**一、项目背景：**

随着物联网（IoT）、智能设备和人工智能（AI）技术的快速发展，边缘计算成为了现代技术架构中的一个重要组成部分。边缘设备通常指的是那些接近数据源端、具备一定计算能力的终端设备，如智能摄像头、无人机、机器人、智能穿戴设备等。这些设备通过内置的传感器和处理单元能够实时捕获并处理数据，从而实现对环境的感知和智能决策。

图像识别技术作为计算机视觉领域的重要应用，已经在多个行业中得到了广泛的应用，如安防监控、智能交通、医疗影像分析、工业自动化等。然而，传统的图像识别系统大多依赖于云端处理，要求将大量图像数据传输至云端进行计算分析，这不仅带来巨大的网络带宽和延迟负担，也存在隐私安全等风险。

因此，如何将图像识别算法有效地部署到边缘设备上，成为了当前计算机视觉领域的一个重要研究课题。边缘设备具有本地计算和存储能力，能够在数据采集端进行初步处理和实时分析，减少对网络带宽的依赖，同时降低数据传输的延迟，提升实时性。与此同时，由于边缘设备的计算资源和功耗限制，如何在有限的硬件环境下实现高效、准确的图像识别，仍然是一个具有挑战性的技术难题。

本项目旨在探索并解决在边缘设备上部署图像识别算法时面临的挑战，设计高效的模型和优化策略，以实现在有限的面积上部署高速及高精度的图像识别解决方案，推动边缘计算在智能感知领域的应用与发展。

**1.当前边缘设备图像识别方案的问题分析**

随着边缘计算和人工智能的快速发展，边缘设备上的图像识别应用逐渐成为热点。然而，当前边缘设备上的图像识别方案在实际应用中仍面临一些显著的挑战和局限性。

1. **面积和硬件资源限制**  
   边缘设备通常具有较小的体积和有限的计算资源，尤其是在低功耗设备中，这使得其在执行复杂的图像识别任务时面临很大的挑战。大多数现有的图像识别模型，如卷积神经网络（CNN）等，通常需要较大的存储空间和强大的计算能力，而边缘设备的硬件资源（如处理器、内存、存储等）通常无法满足这些需求。为了在边缘设备上高效运行这些模型，必须对模型进行压缩和优化，但这种优化可能会牺牲模型的准确性或计算效率。
2. **高速和高准确率的要求**  
   边缘设备上的图像识别任务常常要求在实时性和准确性之间找到平衡。例如，在自动驾驶、安防监控等场景中，识别任务必须在极短的时间内做出决策，以确保系统的响应速度和安全性。然而，传统的图像识别模型往往需要较长的计算时间和高昂的计算资源消耗，导致无法满足边缘设备对低延迟和高实时性的要求。此外，边缘设备的计算能力有限，这使得在保证准确率的前提下，模型往往会受到计算速度的限制。
3. **功耗和计算效率**  
   由于边缘设备往往依赖于电池供电，功耗控制是设计高效图像识别系统时必须考虑的另一个重要因素。过高的功耗不仅会影响设备的续航时间，还可能导致设备过热、性能下降等问题。因此，如何在保证高精度的同时，减少功耗，提升计算效率，是边缘设备图像识别方案中的关键问题。
4. **算法优化的挑战**  
   为了使图像识别模型适应边缘设备，常见的优化方法包括模型剪枝、量化、知识蒸馏等。然而，这些方法虽然可以有效地减小模型的体积和计算复杂度，但可能会导致精度下降。如何在减小模型大小、提高计算效率和保持较高精度之间找到合适的平衡，是当前研究的一个难点。

**二、项目简介**

本项目将以手写体数字识别任务为基础，探索并提出一套针对边缘设备的高效图像识别解决方案，具体目标包括以下几点：

1. **轻量级模型设计**  
   通过设计和选择适合边缘设备的轻量级深度学习模型（如BNN等），减少模型的参数量和计算量，从而降低边缘设备的硬件需求。这些轻量级模型在保证图像识别精度的基础上，能够大幅度减少计算资源的消耗，提高处理速度。
2. **模型压缩与优化**  
   结合模型剪枝、量化和量化损失感知策略等技术，进一步优化图像识别模型，减小模型的体积和计算量。这些优化技术能够在不显著降低识别精度的情况下，显著提升边缘设备的计算效率和响应速度，从而满足实时性要求。
3. **低功耗优化策略**  
   在模型设计和实现过程中，重点考虑功耗控制问题。通过减少不必要的计算操作、使用低功耗的硬件架构（如门控时钟等），以及采用节能算法优化，尽可能降低系统的功耗，使得边缘设备能够在长时间内稳定运行。
4. **定制的网络结构与硬件契合**  
   针对边缘设备的硬件架构，利用硬件和软件协同优化的策略，以最大限度地发挥硬件的计算能力，确保高效的实时处理。进一步提升图像识别任务的处理速度和效率。
5. **手写体数字识别任务验证**  
   本项目将选择手写体数字识别作为具体任务进行验证。手写体数字识别在很多实际应用场景中具有重要意义，如银行票据处理、自动表单识别等。通过对手写体数字识别任务的实现，探索不同优化方案对识别精度、计算速度和功耗的影响，最终提出一个能够在边缘设备上高效执行的解决方案。

本项目将通过上述优化策略，针对当前边缘设备图像识别方案的瓶颈问题，提出有效的解决方案，以实现高精度、低延迟、低功耗的图像识别功能，推动边缘设备在智能感知领域的应用发展。

**三、软件算法设计:**

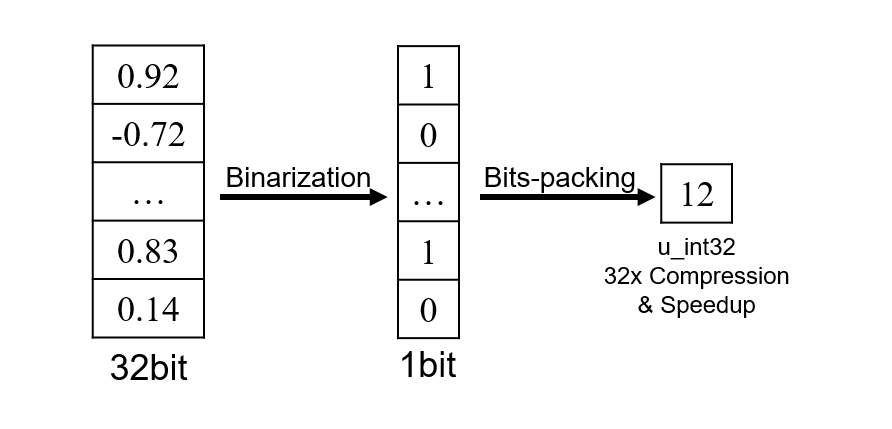
1. **轻量化网络选取**

**采用BNN而非CNN进行轻量化图像识别的方案**

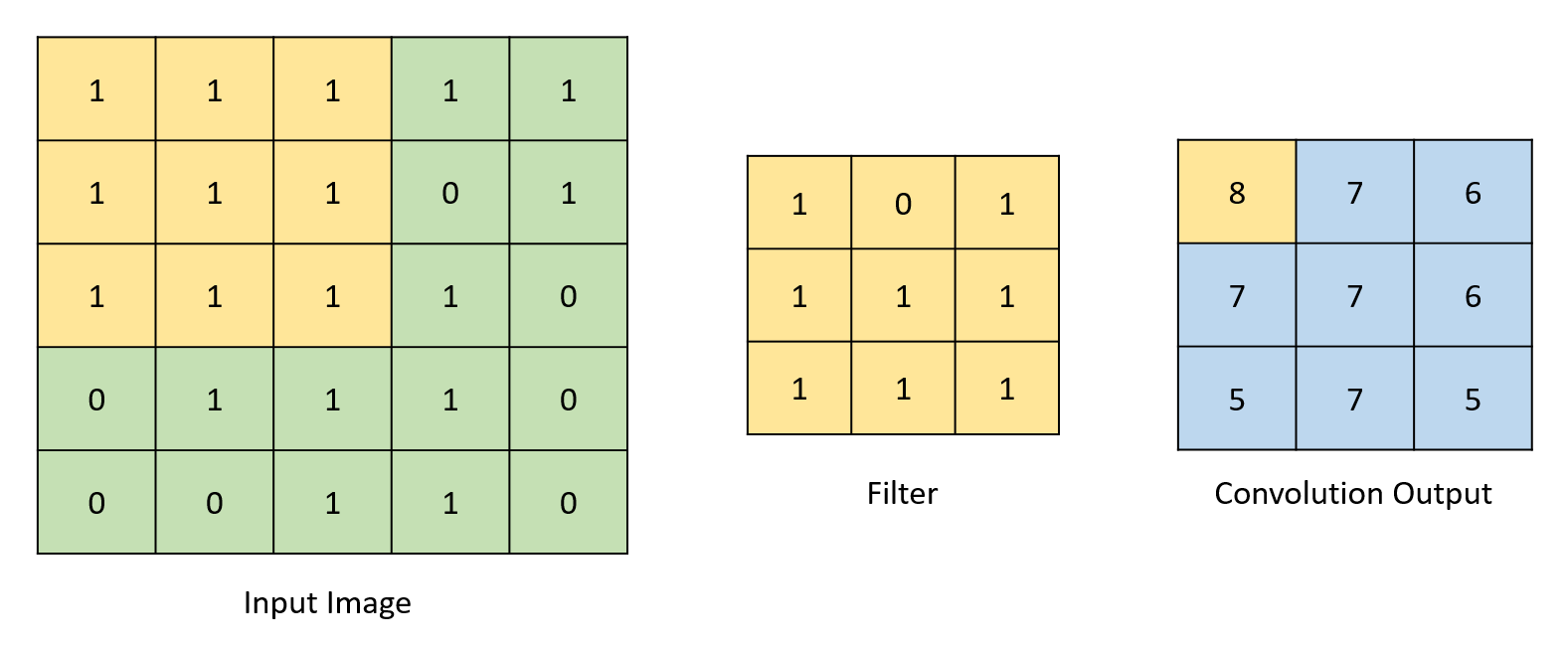
在边缘设备上进行图像识别时，计算资源和功耗是必须考虑的关键因素。传统的卷积神经网络（CNN）虽然在图像识别任务中取得了显著的成功，但由于其较大的计算复杂度和存储需求，难以适应边缘设备的资源限制。为了实现高效的图像识别，二值神经网络（Binary Neural Networks，BNN）作为一种轻量化的网络结构，近年来引起了广泛关注。BNN通过将神经网络的权重二值化，大幅度减少了计算资源的需求，从而提供了一种适用于边缘设备的高效解决方案。

1. **BNN的基本概念**

BNN是一种将神经网络中的权重和激活函数限制为二值（±1）而不是实数值的神经网络模型。与传统CNN不同，BNN的核心思想是在前向传播和反向传播过程中，所有的权重都被强制转换为二值形式。这意味着在计算过程中，不再进行浮点数运算，而是通过二值乘法和加法操作来代替常规的矩阵运算。由于二值计算在硬件上的实现更加高效，BNN可以显著减少计算复杂度并降低存储需求。



**图1. 全精度权重的二值化与压缩操作**



**图2. 二值化卷积核卷积操作**

1. **BNN与CNN的对比**
2. **计算复杂度**  
   在传统的CNN中，每个卷积操作都涉及到浮点数的乘法和加法，这对于边缘设备来说是一个沉重的负担，尤其是在实时性要求较高的应用场景中。相比之下，BNN将权重二值化后，计算过程主要依赖于二值乘法（XOR操作）和加法，极大地降低了计算量。二值化操作不仅简化了矩阵运算，还可以通过硬件优化（如专用的二值计算加速器）进一步加速计算速度。
3. **存储需求**  
   CNN的模型通常需要大量的存储空间来存储高精度的权重和激活值，尤其是在大规模数据集上训练时，存储需求会呈指数级增长。而BNN通过将每个权重限制为一个比特（0或1），显著减少了存储空间的需求。BNN的权重矩阵可以用二进制格式存储，相比传统FP32的CNN节省了大约31倍的存储空间，使得模型能够在边缘设备上运行。
4. **精度与性能**  
   虽然BNN的二值化操作会对精度造成一定影响，但通过合适的网络结构设计和优化策略，BNN可以在许多应用中保持与CNN相当的识别精度。例如，通过使用较深的网络结构和适当的训练技巧（如量化损失感知策略、scaling策略和剪枝等），BNN能够在精度损失和计算效率之间取得良好的平衡。事实上，针对手写体数字识别等任务，BNN已经在一些公开数据集（如MNIST、CIFAR-10）上取得了与CNN相近的性能。

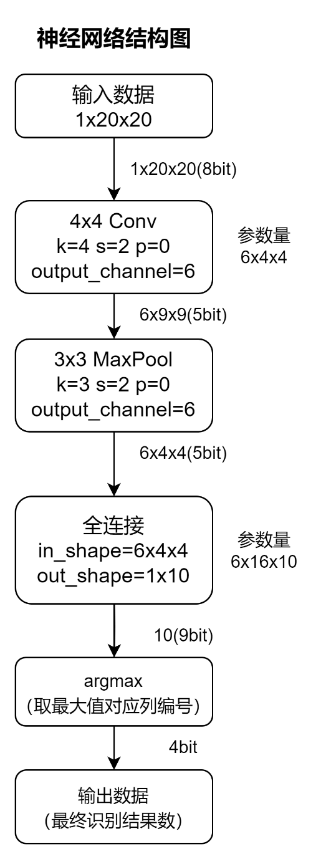
**总结**

采用BNN进行轻量化网络设计，为边缘设备上的图像识别提供了一种高效、低功耗的解决方案。通过二值化计算，BNN不仅减少了模型的存储需求和计算复杂度，还能在保持较高精度的同时，实现快速、实时的图像识别处理。通过在手写体数字识别任务中的应用，本项目将验证BNN在边缘设备上的可行性，推动边缘计算技术在智能感知领域的应用发展。

1. **定制化网络结构设计**

对MNIST数据集图片进行中心裁剪得到输入数据为20\*20的图像矩阵，经过卷积核（4\*4\*6通道，s=2）得到卷积层输出9\*9\*6矩阵，再经过池化层（3\*3，s=2）得到池化层输出4\*4\*6矩阵，然后通过全连接层（4\*4\*6\*10）得到全连接层输出1\*10，最后通过argmax输出预测的0-9其中的一个数字。通过定制的网络结构可以发现全连接核与卷积核大小相同，因此可以通过复用卷积核进行全连接层的计算来减小面积开支。

1. **网络结构中可选择的超参量和选择时的考虑因素网络结构中**

网络结构可选的超参量：

1. **输入数据压缩程度**
2. **卷积层、池化层、全连接层的大小**
3. **通道数**

选择的综合考虑因素：

1. **面积限制**
2. **预测准确率要求**
3. **与硬件适配性（e.g. 卷积层与全连接层复用）**
4. **各参量计算过程**

设x为输入数据的大小，a为复用的卷积核与全连接层尺寸，b为池化层尺寸，根据卷积的操作规则可以得出以下公式：

其中要求x、a、b为整数且必须处在一定合理的区间，通过适当迭代后我们选择x=20，a=4，b=3。卷积核的通道数越大预测准确率越高，但是过大的并行卷积核会产生无法承受的面积代价，已通过python验证选择通道为3、6、10、12时预测准确率大概为85%、91%、92%、93%，最终综合考虑面积、准确率与速度的要求后选择通道数为6。

1. **训练问题与解决方案**

**(1) 二值化前向函数不可导**由于二值化操作使前向函数中加入了一个sign函数，sign函数具有不可导性，这会在训练反向传播时产生一些问题，最直接的解决方法就是使用STE (Straight-Through-Estimator), 但是有时简单的STE并不能达到期望的精度，于是本次实验我们采用了Sigmoid函数来近似sign函数从而进行反向传播训练。

**(2) 量化与精度损失**

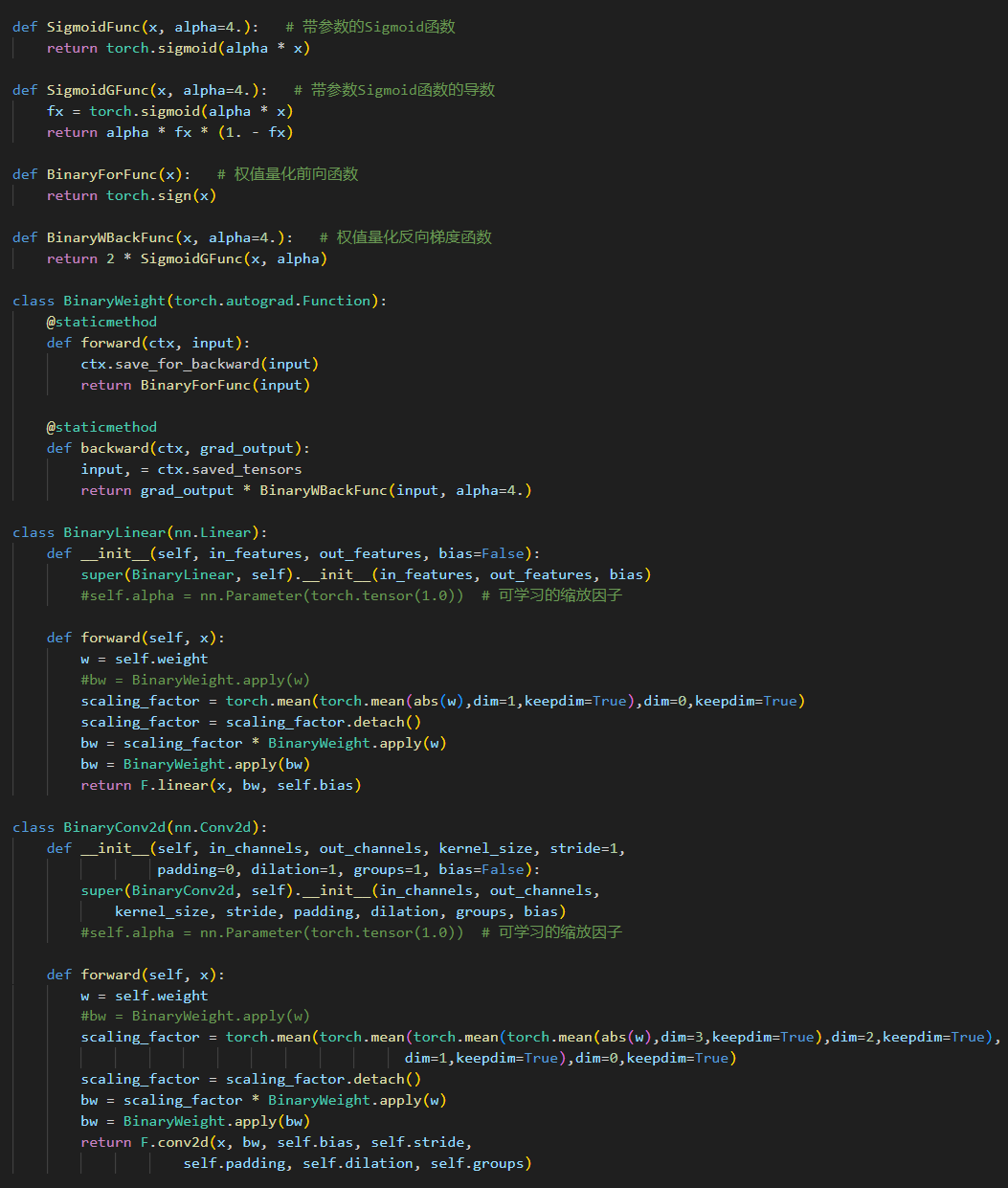
由于全连接层巨大的参数量导致暂存数据和权重数据的存储压力显著增加，为了减小存储压力，本工作采用一种量化策略减小暂存数据的位宽。通过打印暂存值发现大部分数据在高位宽出现数据冗余的现象，通过有选择的截取位宽可以减小存储压力的同时控制精度的小幅度下降。由于本工作二值化量化和数据位宽量化的量化操作导致预测准确率相较于全精度网络下降，为了减小精度的损失，在训练时引入这种量化损失帮助网络能应对量化误差，即量化损失感知策略。具体操作为在网络训练的代价函数中加入前馈时量化误差，将减小量化误差也作为网络训练的目标。同时为了使梯度回传时能与原梯度处于近似相等的梯度空间中，我们引入了一个scaling策略来缩放回传的梯度值

1. **模拟硬件的推理过程**

实际的Pytorch环境只能支持全精度的浮点数运算，为了能更加模拟实际的电路硬件的推理过程，我们定义了一个watch\_net可以实时打印任何一个数据在任何一个层的中间激活值，主要是使用软件模拟二值化乘加的补码运算与位宽截取的操作

* 1. **代码附录**

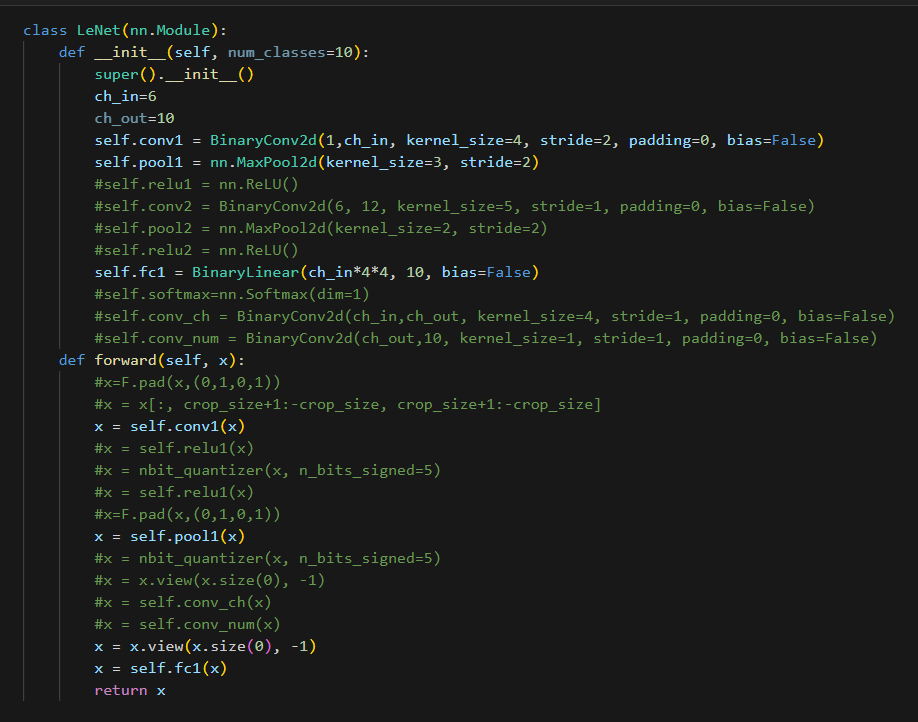
1. **训练需要的函数与类(class)定义**

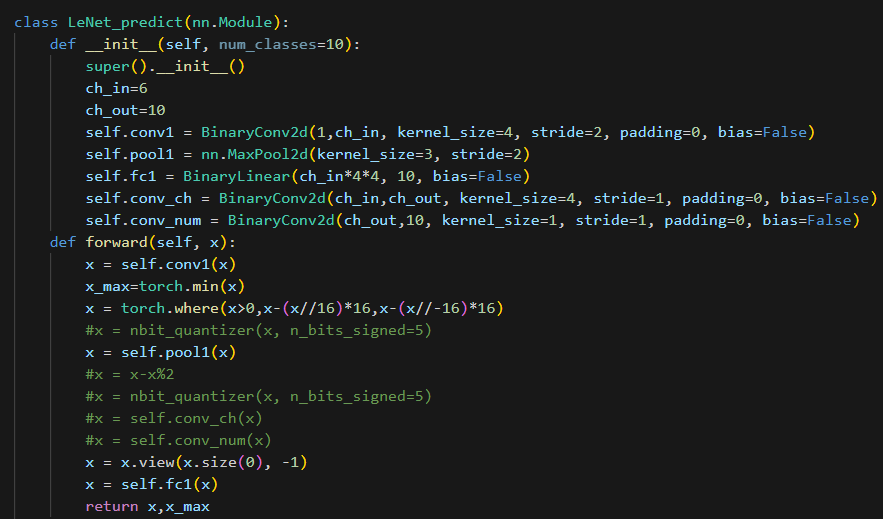


1. **数据集加载过程**

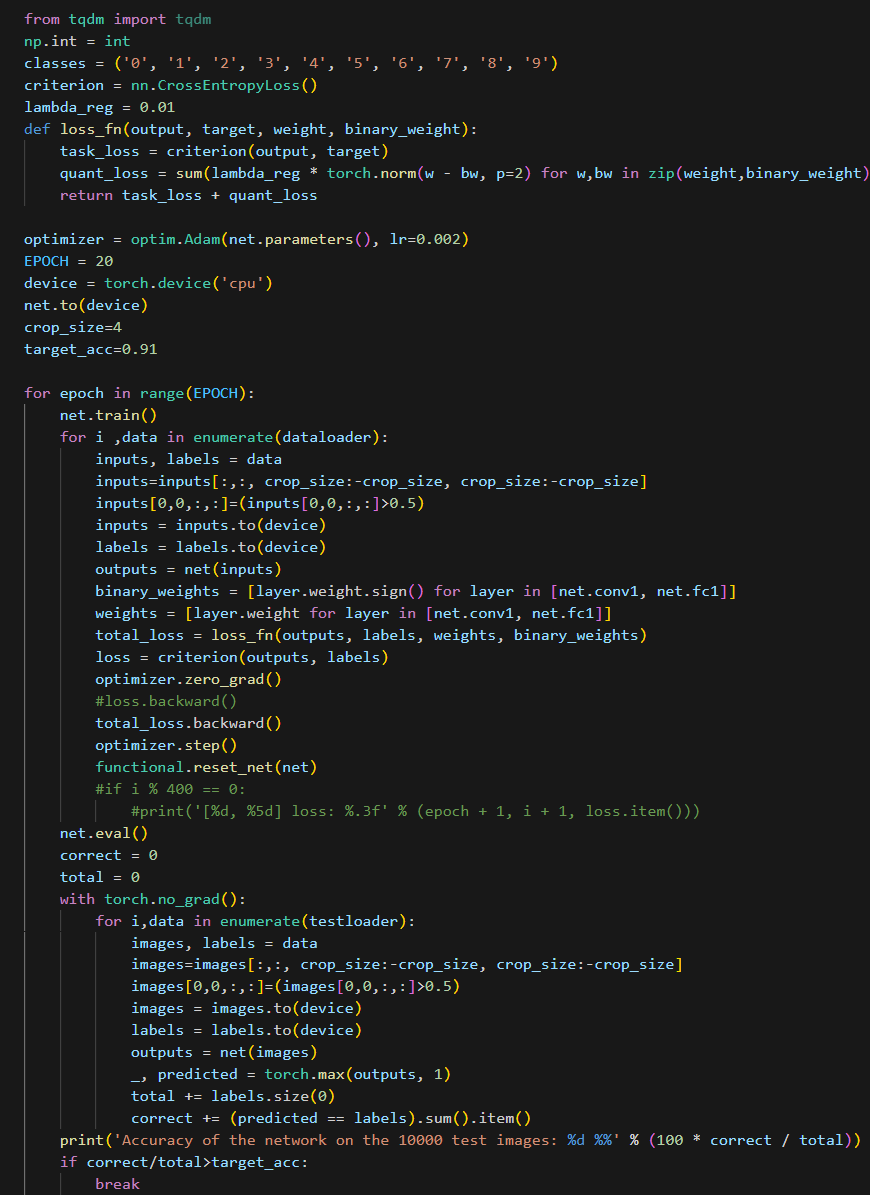


1. **训练与预测网络类的定义**

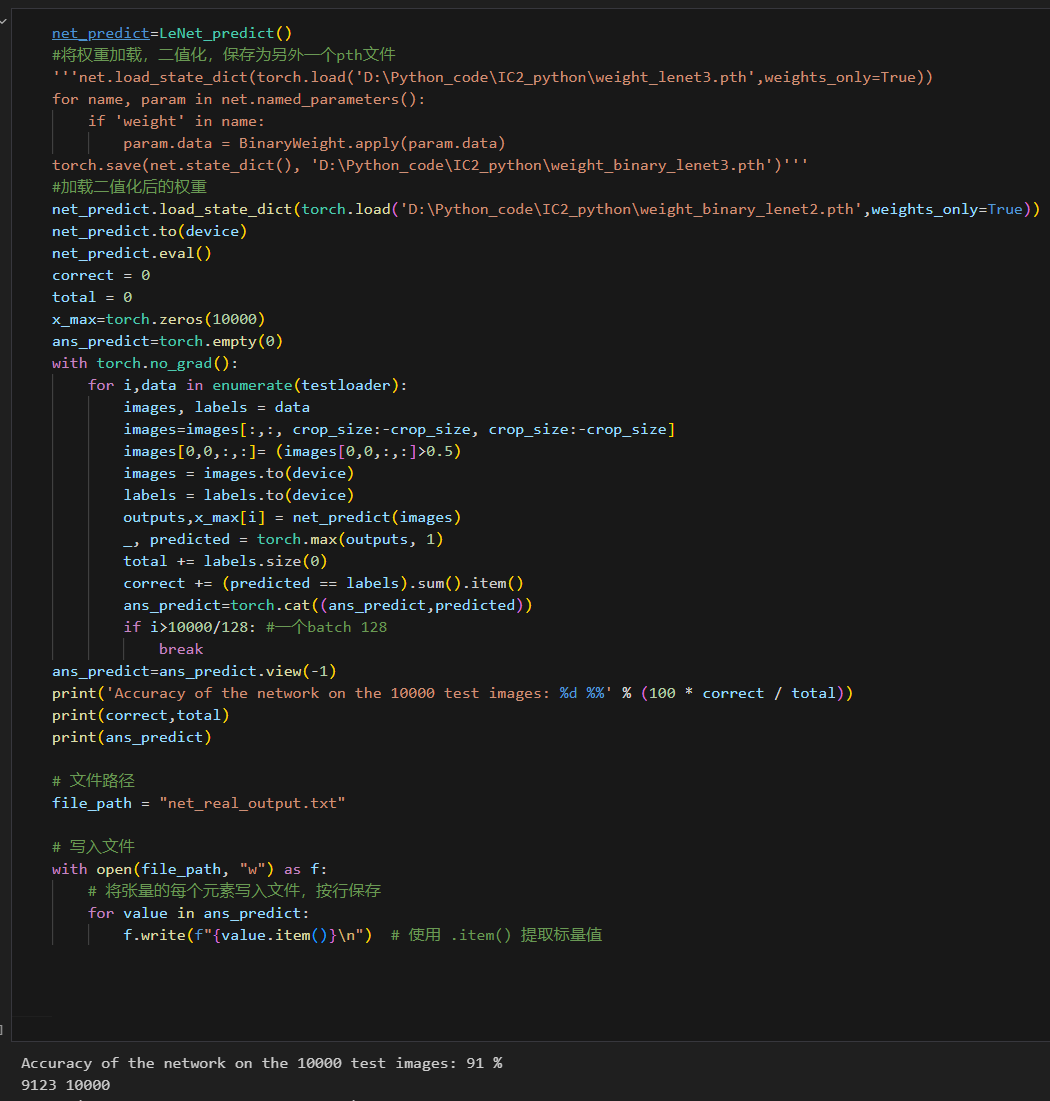




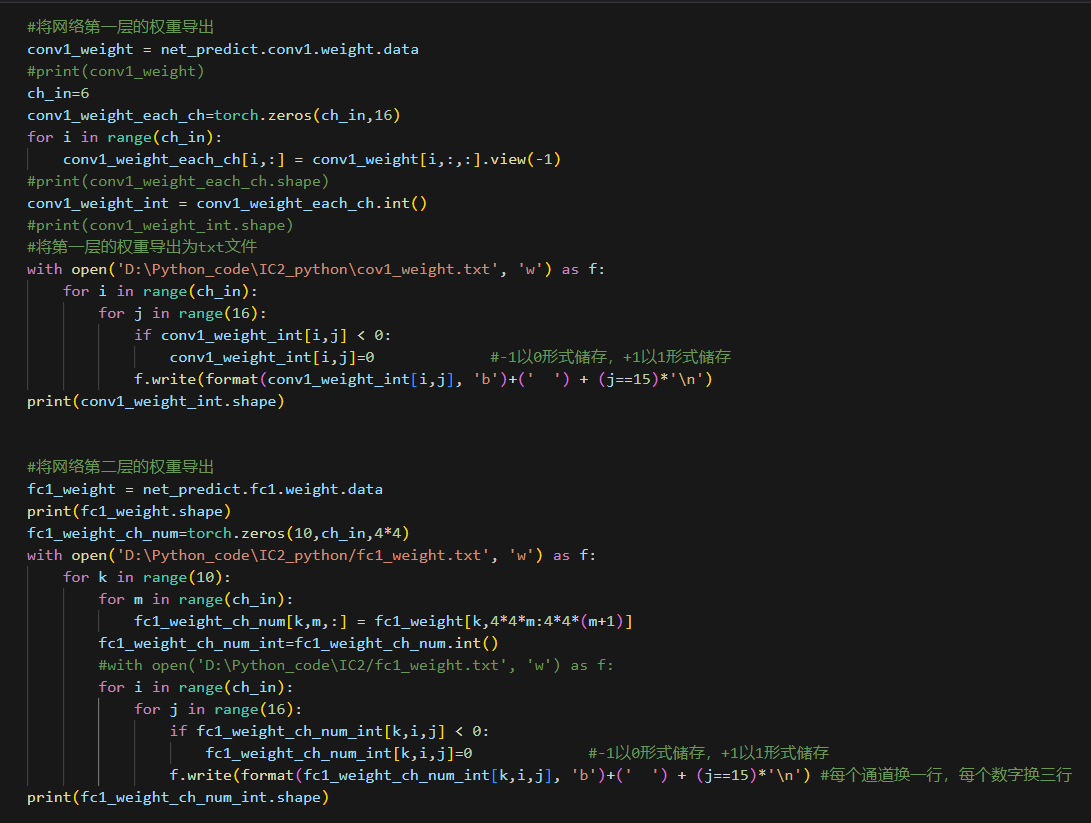
1. **训练过程**



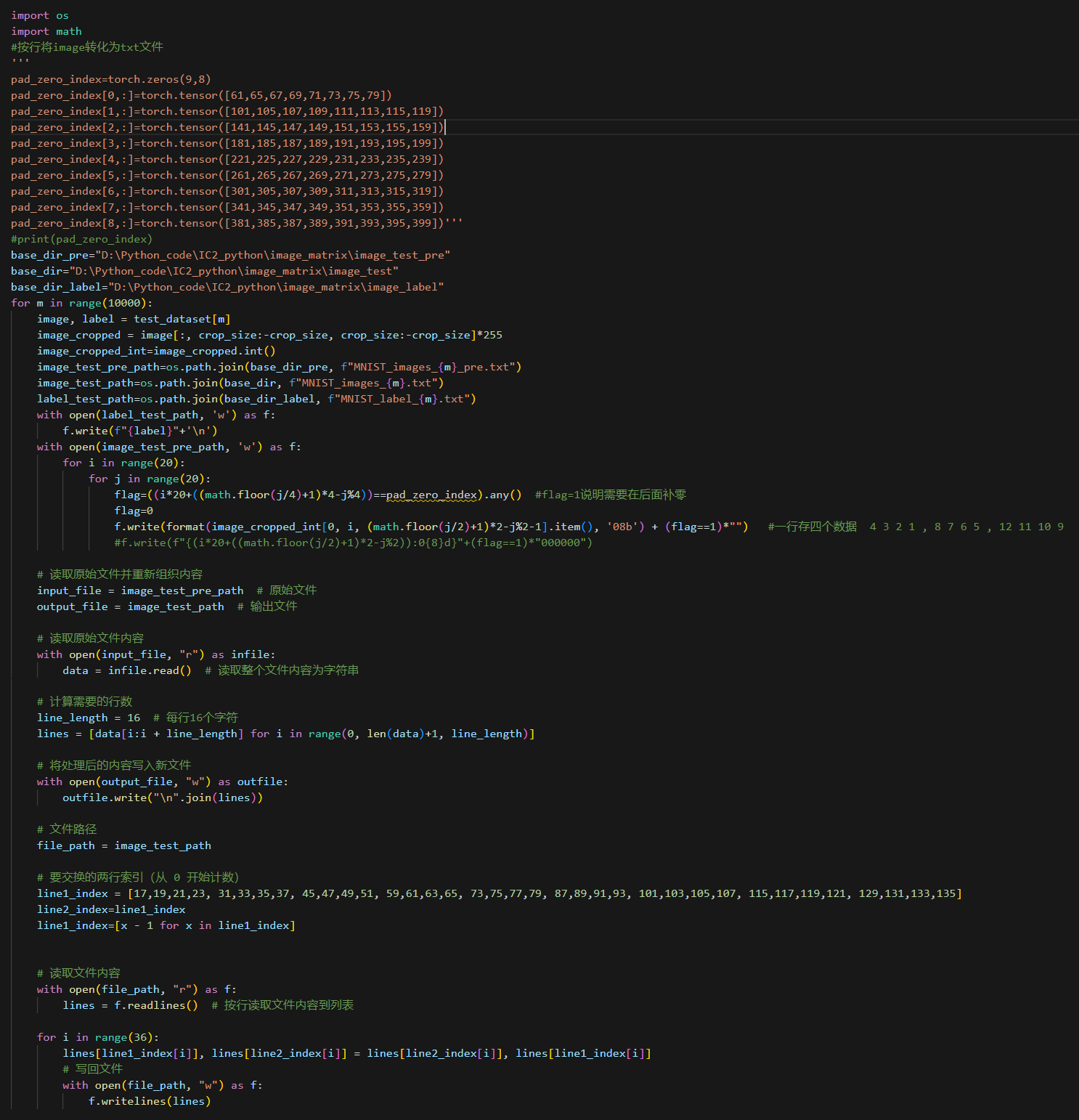
1. **推理预测过程**



1. **网络权重导出过程**



1. **测试集数据导出过程**



1. **Debug使用的watch\_net定义**

