1s$ 内通过$1m^2$ 传导的热量，单位记为 $J\cdot m^{2^{-1}}\cdot K^{-1}\cdot s^{-1}$）为 H，且传导散热功率恒定，散热面积为S。假设锅炉内有质量为 M 的水蒸气，水蒸气初始温度为 T_1，加热后温度为 T_2，水的比热容为 C。不考虑电阻丝比热容和电阻丝升温、热传导，过程中，电阻丝产热为 Q。研究控制量（电压）与被控量（温度变化 ΔT）之间的关系。

t 时间内水蒸气吸热，温度从 T_1 升高到 T_2，吸收的热量：

$$Q_t = CM(T_2 - T_1) \tag{4-2}$$

考虑较为简单的干扰即锅炉散热，同时假设电阻丝产热一部分被用于加热水蒸气，余下部分全部被耗散掉，散热量 Q_a 与炉内外温度、散热面积和热传导系数相关。假设炉外温度为 T_3，单位时间内散热 Q_a 由以下公式给出：

$$Q_a = HS[(T_2 - T_3) - (T_1 - T_3)] = HS(T_2 - T_1) \tag{4-3}$$

t 时间内电阻丝产热为：

$$\Delta Q = Q_t + Q_a t \tag{4-4}$$

单位时间内，电阻丝产热：

$$q = \frac{U_0^2}{R} \tag{4-5}$$

当达到热平衡时，产热和散热平衡，在热平衡点 (U_o, q_o) 对式 $(4-5)$ 求导，得到：

$$q' = \frac{2U_o}{R}du \tag{4-6}$$

式 $(4-6)$ 是一种线性的近似，可以知道，对任意的电压和对应的单位时间产热，均满足这个产热计算公式。由此可以得到电阻在电压为 U 时，增加电压 ΔU 时电阻单位时间产热变化量：

$$\Delta q = \frac{2U}{R}\Delta U = KU\Delta U \tag{4-7}$$

根据式 $(4-4)$、式 $(4-7)$ 列写热平衡方程，即单位时间内电阻丝产热等于水吸收的热量加上散热量：

$$MC\frac{d(T2-T1)}{dt}+HS(T2-T1)=KU\Delta U \qquad (4-8)$$

令 $\Delta T=T_2-T_1$，$K'=\frac{MC}{HS}$，根据拉普拉斯变换，可以得到系统传递函数为：

$$G(S)=\frac{\Delta T(S)}{\Delta U(S)}=\frac{K}{TS+1} \qquad (4-9)$$

式 $(4-9)$ 表明，在散热恒定且只存在散热因素的情况下，电热锅炉的传递函数是一阶的。为了增加系统的复杂度，本设计将传递函数设置为了二阶，这是合理、可行的。

3.6 PID TCP/IP 通信模块实现

为了实现数据收发在字节粒度上的控制，尝试引入 MATLAB 编程语言是有必要的。Simulink 支持通过 MATLAB Function 模块引入 MATLAB 的部分 API。而 tcpclient 相关的 API 需要外部函数的支持，会有一定的性能开销。此外，MATLAB Function 将常见基本类型外的对象的类型都处理成 mxArray 即数组类型，访问对象的属性存在限制。这也意味着，无法通过 tcpclient 对象访问到缓冲区中的字节数，不能很好地进行同步控制。对此，本设计采取实际验证的方式，为了最大限度地保证速度，不引入除 MATLAB FUNCTION 之外的其他耗时较大模块，使用收、发不间断轮询的方式，经过验证，可以满足同步性要求。代码结构如图 3-5 所示

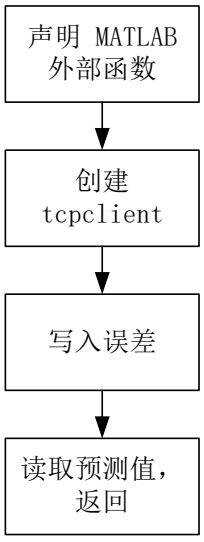

图 3-5 Simulink TCP/IP 代码流程图

以上代码段创建了 TCP 客户端，与 IP 地址为 127.0.0.1 的主机（本机）建立 TCP/IP 连接，绑定端口号为 65432。接着，向缓冲区写入 double 类型的误差数据，写入后尝试从缓冲区读取 1 个 double 数据，即 Python TCP/IP 通信服务器回传的 LSTM 预测结果。最后，按照 MATLAB Function 的要求，显示地声明函数返回值 pid 为单维度实数。至此完成了 TCP/IP 通信 PID 控制系统部分的工作。输入将从温度控制系统的输出获取，设定温度和当前温度相减得到控制误差 err，通信模块将收到的误差预测结果传给 PID 控制器。

当函数模块的输入端口检测到输入值时，会调用 MATLAB Function 函数。在函数中反复创建数据对象也会产生一些资源开销。

3.7 本章小结

本章首先分析了 PID 控制系统的结构，对优化控制系统各个组成部分做了总体介绍。分析了主要功能点的可行性；其次，深入介绍了 PID 优化控制系统结构，推导了简单工况下的传递函数，证实了传递函数选取的合理性；最后，介绍了 PID 部分的 TCP/IP 通信模块的实现。

4 LSTM 相关模块的设计与实现

本节介绍 LSTM 相关模块的设计与实现，对 LSTM 预测模块结构和对应的数据集获取做了较为细致的分析。

4.1 LSTM 预测模块结构

本设计中，LSTM 预测部分采用 TensorFlow Keras 实现，具有 1 层 LSTM 隐藏层用于预测，1 层全连接层用于输出。本设计的 LSTM 长短期记忆神经网络结构图如下图 4-1 所示：

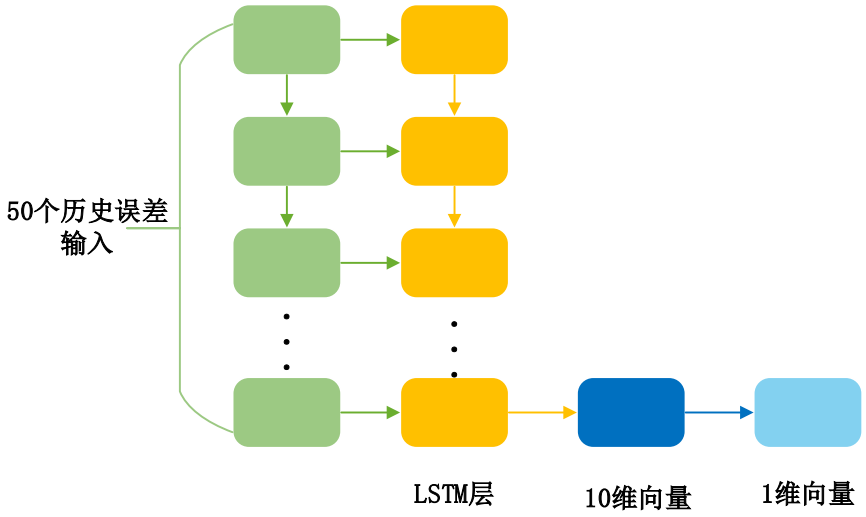

图 4-1 LSTM 结构图

设计采用 50 个历史温度数据作为一组输入，即 LSTM 内有 50 个 LSTM 单元级联，目标是预测得到下一个时间点合适的输入误差。因此全连接层只需要最后一个 LSTM 单元的信息，将设定的 10 维向量降低为 1 维，得到预测输出。

LSTM 层采用的激活函数为 ReLU。激活函数的意义是为线性的回归模型提升非线性部分，增强模型的准确度和表达能力。ReLU 函数全称为 Rectified Linear Unit，即规整线性单元。它的表达式是：

$$ReLU(x) = \begin{cases} x, & x \geq 0 \\ 0, & x < 0 \end{cases} \qquad (4-1)$$

ReLU 激活函数在 $x < 0$ 时梯度为 0，如果 ReLU 函数遇到负数输入，之后神经元的梯度可能一直保持为 0，为避免这种情况，使用 ReLU 激活函数时需要

注意对学习率进行合理的设置，防止梯度停留在负数区域，出现神经元坏死的情况；但相对的，输入 $x > 0$ 时， ReLU 函数具有良好的梯度。而 sigmod 和 tanh 激活函数在 [-1,1] 梯度较好，在其他区域函数趋于平坦，为了防止隐藏层的输出趋于相同，有必要进行归一化防止数据饱和，同时防止训练梯度出现问题。而 ReLU 函数则不必进行归一化。

本训练集中特征维度小，易于处理，采用 ReLU 激活函数，没有进行归一化，也取得了较好的效果。

4.2 LSTM 数据集相关探究

LSTM 训练采用监督学习的办法，需要获取完备的数据集。与传统的监督学习任务如动物图像识别不同，动物图像识别任务的数据集可以由工作人员搜集动物图片作为特征，并人为地打好标签，即形成图像识别地数据集。而本研究中，获取数据集需要更进一步的思考。

4.2.1 传统 PID 的不足与改进对策

评价控制系统的性能可以审查系统的动态性能指标，即上升时间 tr，超调量 $\sigma\%$、峰值时间 t_p 以及调节时间 t_s。一个好的系统应当具备较小的上升时间，较小的超调量和峰值时间，同时要具备更小的调节时间。控制参数带来的动态性能指标的差异，可以直观地反映在最终的控制结果的图像上。从控制结果的图像的角度来考虑，振荡越少、超调越小的图像被认为是越好的。图 4-2 展示了 PID 控制曲线 1，图 4-3 展示了 PID 控制曲线 2。可以从图上直观地看到，曲线 2 的振荡时间比曲线 1 要长，且仿真结束后并未收敛。可以认为曲线 1 更符合实际控制的标准。

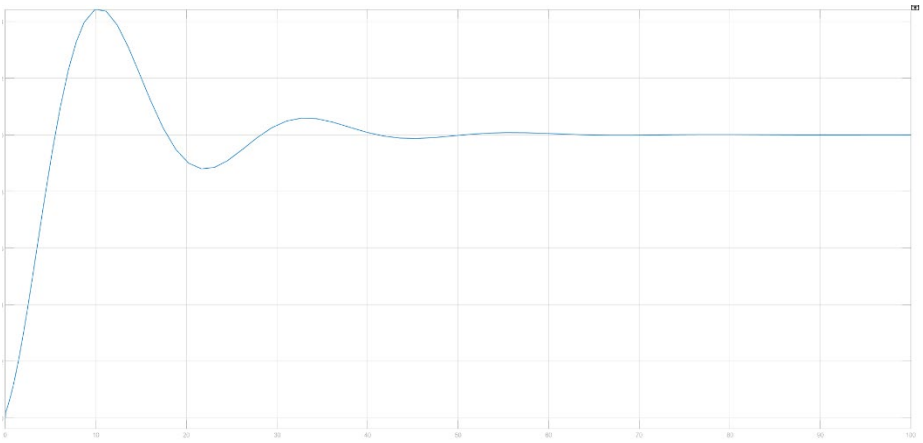

图 4-2 PID 控制曲线 1

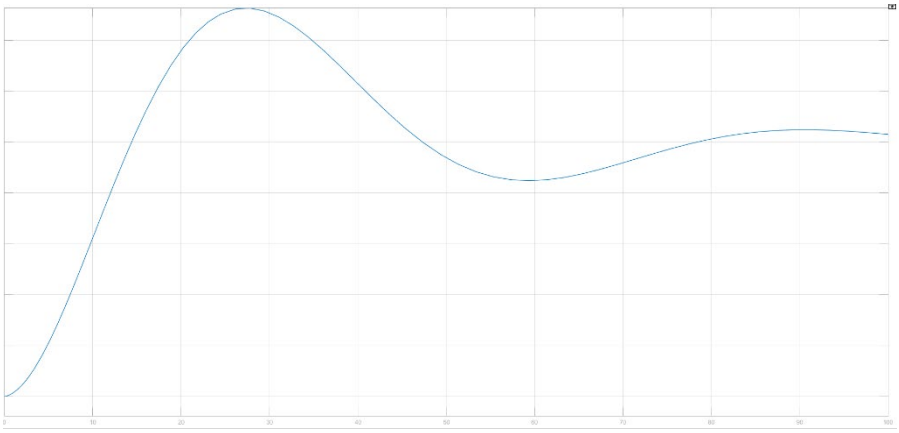

图 4-3 PID 控制曲线 2

影响 PID 控制器的控制性能的三个参数为 P 比例、I 积分、D 微分，三个参数的配置与被控对象传递函数紧密联系。针对不同的被控对象，需要对 PID 控制器赋予不同的参数，才能取得相应的相对较好的效果，而实际工况的被控对象往往很复杂，这为 P、I、D 三个参数的整定带来了挑战。同时，对于一个非常复杂的系统， PID 的控制效果依然存在极限，这个极限在最坏情况下，可能仍然无法令人满意——原因可以预想到，因为一旦 PID 参数确定、被控对象传递函数确定（对于实际工况，传递函数可以认为是确定的，并且通常是复杂的），整个控制系统随即确定，控制系统的输出也随即确定，它出由 PID 参数和被控对象共同决定。在 PID 参数不能轻易调节的情况下，可以预见调节的瓶颈。

为了摆脱这种可以预见的束缚，同时尽可能保留 PID 控制器本身结构简单，方便操作的优点，前文谈及了一些研究者已经实现了的模糊 PID 控制器的办法，

做法是根据实际场景，利用预设的模糊规则动态地调节 PID 参数，因此，PID 参数固定导致输出相对固定而难以优化、且 PID 参数不方便整定的瓶颈被打破了。但是，在本设计设想的 PID 控制器参数不可调节的情况下（比如，PID 已经投入了实际的工业锅炉温度控制中，对它的参数调节已经变得困难），上述做法也将失去优势。

本设计考虑对输入的原始误差进行操作，从而改变 PID 控制器的输出，修正控制结果，突破 PID 参数和控制函数对误差的限制，让原始的误差先经过一个和控制系统无关联的 LSTM 模块进行处理，预测一个更合理的误差，输入 PID 控制器，从而得到更优良的结果。

4.2.2 数据集获取

PID 控制器和被控对象可以看成是一个事实上的系统函数 $S(x)$，它接受误差，得到控制结果，如图 4-4 所示：

图 4-4 系统函数$S(x)$

在这种情况下，$S(x)$ 本身的性质就决定了误差输入，也就决定了控制结果，因此，$S(x)$ 的输入是不自由的。但只要能设计合理的误差输入序列，摆脱 $S(x)$ 自身性质的束缚，配合仿真时间变量，经过 PID 控制公式计算，再输入传递函数系统，得到“更好”的输出是完全可能的。为此，引入函数$f(err)$，它处理原始误差 err，得到 PID 控制器输入 x（预测的优化误差）。系统$S(x)$ 以x 和时间 t 作为变量得到输出 y_{out}，反馈得到接下来要处理的误差 err'。图 4-5 展示了这个流程。

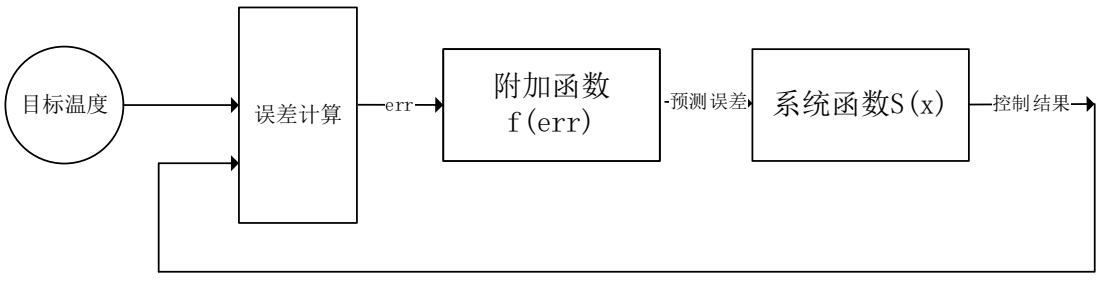

图 4-5 引入了附加函数后的结构

在本设计中，LSTM 充当了 $f(err)$ 的职责。在它被训练好发挥作用之前，需要得到数据集。在电脑仿真情景下，更好的控制结果可以通过一些 PID 整定工具获得，从而可以粗略估计附加函数的输入 err。同时，知道了控制结果，系统函数 $S(x)$ 已知，配合仿真时间要求，理论上就可以反推其输入误差。本设计并未采用上述做法，而是假定附加函数 $f(err)$ 可能满足对时间变量比例、积分、微分的形式，$f(err)$ 最初采用一个额外的 PID 控制器经调节参数实现，实际效果证实猜想可行，是一种基于经验的做法。

4.3 LSTM 训练

原始数据集是输入误差 err 附带时间戳，预定目标为 10，采样时间 0.01s。如图 4-6 所示。

Time	Data
0秒	10
0.01秒	8.683916
0.02秒	7.191309
0.03秒	6.432368
0.04秒	6.118655
0.05秒	5.959231
0.06秒	5.827002
0.07秒	5.689807
0.08秒	5.54674
0.09秒	5.401803
0.1秒	5.257422
0.11秒	5.114439
0.12秒	4.973011
0.13秒	4.833127
0.14秒	4.694768
0.15秒	4.557937
0.16秒	4.422649

图 4-6 部分数据集

LSTM 神经网络设计目标是通过历史的 50 个原始误差，预测接下来的时间误差。考虑到时间采样点不足 50 的情况，训练时创建长度为 50 的序列，元素初始值为预定温度值，后续将 lstm 收到的误差纳入序列并删除序列头部元素进行 50 个元素一组的预测任务。添加 LSTM 隐藏层，units 设置为 10，激活函数为 ReLU；添加一层 Dense 层对 LSTM 10 维的输出进行降维。模型采用 adam 函数优化，损失函数为 mse 均方差函数。Adam 优化器可以缓解梯度震荡问题，能够让 loss 较快地收敛。MSE 则是常用的损失评价函数，公式为：

$$MSE = \frac{1}{N}\sum_{i=1}^{N}(real_i - predict_i)^2 \qquad (4-2)$$

$real_i$ 代表真实值，$predict_i$ 代表测试模型的预测。训练集样本总数为 N，通过损失函数评价，LSTM 网络在多轮迭代过程中会借助得到的 loss 值更新网络参数，试图降低 loss 值。如果训练设置合理，loss 值最终会收敛在一个较小值，表示真实参考值值与训练值的差距非常小，可以认为此时的模型已经训练完毕。训练过程中的 loss 变化趋势如图 4-7 所示：

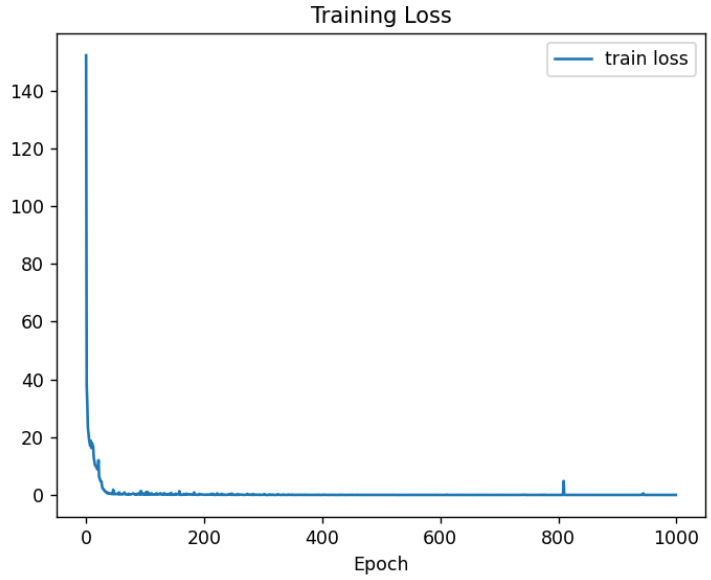

图 4-7 训练中 loss 值变化趋势

可以看到，训练不足 100 轮时 loss 已经趋近 0，800 轮附近有小幅波动，随后 loss 趋近 0。

本设计采用 sklearn 库随机划分训练集$[X, y]$和测试集$[X_{test}, y_{test}]$，两个集合的合集为整个数据集，但彼此互斥。通常训练集占据更大比重，训练集用于训练 LSTM 神经网络，测试集用于检验训练好的模型的预测效果。检查模型训练的效果的方式为：模型基于 X_{test} 预测 y_{test}'，接着使用 y_{test} 和 y_{test}' 计算均方差：

$$mse = \frac{1}{N}\sum_{i=1}^{N}\left(y_{test_i} - y_{test_i}'\right)^2 \qquad (4-3)$$

好的模型应当得出的结果是 mse 接近于 0。实际测试集检验 mse 为 3.804795755107494e-05，符合预期。

4.4 Python socket TCP/IP 通信模块

Python 的 socket TCP/IP 服务器的逻辑是，先创建本机服务器 Server，然后绑定与客户端一致的端口进行监听。当收到 8 字节数据时（double 类型），存储字节，对数据进行转换并执行预测。预测的原始数据是 numpy.ndarray 的形式，通过 item(0,0)属性访问到其中的数据（1 维）后，使用 struct.pack 函数以 double 8 字节形式封装，使用 client.send 函数写入缓冲区，等待客户端接收。

此外，还增加了重连的逻辑，即发生 IO 异常或连接重置异常时，尝试重新建立连接。整体结构如图 4-8 所示。代码流程图如图 4-9 所示。

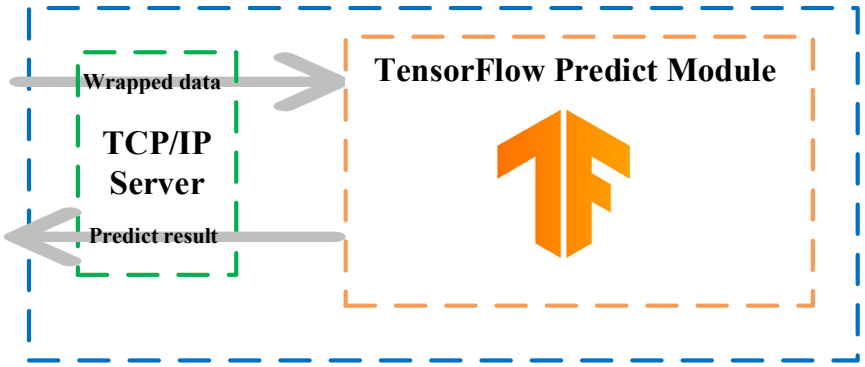

图 4-8 预测模块和 TCP/IP 通信模块结构

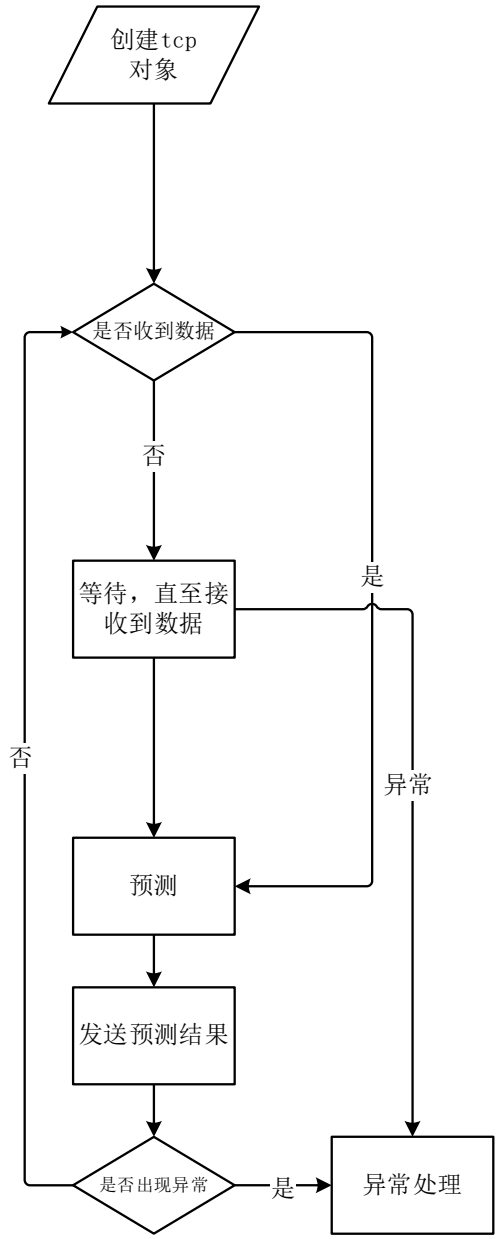

图 4-9 代码流程图

4.5 本章小结

本节主要讨论了 LSTM 相关模块的设计和实现问题。首先介绍了 LSTM 预测模块的结构，接着从获取数据集这个问题展开，分析了传统 PID 的不足，列举了模糊 PID 优化的方法。随后指出模糊 PID 优化存在的问题，并针对新的工况下的 PID 优化做了分析，提出了误差控制的思路。随后通过经验的手段，利用 PID 整定得到了合适的数据集，对 LSTM 进行了训练，分析了 LSTM 训练的评价指标。最后，介绍了 Python socket TCP/IP 通信的实现方式。

5 设计验证

本节介绍仿真测试流程，并对仿真结果进行分析。

5.1 仿真

此时，系统整体结构如下图 5-1 所示：

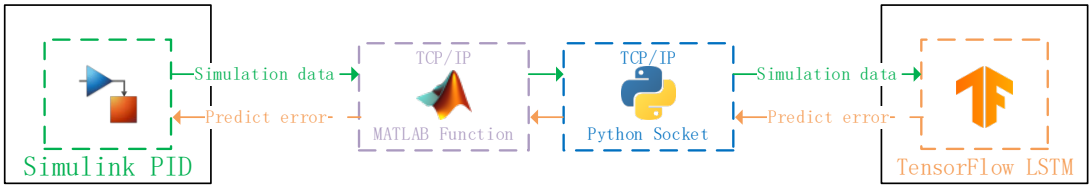

图 5-1 系统整体结构

根据 TCP/IP 要求，先运行 Python 服务器部分，在 Pycharm 软件内点击“运行”箭头按钮，程序开始运行，TCP 服务器开始等待监听：

PID 模块因为一开始并没有获取实际的数据集，故先采用一个设定值 10，接着增补一个 PID 控制器作为假想的优化函数，使用控制变量的方法调节 PID 控制器的参数，也可以采用开环响应整定法和闭环振荡整定法，后者需要一定的计算。[20]

控制变量法（单一变量法）是一种凭借实际仿真结果整定 PID 参数的基础方式，它依次连接 P、I、D 三种信号线，每次只调节一个参数，保证图像调整到合适的、可收敛的情形，最终得到的参数整定结果即可满足 PID 控制的需求。这种办法具有简单易操作、可靠性高的优点。

采用控制变量法调节优化函数（PID 形式）和原始 PID 控制器参数的结果如表 5-2 所示：

参数	优化函数	原始PID
P	5	15
I	0.02	0.9
D	3	5

表 5-1 整定后参数

运行保存得到 err.mat、pid.mat，表征原始误差和需要给 PID 的输入。均为时间序列形式。

通过以下命令可以将 err.mat 存储成 csv 格式的文件，供后续训练程序读取构造数据集：

```
load('err.mat', 'err')                          % 将 err.mat 加载为变量 err：
err_tt = timeseries2timetable(err)              % 将时间序列转换为时间表
writetimetable(err_tt, '2022_err_result.csv',
'Encoding', 'GB2312')                           % 将时间表保存为 2022_err_result.csv
```

接下来获取 csv 文件，进行模型训练，之后便可以用于验证。

开启 Simulink 仿真，如图 6-2 所示，点击“运行”绿色箭头按钮，系统开始运行。

运行完毕，得到仿真结果图像如图 5-3 所示：

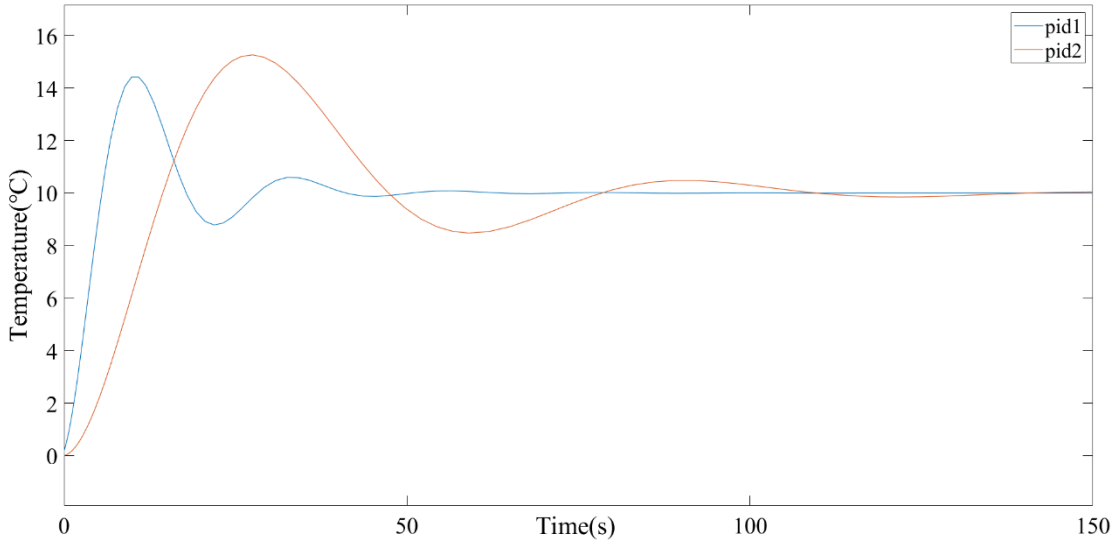

图 5-3 测试仿真图像

5.2 数据分析

对一个控制系统，应当从动态性能指标来评价以上优化结果。

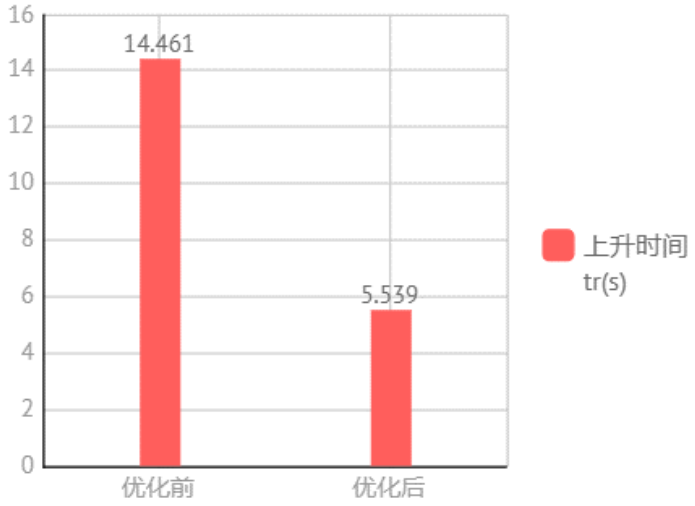

图 5-4 优化前后上升时间 t_r对比

如图 5-4 所示，上升时间 t_r（Rise Time） 表征从开始仿真到达到预定温度值的时间，通常情况下，希望系统可以有较短的上升时间，即更快地提升到预期值并开始调节。优化前 t_r 为 14.461 s，优化后 t_r 为 5.539s。缩短为原来的 38.3%。

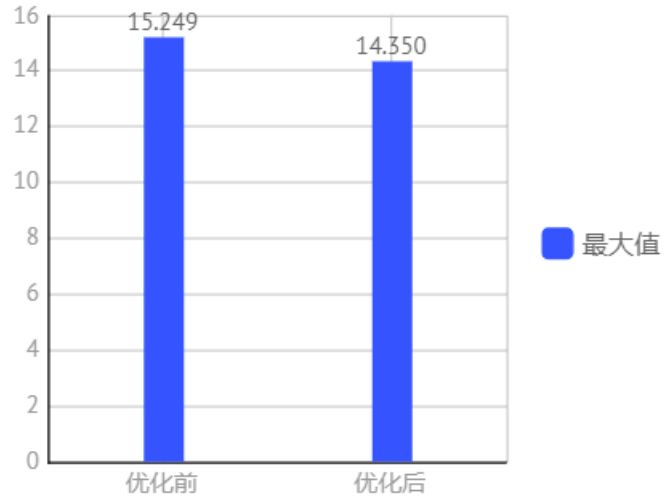

图 5-5 优化前后最大值对比

如图 5-5 所示，调节最大值表征系统响应最大值（温度最大值）T_{max}。目标温度为 T_0，可由式（5-1）计算超调量（率）$\sigma\%$。

$$\sigma\% = \frac{T_{max} - T_0}{T_0} \times 100\% \qquad (5-1)$$

根据式（5-1），计算得到优化前超调量为 52.49%，优化后的超调量为 43.50%。

接下来考察峰值时间 t_p（Peak Time）。峰值时间表征了响应超过其终值到达第一个峰值所需的时间，可以衡量系统的响应速度。如图 5-6 所示。

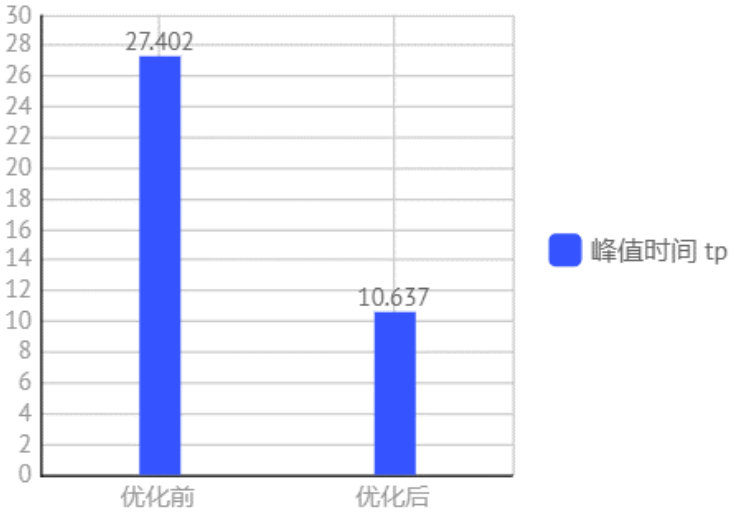

图 5-6 优化前后峰值时间 t_p对比

优化前的峰值时间为 27.402 s，优化后的峰值时间为 10.637 s。计算得到优化后的峰值时间缩短到优化前的 38.82%。

考察调节时间 t_s（Setting Time），它表征系统响应达到并保持在终值±5%之间振荡所需的最短时间。t_s 是评价系统响应速度和阻尼程度的指标之一。优化前后的 t_s 如图 5-7 所示。

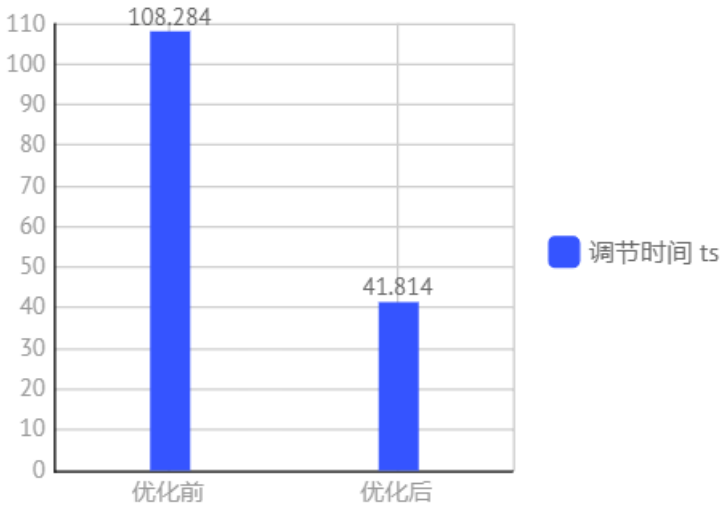

图 5-7 优化前后调节时间 t_s对比

优化前的调节时间为 108.284 s，优化后的调节时间为 41.814 s，计算得到优化后调节时间缩短到调节前的 38.62%。

5.3 本章小结

本章节首先介绍了在 Simulink 软件和 Python IDE Pycharm 中设计验证的流程。之后，根据实际仿真图像，从系统动态性能角度进行对比，分为上升时间 t_r、超调量 $\sigma\%$、峰值时间 t_p 以及调节时间 t_s 四个方面进行比较。经过数据对比计算得到，动态时间性能指标性能提升较大，时间缩短为原来的约 40%；优化后也得到了更小的超调量，但效果不是特别明显。

通过上述实际数据比较，可以认为本设计采取的优化方法是有效的。

6 总结与展望

PID 的控制优化一直是工业控制研究界关注的热点问题。本设计介绍了已有的几种优化方法，主要分析了模糊 PID 控制器的改进点以及其存在的不足之处，针对 PID 控制器在不同工况下的特点，提出了一种在锅炉温度控制工况下，对 PID 控制器进行非参数调节优化的思路，即对误差进行预测处理，通过 LSTM 模型对历史数据进行处理得出合理的误差预测结果，使得控制结果具备更好的动态性能。

本设计实现了一个 TCP/IP 实时通信框架，配合 Simulink 搭建的 PID 控制仿真系统与 TensorFlow LSTM 预测程序，在 PC 端完成了锅炉温度控制的在线模拟仿真，验证了优化思路的可行性和有效性。本设计的探究思路对后续深入研究有一定参考价值，特别适合 PID 控制器参数固定或不易调节的情况，在仿真分析阶段就可以得到理论上较好结果，并直接运用于控制流程中。

本设计完成的工作概述如下：

1. 使用 Simulink 仿真平台，借助 Simulink 提供的模块，结合自实现的 MATLAB FUNCTION TCP/IP 通信客户端代码，构建了一个可以将仿真数据对外传送，适合后续对数据进行分析处理的 PID 仿真系统；
2. 根据实际需要，设计了 LSTM 相关结构，使用 TensorFlow Keras API 实现了 LSTM 神经网络，使用历史的 50 个历史采样点误差数据，对接下来的一个采样点的输入误差进行预测。
3. 完成一个实时通信框架。在 Simulink 部分使用 MATLAB 语言实现了一个 TCP/IP 客户端，在 Python 部分使用 Python Socket API 实现了 TCP/IP 服务器。这使得 Python LSTM 模块能够在线地进行预测、传输预测结果，系统能够应对实际工况中的更多问题。相比于离线的训练、预测，用既有的预测结果去检验优化的 PID 系统的性能，在线仿真方式更加便捷、同时让本优化方式真正具备了实际使用的意义。
4. 提出了一种优化思路，即修改原始误差，突破 PID 参数和被控对象传递函数对误差的限制，引入其他模块对误差进行优化处理。提出了优化函数的一种可能形式，即优化函数包含比例、积分、微分因子，对原始误

差先执行基于时间序列的运算处理。并通过额外 PID 参数整定，得到了符合要求的数据集。

除了上述完成的工作外，设计中存在一些不足、待改进和深入的部分，主要包括以下几个方面：

1. 优化函数的相关探究还停留在主观猜想可能形式并通过实际仿真验证的阶段。对“更好的结果”定义较为模糊，目前是从图像上获得直观的反馈和几个动态指标的定量分析来判断。后期可以尝试对优化函数的可能形式或者弱化形式进行推导，使用控制变量的思路，针对不同的优化指标进行函数设计，或可以基础线性函数拟合为起点做起，得到较为通用的优化函数的形式。
2. LSTM 的作用在本设计中是替代优化函数$f(err)$。而 $f(err)$ 已经经过理论推导，LSTM 的实际作用是根据 $f(err)$ 的数据集完成训练，接入系统进行后续实际验证，因此本设计 LSTM 预测模块的适用范围较为单一，仅限于当前工况，不具有很强的普适性。本设计探寻优化函数的思路以及对优化函数形式的猜想也未必适用于更为复杂的工况，后期可以从性能评价指标入手，预定义性能指标评价规则，如对一次仿真中的调节时间、震荡次数、超调量等赋予不同的评分权值，设置以权重为参数的线性函数或更为复杂的非线性函数形式，通过强化学习的手段，自动地完成对优化函数参数的探索过程，从而可以适应任意场景的优化。
3. Simulink 平台的外部通信速度较为受限，仿真时间存在优化空间。Simulink 中使用 MATLAB Function 会造成额外的外部函数调用性能开销，在后续的优化中，可将本设计中的 Simulink 系统全部转换为 MATLAB 程序设计语言实现，以提升系统运行速度。

参考文献

[1] 路桂明.基于模糊 PID 控制的电锅炉温度控制系统的研究[D].2007

[2]多晓艳.工业锅炉智能控制系统的研究[D].2019

[3]张文胜,龚希波.能源工程.电热锅炉的市场前景和发展方向[J]. 能源工程 2001,(05),52-55

[4]王丽娟.模糊控制技术在电热锅炉温度控制系统中的应用研究[D].2007

[5]胡寿松.自动控制基础[M].北京:科学出版社,2013.3

[6]吴宏鑫,沈少萍.PID 控制的应用与理论依据[J].控制工程.2003-01

[7]刘金琨.先进 PID 控制 MATLAB 仿真[M].北京:电子工业出版社,2004.9

[8]王伟,张晶涛,柴天佑.PID 参数先进整定方法综述[J].自动化学报.2000,(03)

[9]孙林军.智能 PID 控制研究[D].2003

[10]Ziegler, J.G and Nichols, N. B. Optimum settings for automatic controllers. Transactions of the ASME: 759–768. 1942

[11]何芝强.PID 控制器参数整定方法及其应用研究[D].2005

[12]何佳佳,侯再恩.PID 参数优化算法[J].化工自动化及仪表.2010,37(11)

[13]贾歆玮,唐艳军,程益民.基于模糊逻辑的纸张定量水分自适应 PID 控制[J].中国造纸. 2022,41(04)

[14]程龙,韩森,李彤彤,田艳兵.基于遗传算法的滑模控制系统在锅炉温度控制中的应用研究.现代计算机(专业版). 2016,(24)

[15]刘金琨,孙富春.滑模变结构控制理论及其算法研究与进展[J].控制理论与应用.2007,(03)

[16]李盛伟,张来,梁海深.基于串级 PID 的相变储能电锅炉温度控制技术.热能动力工程.2020,35(08)

[17]Ayako Mikami.Long Short-Term Memory Recurrent Neural Network Architectures for Generating Music and Japanese Lyrics.2016 Honors Thesis, Boston University

[18]天工在线.中文版 MATALB2018 从入门到精通：实战案例版[M].北京:中国水利水电出版社,2018.08

[19]陶翠,白焰,张方杰.锅炉仿真实验系统的建模[J].工业控制计算机. 2009(10)

[20]胡晚霞,余玲玲,戴义保,何亨文.PID 控制器参数快速整定的新方法[J].自动化与仪器仪表.1996(05)

致谢

光阴荏苒，我在湖南大学的本科学习生活时光已经接近尾声。慨叹时光飞逝之余，我思考了很多事。这四年，我学到了新知识，对世界有了新的认识，对自己的未来有了规划，看到了自己想要成为的模样。这样的转变，与一路上帮助我的人是分不开的。

我要特别感谢指导我的佘兢克老师。本论文能够顺利完成的基础是我有从大三开始在老师的科研小组内进行研究的经验。佘老师平易近人，对待学问一丝不苟，在严格要求我完成指定任务的同时，也积极鼓励我探寻新思路、新想法，与学长学姐多多交流。佘老师关于“本科生科研”的尝试是非常有远见的，这让我在本科中期阶段，我就从这样的科研中得到了锻炼，为后续打下了基础。

我还要感谢杨溯源学姐，她对我的工作提出了很多有益的意见；我时常和她就我不理解的一些问题进行探讨，她总是耐心解答；有帮助自己的学长/学姐是非常令人感到安心和愉快的一件事。如过没有她的帮助，我还要在很多问题中探索更长时间。

我衷心感谢张英杰老师和他的科研小组里的学长学姐。张老师教授了我本科智能控制课程，他对待备课一丝不苟的态度给我留下了极其深刻的印象。在我的毕业设计遇到控制相关的问题时，我也曾向张老师寻求过帮助，他和研究生们不顾自己工作繁忙，欣然接受了我的请求，对我的疑问进行了分析和解答，让我茅塞顿开，彻底打开了思路。没有他们的帮助，这份毕业设计是不能完成的。

我要感谢我的父母。他们的支持始终是我前进的最大底气。有了他们的“后援”，我才能在人生路上奋力前行，拥有试错的勇气。没有他们，我不可能站在今天这个位置上。

最后，我要感谢我的同学和其他老师们，感谢开源社区的朋友们。通过与他们接触，我广泛地学习知识，逐渐形成了开放、包容的心态以及对待自己感兴趣的事情一丝不苟的态度。这使我受益匪浅。

人生路上，我得到了许许多多的人的帮助。离开校园后，我将怀着一颗感恩的心，在新的路上前行。在湖南大学所学习到的一切将是我最珍贵的宝物。