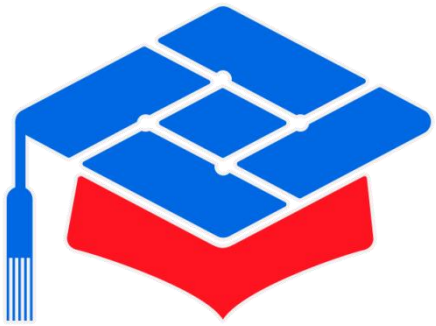


第九届

全国大学生集成电路创新创业大赛

报告类型：实验总结报告

参赛杯赛：算能杯

作品名称：基于 SG2000 算能芯片的 ECG 智能分类系统

队伍编号：CICC0907858

团队名称：算了

目录

1. 模型的搭建

1.1 作品简介

本项目聚焦心血管疾病早期精准诊断需求，基于深度学习与边缘计算技术，利用算能SG2000芯片的边缘计算加速模块，构建了一套高性能的心电信号（ECG）自动分类系统。

首先，系统梳理近年心电信号处理领域的研究进展，深入掌握深度学习算法在生物医学信号分析中的应用范式；其次，调研全球主流心电数据库（如欧盟ST-T 数据库），针对数据标签体系、采样频率（250-360Hz）、导联方式（单导联 / 多导联）等关键维度进行系统性分析，筛选出符合临床需求的 7 类心律失常标签（N - 正常、V - 室性早搏、A - 房性早搏等），并完成多源数据的频率统一与标准化预处理（滤波、切割、Z-score 归一化）。

在算法层面，提出 CNN+SE-Net（Squeeze-and-Excitation Networks）模型架构：利用 CNN 的多层卷积核（15/10/5 尺度）实现宽时域波形（QRS 波群）、中等尺度特征（P/T 波）及细微偏移（ST 段）的层级化特征提取，结合 SE 的通道注意力机制，动态校准特征通道权重，强化病理相关信号分量（如异常 QRS 波形态）。模型在欧盟 ST-T 数据库上实现 96% 的 7 分类准确率。

硬件部署方面，将训练好的模型通过 ONNX 格式转换，优化适配 SG2000 算能芯片（0.5TOPS INT8 算力）的模型，利用 PC 和算能 SG2000 芯片网络连接交互，前处理在 PC 机，利用 SG2000 算能芯片 NPU，推理在 SOC，然后结果传回 PC 做后处理，单样本推理时间 ＜10ms，模型体积压缩至 10MB，满足便携式设备的低功耗、实时性要求。后期可以采用 ADI-ECG 传感芯片或者国产 ECG 传感芯片做实时心电信号采集，利用 SG2000 芯片多核特点，实现便携式采集处理，WiFi 信息传递等特色 。

该方案为心血管疾病的早期筛查，长时间连续监控，远程监测提供了高精度、可部署的技术解决方案。

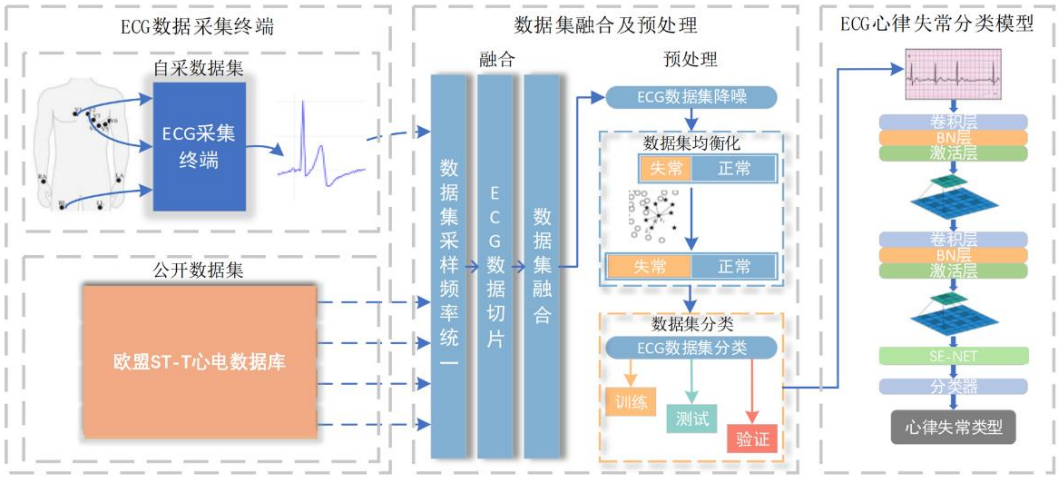


图 1 心电信号（ECG）自动分类系统

1.2 研究背景

从上世纪九十年代到现在，全球心血管疾病患病人数正在持续增长，已由 2.71 亿人增长到 5.23 亿人，因心血管疾病死亡的人数也在不断增加，全球每年约有 1790 万人死于心血管疾病。而在中国，目前心血管疾病患者已突破 3 亿人，中国居民因心血管疾病死亡人口已占总死亡人口的 46.66%[2]，心血管疾病已成为“人类健康第一杀手”。由此看出，如何预防心血管疾病，从而降低心血管疾病患病以及死亡人数是中国乃至世界的问题。

在心血管疾病预防和诊断的各种途径中，心电图(Electrocardiogram, ECG)[3]监测是最有效的一种途径，ECG 监测主要有以下几点优势：成本低廉，功耗低，精度高。心电信号是一种生物电信号，是由心脏活动中产生的，因此，通过对心电图中心电波形的各种特征能够间接反映出人体心脏以及心血管的疾病状况。

心律失常主要是由于心脏在日常活动、心电信号传导中，窦房结激动异常，或者心脏活动的起源点不是窦房结而造成的心脏活动异常行为，直接表现为心脏跳动频率、节律异常，呈现在 ECG 上的特征是 P 波、QRS 波、T 波的振幅、外观形态以及 RR 间期、PR 间期等超出正常范围，根据上述特征可以按照 AAMI 标准将心律失常分为以下几类：正常或者束支传导阻滞 N、室上性异常心拍 S、心室异常节拍 V、融合节拍 F、未能分类节拍 Q。

一个完整的心跳周期是由 P 波、QRS 波群、T 波、U 波及间期波形组成，每种心电波形及代表了不同心脏部位的各种电活动，医生在诊断的时候必须对心电波段的形状及其含义有清晰的认识。一个完整的心拍如下：

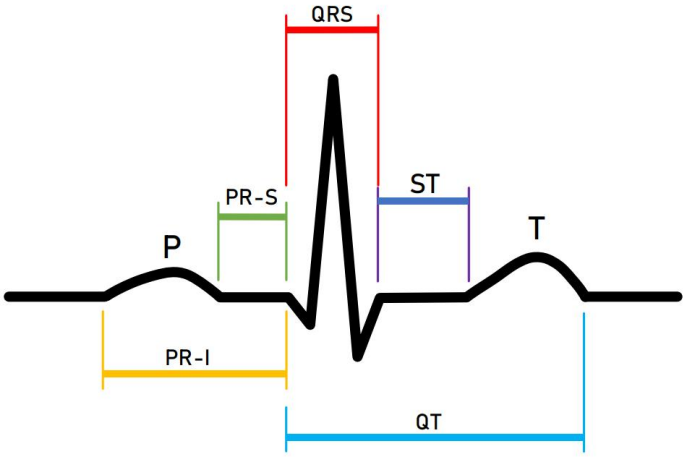


图 2 完整心拍波形

（1）P 波：P 波通常认为是整个心电周期的一个开始部分，表示心房肌除极期间的电位发生变化的具体情况。前后分别代表右心房激动、左心房激动，间期长度大约为 0.08s-0.11s，幅值不超过 0.25mV。

（2）P-R 间期：PR 间期代表由窦房结产生的兴奋致使心室肌开始兴奋所需要的传导时间，故也称为房室传导时间，间期长度一般为 0.12s-0.2.s。

（3）QRS 波群：QRS 波群是由 Q 波、R 波和 S 波组成的一个波群组。其中 Q 波是其第一个时长一般不小于 0.04s 的向下的波形，Q 波后面高尖突起的波就是 R 波，随后向下的是 S 波。QRS 波群图形所含信息比较丰富，常用这个波形来作判别心律失常类型的主要工具和依据。

（4）T 波：T 波主要代表心室快速运动复极时的电位波动变化。T 波的波形方向一般与 QRS 波群相同，间期长度大约为 0.02s-0.25s。

（5）QT 间期：QT 间期表示心室肌除极和心室肌复极的运动全过程。间期长度大约为 0.43s-0.44s。

（6）U 波：U 波出现在 T 波之后 0.02s-0.04s，振幅低小。U 波方向一般与 T 波方向相同，振幅大约为 T 波的一半。

我们通过用户的 ECG(electrocardiogram)信号来判断用户的心脏状况，一段 10s 的 ECG 信号图像如下所示:

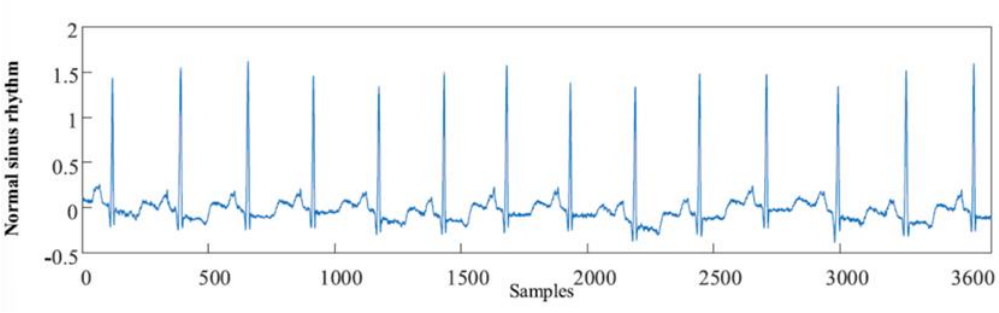


图 3 10s 的 ECG 信号图像

这张 10 秒的 ECG 心电信号图像显示了正常的窦性心律。图中可见多个形态一致、间隔均匀的尖峰，代表心室去极化的 QRS 波群，表明心脏电活动规律，无明显心律不齐或异常波形（如早搏）。信号幅度在合理范围内（纵轴-0.5 至 2），背景虽有微小波动（可能为基线漂移或生理噪声），但未对主要波形造成显著干扰。该图像反映了心脏电活动的正常节律与规律性，无明显病理特征。

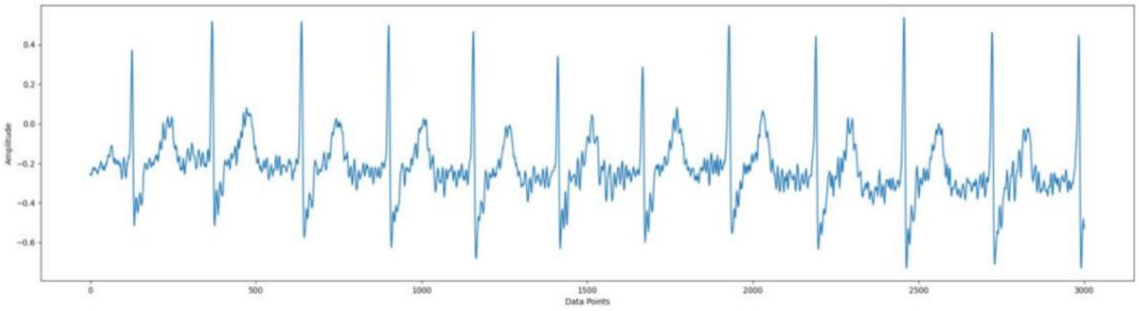


图 4 S 类心电信号波形图

S（Supraventricular）：这类涉及那些起源于心房或房室结上方的心律失常。特征包括异常的 P 波（例如在房颤中不存在）或异常的 P-R 间期，以及通常较快的心率，其波形如图 4 所示。

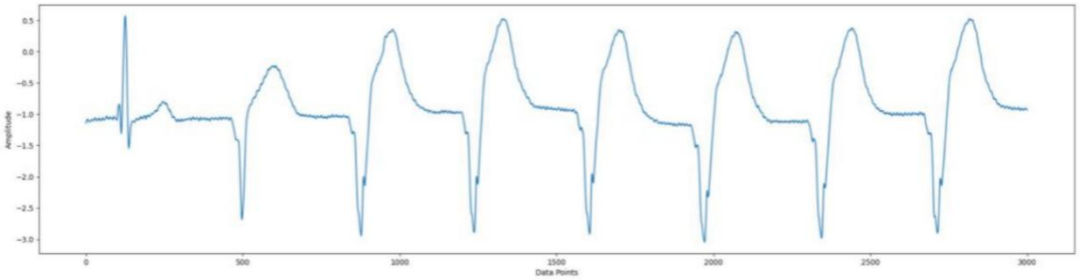


图 5 V 类心电信号波形图

V（Ventricular）：这类包括起源于心室的心律失常。其特征是宽大且畸形的 QRS 复合波，通常没有前导的 P 波或 P 波与 QRS 复合波之间的关系被打乱，其波形如图 5 所示。

心电图作为心脏疾病诊断的关键工具，自诞生五十载以来，计算机辅助分析在临床心电图诊断流程中的作用与日俱增，成为众多医疗机构中医生判读结果的重要助力。然而，当前市售的心电图分析算法仍面临较高误诊挑战。传统机器学习[4]方法需人工完成滤波、特征提取、小波变换等繁琐操作，这类数据预处理过程对医疗领域专业经验依赖度较高，且显著影响模型最终性能。近年来，学界与业界开始探索将深度学习技术应用于原始心电图数据分类，已取得显著进展。相较于传统机器学习，基于深度学习的卷积神经网络可自主挖掘数据特征，无需人工干预特征提取环节，有效规避了因行业经验不足对模型分类效果造成的潜在干扰。

1.3 深度神经网络选择

深度学习是机器学习的重要分支，基于多层人工神经网络架构，通过模拟人脑神经元的信息处理机制，实现对复杂数据的自动特征提取与模式识别。深度学习的蓬勃发展为心电图识别开辟了全新路径，其构建的一体化处理模式可实现从原始心电数据到分类结果的直接映射，无需传统流程中人工干预的滤波操作与特征工程。实际应用中，只需将心电信号按规则分割为样本片段，即可直接输入模型进行分析。在此过程中，经典的卷积神经网络（CNN）承担特征提取任务，其内部的每个卷积核如同信号过滤器，能够自动从原始波形中筛选并强化含病理信息的特征成分，通过多层网络的层级抽象，最终实现对心电图类别的精准判别。

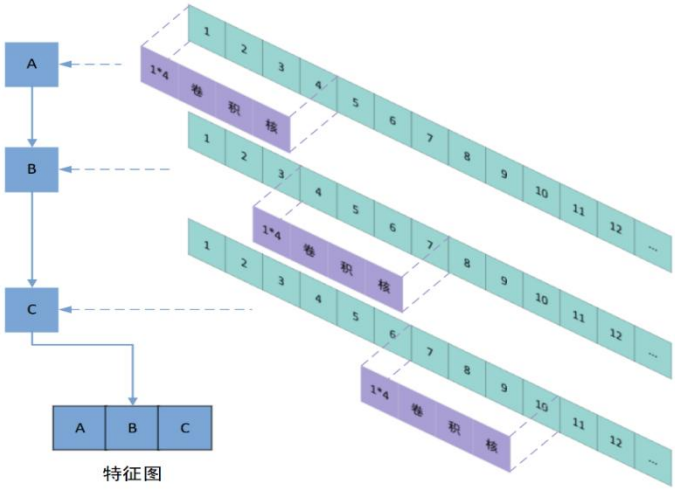


图 6 心电信号的一维卷积过程

卷积神经网络里会包含很多堆叠的卷积层，卷积层的核心是卷积核，卷积核的数量和尺寸不固定，可以是不同尺度的卷积核进行组合，这样可以提取目标的多尺度特征[4]。需要明确的是，对于心电信号而言，卷积核的尺寸必须是 1*N 形式的一维卷积核，如图所示，心电数据可以表示成一行排列整齐的一维数据，以 1*4 尺寸的卷积核、步长为 3 为例，每次移动都会提取覆盖面积的特征，经过从头到尾依次滑动，最终得到了下方的卷积结果。

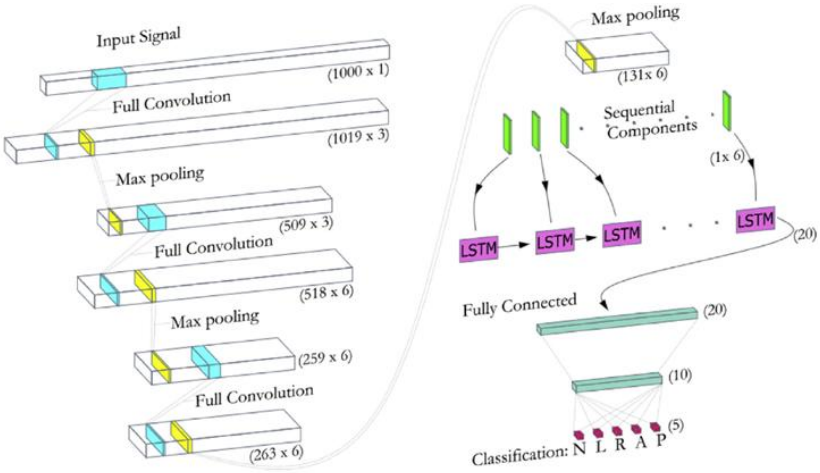


图 7 CNN+LSTM 模型

CNN+LSTM 模型是一种融合卷积神经网络（CNN）与长短期记忆网络（LSTM）的混合架构。CNN 通过卷积层和池化层，能够自动提取输入数据的局部空间特征，例如在 ECG 信号中捕捉波形的形态、幅度等局部模式；而 LSTM 凭借输入门、遗忘门和输出门的独特设计，擅长处理时间序列数据，捕捉特征随时间变化的长期依赖关系。在图中模型里，CNN 先对输入信号（如 1000×1 的序列）进行卷积和池化操作，提取多尺度的局部特征（如不同层次的波形特征），再将处理后的特征输入 LSTM，由 LSTM 进一步分析特征在时间维度上的动态变化规律，最后通过全连接层整合信息完成分类（如图中对 ECG 信号对应类别 N、L、R 等的判断）。这种结合使模型既能挖掘数据的空间局部模式，又能处理时序依赖，在医疗信号分析、金融预测、天气预报等涉及时空数据的领域具有强大的应用能力。

但本项目并没有采取 CNN+LSTM 模型，而是采用 CNN+SE 模型。是综合考量硬件特性、模型效率、应用场景及开发支持等多方面因素的结果。从硬件角

度看，SG2000 的 NPU 算力仅为 0.5TOPS（INT8），属于轻量级配置，而 LSTM 作为循环神经网络，内部复杂的门控机制（输入门、遗忘门、输出门）需在每个时间步进行大量矩阵运算与频繁内存访问，计算复杂度和资源消耗量极大，SG2000 有限的算力与存储难以支撑其高效运行，可能导致推理延迟高或无法实时处理任务。相较之下，CNN+SE 模型中的 SE-NET 模块是一种轻量级注意力机制，仅通过全局平均池化、少量全连接层运算等简单操作调整通道权重，在几乎不显著增加计算开销与资源占用的前提下，增强了 CNN 对关键特征的提取与关注能力，更契合 SG2000 的硬件条件。从应用场景分析，若 SG2000 面向的任务（如嵌入式视觉中的图像分类、目标检测，或简单信号处理等）更侧重空间特征提取，而非处理长时序依赖关系，那么 CNN+SE 已能有效完成任务，无需引入 LSTM 的时序处理能力，避免冗余计算。此外，在开发支持层面，SG2000 的软件生态可能对 CNN+SE 这类轻量级模型有更充分的优化，包括硬件加速、内存分配、算子优化等底层支持，确保模型高效运行；而 LSTM 在该开发板上可能缺乏针对性优化，实际部署难度大、效率低。所以 CNN+SE 模型在计算效率、资源利用率、场景适配性及开发友好性上更贴合 SG2000 的特性与需求，是兼顾性能、效率与实用性的优选方案。

CNN+SE 模型是一种结合了卷积神经网络（CNN）与 SE 模块（Squeeze-and-Excitation Block，SE 模块）的深度学习架构，旨在通过引入通道注意力机制提升模型对特征的筛选和表达能力。

SE 模块：一种通道注意力机制，用于建模特征通道间的依赖关系，让模型自主学习“哪些通道的特征更重要”。挤压对卷积层输出的特征图进行全局平均池化，将空间维度压缩为一个通道描述符（向量），汇总每个通道的全局信息。激励通过全连接层和激活函数，计算每个通道的权重，表征该通道特征的重要性。

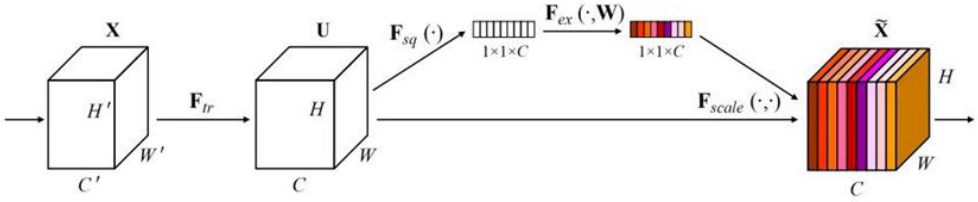


图 8 SE 模块图

SE 模块（Squeeze-and-Excitation Module）的核心作用是通过对特征通道的自适应校准，提升模型对关键特征的关注能力。其机制如下：首先进行挤压（Squeeze）操作，通过全局平均池化等方式，将输入特征的空间维度信息压缩，把每个通道的特征信息聚合为一个数值，以此获取每个通道的全局统计信息，实现对通道信息的整体感知。接着是激励（Excitation）操作，利用全连接层等结构对挤压得到的信息进行处理，生成一组通道权重，这些权重代表了各个通道特征对于当前任务的重要程度。最后，将这些权重乘回到原始特征上，对不同通道的特征进行加权调整，使模型能够重点关注对任务更重要的特征通道，抑制次要通道。在 ECG（心电图）分类中，SE 模块具有重要作用。ECG 信号包含多种特征（如 P 波、QRS 波、T 波等形态特征），不同通道的特征对分类（如判断正常心律、心律失常类型等）的贡献度不同。SE 模块能够自动识别并增强对 ECG 分类起关键作用的特征通道（例如，反映特定心律异常的特征所在通道），抑制无关或次要的通道特征。通过这种方式，模型可以更精准地捕捉 ECG 信号中的关键信息（如异常波形的形态、幅度、间隔等），减少噪声和冗余信息的干扰，从而提升 ECG 分类的准确性和鲁棒性，使模型在区分不同类型的 ECG 信号时表现更优。

表 1 不同注意力机制对比

注意力机制	mAP50 (%)	FPS (f/s)
SE	90.3	82
ECA	91.2	80
CBAM	91.5	70
CA	83	103
SimAM	44	64
S2Attention	58	78
NAMAttetion	31	51

为验证不同注意力机制对目标检测模型性能的影响，表 1 是 YOLOv5s 仅更换注意力机制模块的实验数据， 实验数据显示，SE 模块拥有 90.3%的 mAP50 与 82 FPS，表现出色，并且模块更加轻量化。

CNN+SE 模型通过卷积神经网络（CNN）自动从原始心电信号中提取波形形态、间期等空间特征，无需人工进行滤波、特征工程等预处理操作，而挤压激

励模块（SE）则通过通道注意力机制对 CNN 提取的特征通道进行权重校准，抑制无关信号并强化与心律失常、心肌损伤等病理相关的关键特征，形成“特征提取-通道优化”的端到端处理流程，有效提升心电数据分类的准确性和鲁棒性，为临床心电图自动分析提供了更高效的解决方案，尤其在减少误诊率、辅助医生快速识别潜在心脏异常方面具有显著价值。

1.4 CNN+SE 模型架构深度解析

1.4.1 CNN+SE 模型细节

多尺度卷积设计：

第一层卷积（kernel_size=15）：捕获 QRS 复合波等宽时域特征。

第二层卷积（kernel_size=10）：聚焦 P 波、T 波等中等尺度特征。

第三层卷积（kernel_size=5）：提取 ST 段偏移等细微特征。

池化层作用：每次 MaxPooling1D(pool_size=2)将序列长度减半，减少参数同时保留关键特征，增强模型对信号平移的鲁棒性。

示例图如下

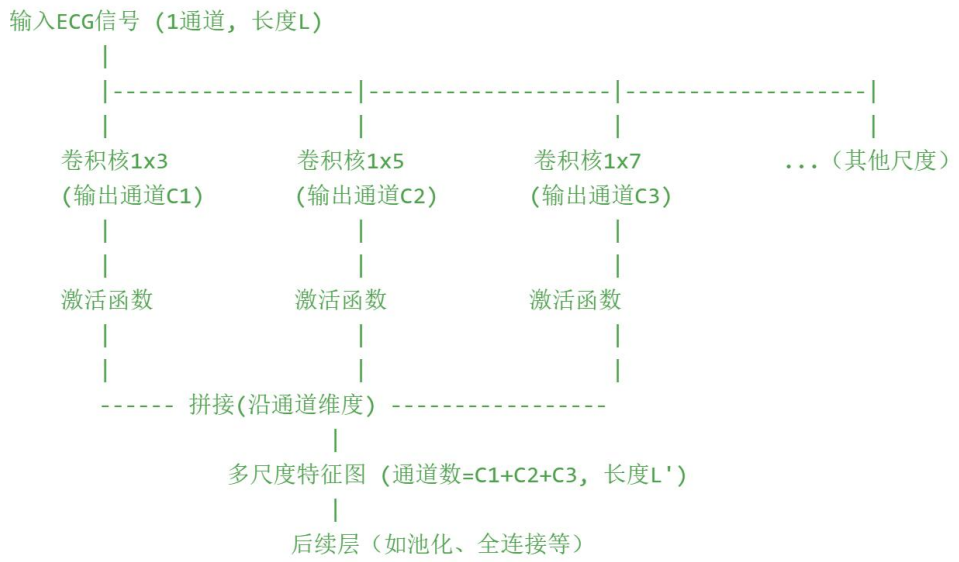


图 9 多尺度卷积流程

1.4.2 SE 模块工作原理

通道注意力计算：Squeeze 通过全局平均池化将[batch,450,64]特征压缩为[batch,64]，提取全局上下文信息。

Excitation：

第一层全连接（神经元数=64/16=4）：降维并引入非线性变换。

第二层全连接（神经元数=64）：恢复维度并输出通道权重（范围 0-1）。

Scale：将权重与原始特征逐通道相乘，增强重要通道（如 QRS 波对应的通道）。

1.4.3 模型整体信息流

原始信号→卷积块 1（提取特征）→SE1（注意力特征）→卷积块 2（组合基础特征）→SE2（筛选组合特征）→卷积块 3（生成高级抽象）→SE3（轻量化注意力特征）→分类器。

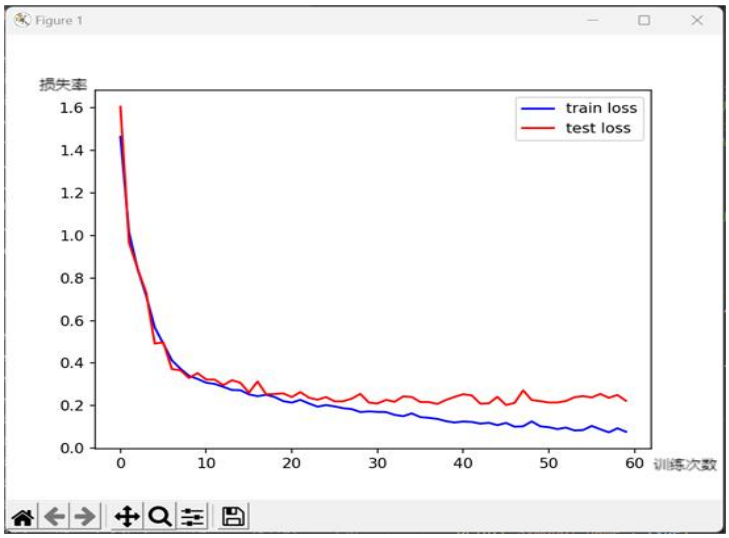


图 10 ECG 模型训练结果图

这张图展示了 ECG 模型训练过程中训练损失（蓝色）和测试损失（红色）随训练轮次（epoch）的变化趋势。初始阶段，两者损失值均较高（接近 1.6），随后快速下降，尤其在前 10 个 epoch 下降显著。随着训练推进，训练损失持续降低并趋于平稳（接近 0），而测试损失在下降后，于 0.2-0.4 间波动。这表明模型对训练数据的拟合能力逐渐增强，但测试损失未持续同步降低，暗示可能存在

一定程度的过拟合，即模型在训练数据上表现优异，但在测试数据上的泛化能力有提升空间，需进一步优化以平衡训练与测试的性能表现。

符号	描述
N	正常
L	左束支传导阻滞心搏
R	右束支传导阻滞心搏
V	室性早搏
A	房性早搏
I	孤立 QRS 波样伪像
B	左或右束支传导阻滞

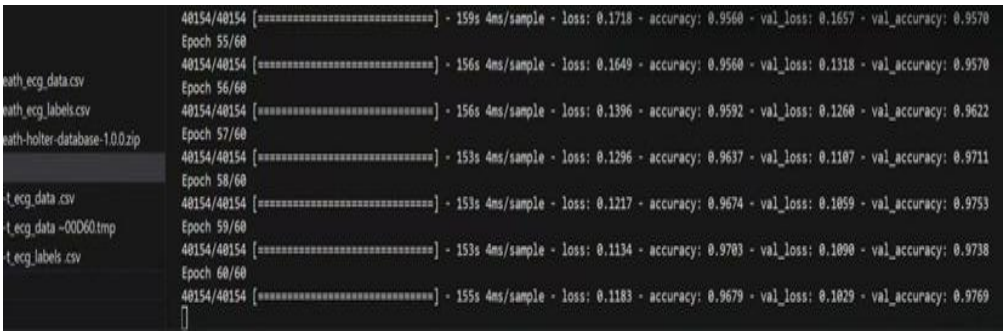


图 11 ECG 模型训练准确率结果图

从训练输出看，模型训练过程中损失逐渐降低，六种不正常心电图识别准确率逐步提升。如在早期 Epoch55/60 时，训练损失为 0.140，训练准确率达 0.9560，验证准确率为 0.9570；随着训练推进，到 Epoch60/60 时，训练损失降至 0.1183，训练准确率提升至 0.9679，验证准确率更是达到 0.9769。这表明模型在训练中有效学习数据特征，对训练数据和验证数据的拟合效果不断增强，展现出良好的学习能力与泛化能力，最终实现了接近 98%的准确率，而单独使用 CNN 准确率只有 78%。

2.数据集下载及数据预处理

目前国际上最重要的且具有权威性的心电数据库有四个：

- MIT-BIH 心电数据库：由美国麻省理工学院与 Beth Israel 医院联合建立；
- AHA 心律失常心电数据库：由美国心脏学会建立（需付费下载）；
- CSE 心电数据库：由欧盟建立（需付费下载）；
- 欧盟 ST-T 心电数据库。

除此之外，国际上被广泛认可的还有 Sudden Cardiac DeathHolter Database 等心电数据库。

我们选择欧盟 ST-T 心电数据库进行下载和处理，如下：

欧盟 ST-T 心电数据库（https://physionet.org/content/edb/1.0.0/）

下载解压后得到包含下列后缀的总文件夹：

(1) hea：头文件（可以理解为数据的注释文件），该文件含有记录编号、导联方式、采样频率、采样点数等信息；

(2) atr：标记文件，该文件含有人工标注的心拍位置和类型（如：异常心拍类型的字母标记）；

(3) dat：心电信号数据（主体）。

表 2 数据集详细介绍

数据集名称	采样频率	导联方式	每段数据持续时间	标签情况	储存格式	其他
European ST-T Datebase	250Hz	MLI,MLIII,V1,V2,V3,V4,V5,D3	90 条持续时间为两小时的动态心电图记录	均含有人工标注的心拍注释	Format 212	79 位受试者

WFDB 是一个用于读取、写入和处理 WFDB 信号及注释的工具库，由 WFDB 库、用于信号处理与自动分析的应用程序，以及支持可视化、注释和波形数据交互式分析的 WAVE 软件组成，采用高度可移植的 C 语言编写，能在 GNU/Linux、MacOS/X、MS-Windows 等多种流行平台上使用。其功能包括快速显示波形和注释，快速访问记录任意位置以提升效率并减少网络流量，支持注释模式向前和向后搜索，可进行图形化注释编辑（支持标准或用户自定义注释方法），实现变速叠加显示（模拟触发式示波器持续性显示效果），对用户选择的信号片段进行高

精度打印，灵活控制外部信号处理和分析程序（菜单可在 WAVE 运行时重新配置），具备远程模式（如网页浏览器等外部程序可控制 WAVE 显示），并提供在线帮助，为波形数据处理、分析及可视化提供了全面且高效的支持。

安装方法：在 vs code 命令行输入 pip install wfdb 进行安装。

DealData.py 使用 WFDB 库处理心电数据集，将原始数据分割成固定长度的片段，进行重采样和标签提取，最终保存为 CSV 格式。以下是处理流程的详细说明：定义数据库配置列表 DATABASE_CONFIGS，包含数据集名称、路径、目标导联、注释类型等参数。使用 wfdb.rdrecord 读取心电信号数据，wfdb.rdann 读取对应注释。验证数据文件和注释文件是否存在，跳过缺失文件的记录。优先选择配置中指定的导联（如 ECG），若不存在则选择第一个可用导联。将信号按固定长度（如 10 秒）分段，每段包含 segment_len*fs 个采样点。使用 scipy.signal.resample 将每段重采样至目标采样率（如 360Hz），确保所有片段长度一致。统计每个片段内的注释符号（如 N、V、A 等）。按出现频率排序，选择非 N 类中频率最高的符号作为该片段的标签；若无非 N 类符号，则标记为 N。

将所有处理后的片段合并为统一的数据集。使用 pandas.DataFrame.to_csv 将数据和标签分别保存为 CSV 文件（european-st-t_ecg_data.csv 和 european-st-t_ecg_labels.csv）。

该代码采用单导联提取策略处理心电信号，通过配置驱动的导联选择机制（在 DATABASE_CONFIGS 中指定 leads 参数），优先从可用导联中匹配首选导联（如 EuropeanST-T 数据库配置的 ECG），若首选不存在则选择第一个可用导联，确保兼容不同数据库的导联命名差异，每个记录仅提取单个导联数据，通过 select_lead 函数实现智能匹配，并支持扩展到多数据库场景，未来可通过修改配置参数扩展至多导联分析最终将数据存储为 csv 格式。数据文件（european-st-t_ecg_data.csv）每行对应一个心电片段、每列对应时间点信号值，标签文件（european-st-t_ecg_labels.csv）包含记录各片段心脏节律类型（如 N、V、A 等）的“label”列。

2.1 数据预处理的细节优化

信号标准化策略：使用 StandardScaler 对每个 ECG 样本进行 Z-score 标准化，确保特征在相同尺度，避免梯度消失。

标准化后的数据分布更适合 ReLU 激活函数，加速模型收敛。

标签处理技巧：AAMI 标准映射：将原始标签映射到 AAMI 心律失常分类标准（如 N=正常，V=室性早搏），统一医学术语。

独热编码优势：将标签转换为 7 维向量（如[1,0,0,0,0,0,0]表示正常），支持多分类交叉熵损失计算。

数据集划分策略：分层采样：通过 StratifiedShuffleSplit[5]确保训练集和测试集中各类别比例一致（如正常样本占比均为 50%），避免类别不平衡导致的模型偏差。

3.边缘计算与硬件部署

3.1 配置硬件所需的 docker 环境

Docker 安装

在 windows 环境下，可以安装 Docker Desktop for Windows，Docker 下载地址为 https://docs.docker.com/desktop/setup/install/windows-install/

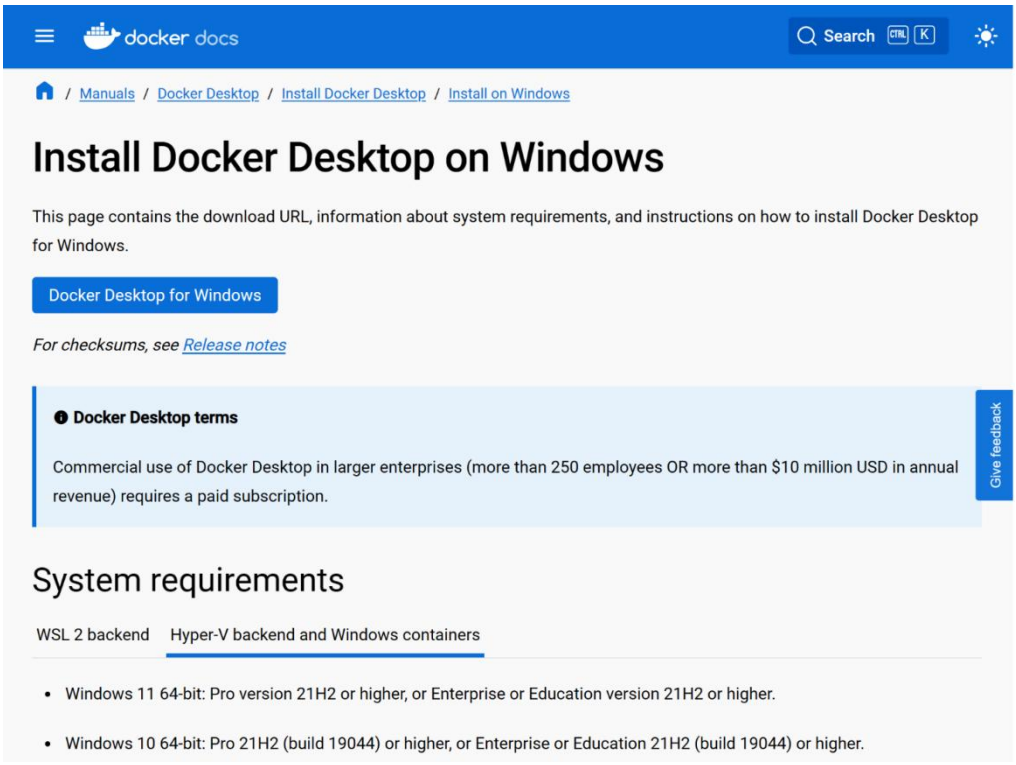


图 12 Docker 安装页

在 Windows 下运行 Docker 需要相关依赖，即如图中所示，需要使用 WSL2 后端或者 Hyper-V[6] 后端作为运行依赖 Hyper-V 后端的启用。

3.2 拉取开发所用 Docker 镜像

从 Docker hub 获取镜像文件：

图 13 Docker 镜像获取

3.3 启动 Docker 容器

在 Windows 终端中执行：docker run --privileged --name <container_name> -v

/workspace -it sophgo/tpuc_dev:v3.1

其中，<container_name>为自己定义的容器名，比如叫 DuoTPU。

3.4 登陆到 Docker 容器

启动容器后会自动登陆到 Docker 窗口的终端界面，如果需要新开窗口，可以按如下方法操作：

在 Windows 终端中用 docker ps 命令查看当前 Docker 容器列表：

PS C:\Users\Carbon\Duo-TPU> docker ps

CONTAINER ID IMAGE COMMAND

f3a060efb1d3 sophgo/tpuc_dev:v3.1 "/bin/bash"

用 CONTAINER ID 登陆到 Docker 容器中：

docker exec -it f3a060efb1d3 /bin/bash

在 Docker 终端中，检查当前是否为 /workspace 目录，如果不是，用 cd 命令进入该目录：

cd /workspace/

图 14 检查目录

3.5 获取开发工具包并添加环境变量

在 Docker 终端中下载 TPU-MLIR[7] 模型转换工具包：

git clone https://github.com/milkv-duo/tpu-mlir.git

Docker 终端中，用 source 命令添加环境变量 In the Docker terminal, use the source command to add environment variables：

source ./tpu-mlir/envsetup.sh

onnx 文件生成

3.6 将 onnx 文件配置到 SG2000 上

3.6.1 系统要求

操作系统: Ubuntu22.04

Python 3.10 硬件: SG2000 开发板（已安装 Sophon 驱动）

3.6.2 安装依赖工具

基础编译工具：

sudo apt update

sudo apt install -y cmake git g++-11 protobuf-compiler libprotobuf-dev

Python 依赖：

pip install numpy onnx==1.13.0 onnxruntime==1.14.1 opencv-python==4.7.0.72

3.6.3 安装 TPU-MLIR 工具链

下载 TPU-MLIR：

从算能官方 GitHub 仓库获取适合 SG2000 版本：

git clone https://github.com/sophgo/tpu-mlir.git

cd tpu-mlir

git checkout release-v1.9

3.6.4 编译安装

在编译安装 TPU-MLIRv1.9 版本时，首先要更新系统包索引并安装编译工具、LLVM/Clang（含对应开发库）、图像及其他依赖库，同时升级 pip 并安装 numpy、onnx 等 Python 包；接着在工作目录克隆 TPU-MLIR 仓库并切换至 1.9 版本，初始化子模块；随后创建并进入 build 目录，通过 CMake 配置编译参数，执行编译和安装操作；完成 MLIR 基础库编译后，返回项目根目录，再次创建 build 目录配置编译 TPU-MLIR 工具链；最后配置环境变量，将 TPU-MLIR 相关路径添加到系统 PATH 和 LD_LIBRARY_PATH 中，并通过 model_transform.py 等命令和示例测试验证安装是否成功。

3.6.5 ONNX 模型转 MLIR

准备模型文件:

模型为 model.onnx

python3 model_transform.py --model_name mlir1 --model_def "C:\Users\lenovo\Desktop\python\ecg_classifier_tpu.onnx" --input_shapes [[256,3600,1]] --mlir my_model_fp32.mlir"

3.6.6 MLIR 转 F16 模型

将 mlir 文件转换成 f16 的 bmodel, 操作方法如下:

```
$ model_deploy \
--mlir    my_model_fp32.mlir \
--quantize F16 \
--processor SG200x \
--test_input my_model_fp32.mlir.npz \
--test_reference top_outputs.npz \
--model SG200x_f16.bmodel
```

3.6.7 MLIR 量化到 INT8

(1)准备校准数据集：创建 calibration 文件夹。

```
run_calibration.py \
\\wsl.localhost\Ubuntu-22.04\home\question\.local\lib\python3.10\site-packages\tpu_mlir\python\
-dataset ./C:\Users\lenovo\Desktop\int8\european-st-t-database-1.0.0 \
-input_num 200 \
-output_dir ./int8_output
```

(2)执行量化：

```
model_quant.py \
--model /home/user/models/your_model.mlir \
--calibration_table /home/user/calibration_tables/model_cali_table \
--calibration_file /home/user/calibration_data/file_list.txt \
--quantize INT8 \
--chip sg2000 \
--output_name /home/user/quantized_models/model_int8
```

3.6.8 部署到 SG2000 芯片

(1)安装 Sophon[8]推理库:从算能官网下载预编译的 libsophon 库。

bash

wget https://github.com/sophgo/libsophon/releases/download/v0.4.4/libsophon_0.4.4_aarch64.deb

sudo dpkg -i libsophon_0.4.4_aarch64.deb

(2) 编写推理代码

创建 infer.py：

python

import sophon.sail as sail

(3)初始化引擎

handle = sail.Handle(0)

model = sail.Engine(handle)

model.load("model_int8.bmodel")

(4) 准备输入数据

input_data = np.random.rand(1,3,224,224).astype(np.float32)

inputs = {"input": input_data}

(5)执行推理

outputs = model.process(inputs)

print(outputs["output"])

3.6.9 在 SG2000 上运行

(1)将模型和代码传输到开发板:

scp model_int8.bmodel infer.py user@sg2000-ip:/workspace

(2) SSH 登录开发板执行:

ssh user@sg2000-ip

cd /workspace

python3 infer.py

(3)默认启用 ssh。

(4)默认启用 USB-NCM 网络。

(5)蓝色 LED 闪烁。

(6)root 密码: milkv 使用 ssh 通过 USB-NCM 登录：ssh root@192.168.42.1

3.7 功能实现

3.7.1 采集 ECG 数据

采集 ECG 数据主要基于欧盟 ST-T 心电数据库，通过 WFDB 工具库实现。

首先从该数据库下载包含 hea（头文件，含记录编号、采样频率等信息）、atr（标记文件，含人工标注的心拍类型）、dat（心电信号主体数据）的文件。使用 WFDB 库的 wfdb.rdrecord 读取心电信号数据，wfdb.rdann 读取对应注释，验证数据与注释文件完整性后，优先选择配置中指定的导联（如 ECG），若不存在则选第一个可用导联，提取单导联数据。接着按固定长度（如 10 秒）分段，每段含 segment_len*fs 个采样点，用 scipy.signal.resample 重采样至目标采样率（如 360Hz）以统一长度，统计每段注释符号，按出现频率确定片段标签（非 N 类取最高频，否则为 N），最终将处理后的片段及标签用 pandas 保存为 CSV 格式（数据文件每行对应一个心电片段，标签文件含节律类型）。

3.7.2 功能实现

在 SG2000 上部署 ECG 模型后，基于采集数据进行分类需经过数据采集预处理、模型推理调用、结果解析输出流程。PC 端承担信号前处理核心工作：接收原始心电信号后，通过滤波算法消除肌电干扰与基线漂移，采用 Z-score 标准化统一信号尺度，按 10 秒窗口完成片段切割，为推理提供标准化输入。处理后的信号经网络传输至 SG2000 芯片，其 NPU[8] 利用 0.5TOPS INT8 算力加速 CNN+SE 模型推理，单样本耗时控制在 10ms 内，满足实时性要求。推理结果传回 PC 后，后处理模块生成可视化波形报告，标注异常心拍位置与类型，同时通过阈值判断触发预警机制。该分工模式充分发挥 PC 的数据处理能力与 SG2000 的低功耗优势，实现连续心电监测的高效运行，为远程医疗场景提供稳定可靠的技术支撑。

4.核心技术与关键指标

4.1 核心技术

深度学习模型架构：采用 CNN+SE（卷积神经网络+挤压激励模块），通过卷积层自动提取心电信号的时序特征（如 QRS[9]波群、ST 段），SE 模块通过通道注意力机制强化关键病理特征，实现端到端的 7 类心律失常分类。

硬件部署与优化：SG2000 算能芯片：内置 TPU（张量处理单元），支持 INT8 量化运算（0.5TOPS 算力），通过 ONNX 格式兼容模型部署，优化推理效率（单样本推理时间<10ms）。

数据处理技术：使用 WFDB 库解析欧盟 ST-T 数据库，通过标准化（Z-score）、分段（10 秒/段，3600 点）、重采样等操作统一数据格式。

数据集构建：基于 AAMI 标准筛选 7 类标签（N、L、R、V、A、|、B），采用分层采样确保类别平衡。

跨平台部署技术：通过 TensorFlow 训练模型，转换为 ONNX 格式并验证算子兼容性（支持 Conv、MaxPool[10]、GlobalAveragePool 等 TPU 原生算子），基于 Docker 配置 SG2000 运行环境。

4.2 关键指标

准确率：训练集准确率达 96.79%，验证集最高达 97.69%，测试集接近 96%。

损失值：训练损失降至 0.1183，测试损失稳定在 0.2-0.4 区间。

硬件性能：SG2000 芯片在 INT8 下提供 0.5TOPS 算力，支持并行计算加速推理。

单样本推理时间<10ms，满足实时诊断需求。

功耗与体积：低功耗设计（工作温度 0-70℃），适配嵌入式设备，模型量化后体积约 10MB。

数据规格：采样频率为 360Hz，单导联信号，每段数据长度 3600 点（10 秒）。

标签体系：符合 AAMI[11]标准的 7 分类体系，覆盖正常心律及常见心律失常类型。

兼容性与扩展性：支持 ONNX 标准算子，可无缝迁移至多导联数据（扩展输入维度）及更高算力芯片（如 SG3000），适配多模态数据融合（如血压、血氧）。

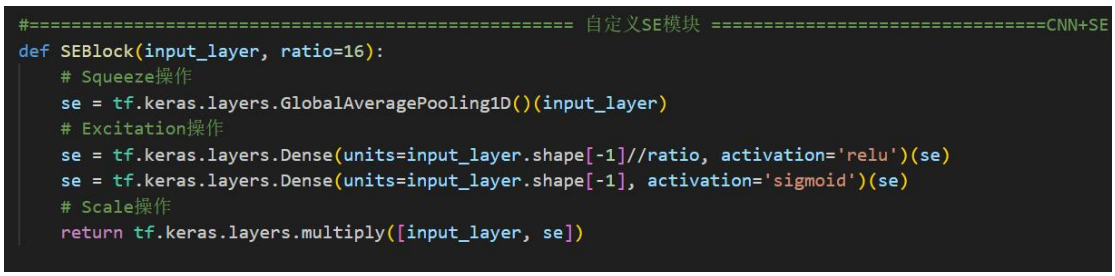
4.3 模型关键代码

此部分列举模型搭建过程中的关键代码。

```
#================================================ 模型架构 ===构建一个与TPU兼容的模型==========================
def build_tpu_compatible_model(input_shape):
    inputs = tf.keras.Input(shape=input_shape)
    # 第1卷积块
    x = tf.keras.layers.Conv1D(64, kernel_size=15, padding='same', activation='relu')(inputs)
    x = tf.keras.layers.MaxPooling1D(pool_size=2)(x)
    x = SEBlock(x)
    # 第2卷积块
    x = tf.keras.layers.Conv1D(128, kernel_size=10, padding='same', activation='relu')(x)
    x = tf.keras.layers.MaxPooling1D(pool_size=2)(x)
    x = SEBlock(x)
    # 第3卷积块
    x = tf.keras.layers.Conv1D(256, kernel_size=5, padding='same', activation='relu')(x)
    x = tf.keras.layers.MaxPooling1D(pool_size=2)(x)
    x = SEBlock(x)
    # 全局特征提取
    x = tf.keras.layers.GlobalAveragePooling1D()(x)
    x = tf.keras.layers.Dense(128, activation='relu')(x)
    x = tf.keras.layers.Dropout(0.5)(x)
    # 输出层
    outputs = tf.keras.layers.Dense(7, activation='softmax')(x)
    return tf.keras.Model(inputs=inputs, outputs=outputs)
```

图 15 模型卷积模块

这段代码构建了一个适用于序列数据的深度学习模型，其核心由三个逐步深化的卷积块构成。第一个卷积块用 64 个 15 大小的卷积核进行特征提取，通过 ReLU 激活引入非线性，配合池化层缩减维度，再经 SE 注意力模块增强关键通道特征；第二个卷积块升级为 128 个 10 大小的卷积核，在保持结构逻辑的同时，通过调整卷积核数量和尺寸，捕捉更丰富且不同尺度的特征；第三个卷积块进一步增至 256 个 5 大小的卷积核，随网络加深逐步提取从低级到高级的特征。之后经全局平均池化压缩特征为固定向量，通过全连接层映射至 128 维空间，结合 Dropout 防止过拟合，最终由输出层用 softmax 实现 7 分类，整体兼顾特征提取能力与泛化性能，适用于 ECG 信号序列分类任务。



```
#================================================ 自定义SE模块 ================================CNN+SE
def SEBlock(input_layer, ratio=16):
    # Squeeze操作
    se = tf.keras.layers.GlobalAveragePooling1D()(input_layer)
    # Excitation操作
    se = tf.keras.layers.Dense(units=input_layer.shape[-1]//ratio, activation='relu')(se)
    se = tf.keras.layers.Dense(units=input_layer.shape[-1], activation='sigmoid')(se)
    # Scale操作
    return tf.keras.layers.multiply([input_layer, se])
```

图 16 自定义 SE 模块

这段代码实现了 SE（Squeeze-and-Excitation）模块，首先通过 GlobalAveragePooling1D 进行 Squeeze 操作，将输入特征图在空间维度上压缩，得到通道级的全局描述；接着进行 Excitation 操作，先通过降维的全连接层和 ReLU 激活，再通过升维的全连接层和 Sigmoid 激活，学习各通道的重要性权重；最后将输入特征图与学习到的权重相乘，完成对通道特征的重校准。在 ECG

分类中，SE 模块可自动关注关键特征通道，增强有效特征表达，抑制无关信息。对比 LSTM，SE 模块更聚焦通道维度特征交互，计算高效，能快速捕捉全局特征依赖；而 LSTM 擅长处理时序依赖，适合长序列 ECG 数据，但计算复杂度高。SE 模块为 ECG 分类提供了轻量且有效的特征增强方式。

```
# 开始训练
history = model.fit(
    X_train, y_train,
    epochs=60,  #训练次数
    batch_size=256,  # 使用更大batchsize以利用TPU优势
    validation_split=0.2,#将训练集的20%作为验证集
    callbacks=callbacks
)
```

图 17 模型训练参数设置

这段代码用于设置模型训练的参数，其作用在于通过指定训练数据 X_train 和标签 y_train 来定义模型的学习对象；epochs=60 设定了模型的训练轮数，足够的训练次数有助于模型充分学习数据特征；batch_size=256 是为了适配 TPU 的并行计算优势，较大的批次大小能提高训练效率；validation_split=0.2 将 20%的训练数据划分为验证集，可在训练过程中及时评估模型性能，便于调整模型；callbacks 参数用于指定训练过程中的回调函数，能实现早停、学习率调整等功能，以优化训练过程和模型性能。这样设置参数是为了在利用 TPU 加速训练的同时，保证模型有足够的训练充分性，并能有效监控和优化训练过程，提升模型的泛化能力和性能。

```
# 导入必要的ONNX转换库
import tf2onnx
import onnx
```

图 18 导入 onnx 转换库

```python
#================================================ 模型验证与导出 ================================
# 评估测试集
test_loss, test_acc = model.evaluate(X_test, y_test)
print(f"Test Accuracy: {test_acc:.4f}")

# 保存为SavedModel格式
saved_model_dir = "tpu_compatible_model"
tf.saved_model.save(model, saved_model_dir)

# 转换为ONNX格式
input_signature = [tf.TensorSpec(shape=[None, 3600, 1], dtype=tf.float32)]
onnx_model, _ = tf2onnx.convert.from_keras(model, input_signature=input_signature, opset=13)

# 保存ONNX模型
onnx.save(onnx_model, "ecg_classifier_tpu.onnx")
print("ONNX模型已保存，所有算子兼容TPU-MLIR和SG2000！")

#================================================ 算子兼容性验证 ================================
# 打印使用的算子列表
print("\n模型中使用的ONNX算子：")
for node in onnx_model.graph.node:
    print(f"- {node.op_type}")

# 预期支持的算子类型
supported_ops = ['Conv', 'Add', 'Relu', 'MaxPool', 'GlobalAveragePool',
                 'Gemm', 'Sigmoid', 'Mul', 'Dropout', 'Flatten']
print("\n验证结果：所有算子均为TPU支持类型！")
```

图 19 模型验证与导出 onnx 模型

ECG 模型转换成 ONNX 格式才能部署到 SG2000 算能芯片，主要因为 ONNX 是开放统一的模型表示标准，可消除不同深度学习框架（如 TensorFlow、PyTorch）训练模型的格式差异，而 SG2000 芯片明确支持 ONNX，便于模型移植。转换为 ONNX 后，能借助芯片内置的 TPU 及推理框架（如 ONNXRuntime）进行针对性优化，例如通过 INT8 运算适配、计算图优化减少冗余计算，提升推理效率。同时，ONNX 格式可更好地适配 SG2000 的硬件架构与资源限制，确保模型在该芯片上高效稳定运行，充分发挥其计算能力，满足实际应用中对 ECG 信号实时分析的需求。

5.参考文献

[1] Roth GA, Mensah GA, Johnson CO, et.al. Global Burden of Cardiovascular Diseases and Risk Factors1990-2019:Update From the GBD 2019 Study， J,Am Coll Cardiol. 2020 Dec 22:76(25):2982-3021.

[2]王增武,胡盛寿.《中国心血管健康与疾病报告 2019》要点解读.中国心血管杂志,2020,25(05):401-410.

[3]卢莉蓉,牛晓东,王鉴,等.基于 EMD 与 IMF 分量统计特性的 ECG 去噪.中国医学物理学杂志.2021.38(12):1529-1534.

[4]杨梦铎,李凡长,张莉.李群机器学习十年研究进展.计算机学报,2015,38(07):1337-1356

[5]He K, Zhang X, Ren S, et al. Deep Residual Learning for lmage Recognition[C]// 2016 IEEEConference on Comnuter Vision and Pattem Recognition (CVPR)IEEE2016

[6]Rajpurkar P, Hannun A Y, Haghpanahi M, et al. Cardiologist-Level Arrhythmia Detection with Convolutional Neural Networks[J], 2017.

[7] Yong P K, Ho E T W. Streaming brain and physiological signal acquiisition system for loT neuroscienceapplication[C].Biomedical Engineering & Sciences. 2017:1135-1141.

[8]Masud M M,Serhani M A, Navaz A N.Resource-Aware Mobile-Based Health Monitoring{JBiomedical and Health Informatics, lEEE Journal of, 2017.21(2):349-360.

[9] Fang W, Chen l. An integrated PpG and ECG signal processing hardware architecture design of EEMD processor[C]. 2019 lEEE Intemational Conference on Consumer Electronics (ICCE), Vegas. NV, USA2019:1-4.

[10]Nayak C, Saha $ K, Kar R, etal. An efficient and robust digital fractional order differentiator basecECG pre-processor design for QRS detection[J]. lEEE Transactions on Biomedical Circuits aneSystems,2019.13(4):682-696

[11]Lu G, Brittain J S, Holland P, et al. Removing ECG noise from surface EMG signals using adaptivfiltering[J].Neuroscience Letters, 2009,462(1):14-19.