

基于深度学习 OFDM 信道补偿技术硬件实现

刘仲谦¹, 丁丹², 薛乃阳³

(1. 航天工程大学 研究生院, 北京 101416; 2. 航天工程大学 电子与光学工程系, 北京 101416;
3. 中国人民解放军 63920 部队, 北京 101416)

摘要: 为了解决部分高性能深度学习神经网络因存在复杂度高及计算量大等缺陷在嵌入式设备中应用效果不理想的问题; 以小型化集成智能无线电设备 AIR-T 为平台实现了基于深度学习的 OFDM 信道补偿技术; 在 FPGA 芯片上不仅实现了 OFDM 信号传输系统模块, 也实现了传统信道估计与均衡模块, 模块对数据进行预处理减轻神经网络工作量以完成神经网络信道补偿技术模块在 Jetson TX2 平台 GPU 上的高效实现; 由实验记录神经网络训练过程中的计算复杂度和参数拟合速度得知, 传统信道估计与均衡模块有效降低了网络训练时的运算次数; 由测试性能方面可知, 经过神经网络信道补偿后的数据误码率比之前传统信道估计与均衡后的误码率有明显降低;

关键词: 正交频分复用 (OFDM); 深度学习; 信道补偿; 嵌入式设备; FPGA; Jetson TX2

Hardware Implementation of Deep Learning Based OFDM Channel Compensation Technique

LIU Zhongqian¹, DING Dan², XUE Naiyang³

(1. Department of Graduate Management, Space Engineering University, Beijing 101416, China;

2. Department of Electronic and Optical Engineering, Space Engineering University, Beijing 101416, China;

3. Chinese People's Liberation Army 63920 Unit, Beijing 101416, China)

Abstract: In order to solve the problem that some high-performance deep learning neural networks are not ideal for the application in embedded devices due to the defects of high complexity and large computation. The Deep learning-based Orthogonal Frequency Division Multiplexing (OFDM) channel compensation technology is implemented on the Artificial Intelligence Radio-Transceiver (AIR-T), which is taken as a platform for the miniaturized integrated smart radio device. Not only the OFDM signal transmission system module, but also the conventional channel estimation and equalization module are implemented on the Field Programmable Gate Array (FPGA) chip. The data for these modules is preprocessed to reduce the workload of the neural network and complete the efficient implementation of the neural network channel compensation technology module on the graphics processing unit (GPU) of the Jetson TX2 platform. The computational complexity and parameter fitting speed of the neural network training process are recorded by the experiment, and the conventional channel estimation and equalization module effectively reduces the number of operations during the network training. From the test performances, the data of the Bit Error Ratio (BER) after the neural network channel compensation is significantly lower than that of the BER after the previous conventional channel estimation and equalization.

Keywords: OFDM; deep learning; channel compensation; embedded devices; FPGA; Jetson TX2

0 引言

近年来, 无线通信系统的性能需求大大提高, 给硬件实现带来了巨大挑战。传统无线通信系统设计是基于模型驱动的理念, 其中每个模块的优化都是通过人们所掌握的知识信息即专家知识建立模型推导而来, 这种优化模式在未来越来越复杂的信道环境条件下将变得越发难以实现。在这一背景下, 这些年来快速发展的深度学习 (DL, deep learning) 技术为解决无线通信系统的算法难题带来了新的思路, 同时也为无线通信系统的硬件实现带来了新的设计理念。深度学习技术可以直接从海量数据中学习所需的

隐藏规律, 利用这些规律做出相应的预测或决策^[1]。其数据驱动的特性正好可以解决传统无线通信系统设计中因依赖专家知识推导优化算法而产生的问题。

目前有关深度学习的大量研究是基于计算机、工作站、服务器等大型平台的运行仿真^[2-3]。随着深度学习技术研究的深入, 部分高性能的神经网络模型被应用在嵌入式硬件设备中, 但在应用过程中理论上性能优异的网络模型普遍存在着复杂度高以及计算量大的问题, 以至于很多网络模型应用到嵌入式设备上效果并不理想^[4-5]。

针对嵌入式设备中网络计算过程耗费大量时间和资源

收稿日期: 2022-03-22; 修回日期: 2022-04-15。

作者简介: 刘仲谦(1994-), 男, 河南鹤壁人, 硕士, 主要从事深度学习在测控通信中的硬件实现方向的研究。

通讯作者: 丁丹(1980-), 男, 江苏南京人, 博士, 副研究员, 主要从事航天测控方向的研究。

引用格式: 刘仲谦, 丁丹, 薛乃阳. 基于深度学习 OFDM 信道补偿技术硬件实现[J]. 计算机测量与控制, 2022, 30(6): 150-156, 162.

的问题目前已有的研究中通常集中于 3 个方面: 1) 采用诸如网络裁剪^[6-7]、低比特数据表示^[8]、模型蒸馏^[9]等方法来减少数据量或计算精度, 此种方法虽然减少了计算量, 但同时降低了网络的性能, 使计算的结果精度保持在实际系统可接受范围内; 2) 在云和终端设备上分布式混合部署深度学习神经网络^[10], 此方法可以有效解决复杂度较高和计算量大的问题, 但不适用一些延时要求较高或无网络支持的场景; 3) 使用多种硬件平台组成的异构计算平台等来加速深度学习算法的计算过程^[11-13], 此种方法要求合理的计算资源分配, 根据不同的网络结构分配不同的硬件平台完成实现, 一旦达到较为理想的资源分配, 此种方法可以有效解决神经网络应用在嵌入式设备上效果不理想的问题。

基于以上分析可得目前已有的深度学习相关硬件实现大部分为系统复杂、体积大、成本高的系统, 而基于深度学习无线通信传输系统的实现趋势为小型化、系统简单、成本低的集成终端设备。本文基于此趋势研究了基于深度学习信道补偿技术的 OFDM 信号传输系统的可集成小型化智能无线电设备实现, 完成了 OFDM 信号收发处理、传统信道估计与均衡算法、基于深度学习信道补偿的板卡模级实现, 推动深度学习在无线系统传输中的进一步实用化。本文选择实现 OFDM 信号系统是由于其具有较高的频谱利用率, 能够有效的抵抗多径效应带来的码间干扰和子信道间干扰, 且 OFDM 信号系统较容易实现, 具有极大的实现价值。另外本文选用 FPGA 芯片与 GPU 集成的智能无线电设备作为实现平台, 可以有效分配不同硬件的计算资源给不同算法模块以达到 OFDM 信号传输系统的快速有序实现。综上所述本文以小型化智能无线电设备为平台实现基于深度学习的 OFDM 信道补偿技术, 通过数据预处理减轻神经网络工作量, 完成神经网络在嵌入式硬件设备中的高效实现, 推动深度学习在无线系统传输中的进一步实用化。

1 系统整体方案及硬件选型

本文设计的 OFDM 信号传输系统旨在实现 OFDM 信号发送及接收前提下在接收端运用传统信道估计均衡和深度学习信道补偿技术结合的方法对接收信号完成进一步的信道补偿从而降低收发两端信号数据比对的误码率。其中硬件实现模块包括 OFDM 信号的产生、传输、接收及后续的最小二乘 (LS, least squares) 估计算法、迫零 (ZF, zero forcing) 均衡算法、深度学习全连接神经网络信道补偿模块。系统的设计如图 1 所示。

本文将系统各个模块设计在一个集成的智能无线电设备上, 其中根据计算资源消耗水平的不同, 系统前端 OFDM 信号收发过程及传统信道估计均衡模块所需资源分配较少, 本文将其设计在 FPGA Xilinx Artix-7 XC7A75T 芯片上实现, 该芯片在单个成本优化的 FPGA 中提供了高性能功耗比结构、收发器线速、DSP 处理能力及 AMS 集成。深度学习信道补偿技术模块所需资源分配较多, 本文将其设计在嵌入式平台 NVIDIA JETSON TX2 多核处理

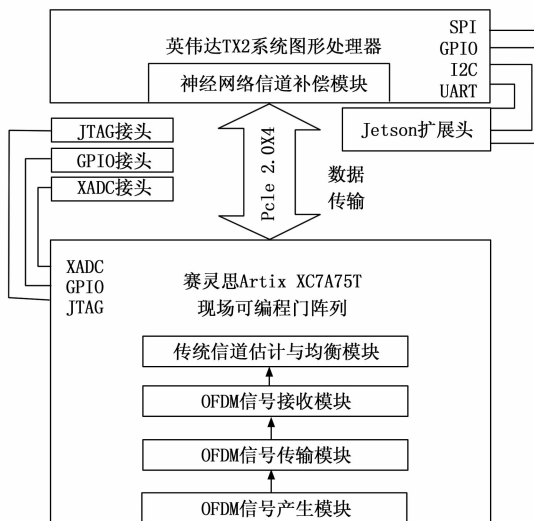


图 1 系统设计框图

器^[14]上实现, 该处理器体积小, 功耗低, 创建了实现高性能并行的计算环境。

2 系统设计

2.1 开发平台简介

根据上文描述硬件实现小型化、低功耗及资源合理分配的趋势, 本文根据 FPGA 芯片和 GPU 处理器的小体积、可集成性设计开发平台。经过市场调研分析各种产品性能, 最终发现 AIR-T 基本满足本文硬件系统实现需求。硬件平台使用 ubuntu18.04 64 位操作系统, ARMv8 架构, 内部构造包括 AD9371 收发器, FPGA Xilinx Artix-7 XC7A75T 芯片和 JETSON TX2 多核处理器。FPGA Xilinx Artix-7 系列芯片具有低功耗, 高性能的特性, 其中 XC7A75T 具有 75 520 个逻辑单元, 3 780 个存储器, 可提供具有 100 MHz 的传输带宽, 具有快速实时计算能力, 符合本文 OFDM 信号传输系统模块设计的实现需求。NVIDIA Jetson TX2 系列模组尺寸比信用卡还小, 可为嵌入式人工智能 (AI, Artificial Intelligence) 计算设备提供出色的速度与能效。其配备 NVIDIA Pascal 架构, 具有 256 个 NVIDIA CUDA 核心, 高达 8 GB 内存、59.7 GB/s 的显存带宽以及各种标准硬件接口, 性能高达 Jetson Nano 的 2.5 倍, Jetson TX1 的 2 倍, 并且功耗低至 7.5 W。Jetson TX2 系列模组非常适用于实时处理需要解决带宽和延迟问题的应用程序, 在实时软件无线电 (SDR, software defined radio) 应用中, 使用 NVIDIA Jetson TX2, 比 Intel 7500U CPU 提高了 250% 的带宽处理, 比 ARM Cortex-A57 (4 核) 提高了 1 350% 的带宽处理。除此之外, 其 GPU 可用于高度并行处理。综合来看, NVIDIA Jetson TX2 在性能上最大程度匹配了本文设计的神经网络信道补偿技术模块。

AIR-T 是一款具有嵌入式高性能计算能力的小型化集成智能无线电设备, 其通过集成 FPGA、GPU、CPU 三个数字处理器完成高性能计算、人工智能和深度学习。该系

统可以用作深度学习算法的高度并行 SDR 处理和深度学习算法的推理引擎^[15]。嵌入式 GPU 支持 SDR 应用程序实时处理大于 200 MHz 的带宽。AIR-T 开发套件支持 Ubuntu16.4 系统, 可以通过 SoapySDR 移植现有的 GNU Radio 应用程序, 也可以使用自定义 GNU Radio 模块部署神经网络或高性能应用程序, 同时, 硬件支持使用 Python 或 C++ 进行编程开发。从开发层面看大大提高了开发效率。总体来说, 本文借助 AIR-T 智能无线电设备实现了一个高性能低功耗的基于深度学习信道补偿技术的 OFDM 信号传输系统。

2.2 OFDM 收发及预处理 FPGA 实现

2.2.1 OFDM 信号收发模块实现

本文实现的 OFDM 信号收发过程主要包括 OFDM 信号数据的产生模块、传输模块和接收模块。其中信号数据的产生过程如图 2 所示。

AIR-T 支持 RFNoC (RF network on chips)^[16] 对 FPGA 进行开发, 可借助 GNU Radio^[17] 创建流图生成 python 脚本完成 FPGA 芯片上模块的实现, 也可使用 python 语言进行自定义模块的开发, 设备的初始化及模块运行框架由 RFNoC 借助 C++ 编译完成, 具体模块参数设置可借助 GNU Radio, 也可以直接通过编写代码指令。

OFDM 信号产生模块的实现过程为首先从 gunradio 库中导入需要调用模块的指令库如 digital, blocks, gr, fft 等。然后通过 blocks 库函数的子函数 random.randint 产生 16384 个随机 0、1 序列, 需要注意的是此处生成的是 Byte 数据, 经过 stream 函数指令将数据转化为 128 位固定长度并添加长度标记, 接着数据经过 repack 指令重新打包, 每 8bits 一组转换为 2bits 为后续 QPSK 映射做准备。QPSK 的映射是借助 digital 库函数指令对 byte 数据进行复数型数据映射, 映射对应结果为 0.707, -0.707, 0.707j, -0.707j, 将映射结果传递给虚拟接收器完成初始数据的生成。此前模块之间的数据传递借助 connect 函数指令完成。RFNoC 借助编译配置可以直接将数据移入和移出 FPGA, 从而在应用程序中无缝使用基于主机和基于 FPGA 的处理, 体现在开发方式上即为 GNU Radio 模块之间的连线和代码指令 connect 函数的使用, 这也为下文实现信号传输模块和

传统信道估计与均衡模块提供了便利。

在已有初始数据条件下然后进行 OFDM 子载波分配, 借用 ofdm_carrier_allocator 指令分配数据子载波及导频子载波, 导频子载波间隔为 1, 数值为 0.707, -0.707, 0.707j, -0.707j 的随机排列, 接着是调用 fft 函数对数据进行快速傅里叶逆变换 (IFFT, inverse fast fourier transform) 形成 OFDM 信号数据, 此处 FPGA 中 FFT 的实现也是通过 RFNoC 编译调用 Xilinx CoreGen IP 完成。最后使用 ofdm_cyclic_prefixer 函数指令给 OFDM 信号添加 16 位循环前缀 (CP, cyclic prefix), 将时域信号传递给虚拟接收器完成 OFDM 信号数据的生成。

信号的传输模块是 AWGN 信道模块的设计, 此处主要借助 blocks 库函数中有关数学运算的模块和 connect 指令, 首先依照信号功率求解公式借助库函数中求和与除法模块运算指令对时域数据进行处理。求解公式为:

$$S = \frac{\sum_{n=0}^n |x_n|^2}{n} \quad (1)$$

x_n 为时域信号虚拟接收器中数据。S 为信号功率模块数据。

然后再借助指数、乘方及相加模块指令完成噪声方差模块和信号叠加噪声模块的实现。噪声方差公式为:

$$S_2 = S \times 10^{(-ebn/10)} \quad (2)$$

ebn0 为可设置数据变量 Eb/N0。S₂ 为噪声方差模块数据。

信号叠加噪声公式为:

$$Y = X + S_2 \times S_n \quad (3)$$

Y 为传输模块虚拟接收器中数据, S_n 为与已知产生信号长度相同的随机序列。

OFDM 信号的接收模块为信号产生模块的反向操作, 接收模块的实现过程是将 OFDM 信号传输模块中的数据作为数据源, 处理模块包括去 CP, FFT 解调, 其开发方式同样采用直接编写 python 代码脚本的方式完成。

2.2.2 传统信道估计与均衡算法模块实现

本文借助 FPGA 芯片开发实现 LS 信道估计及 ZF 均衡模块的方法与上文描述的模块实现方法相同, 需要注意的

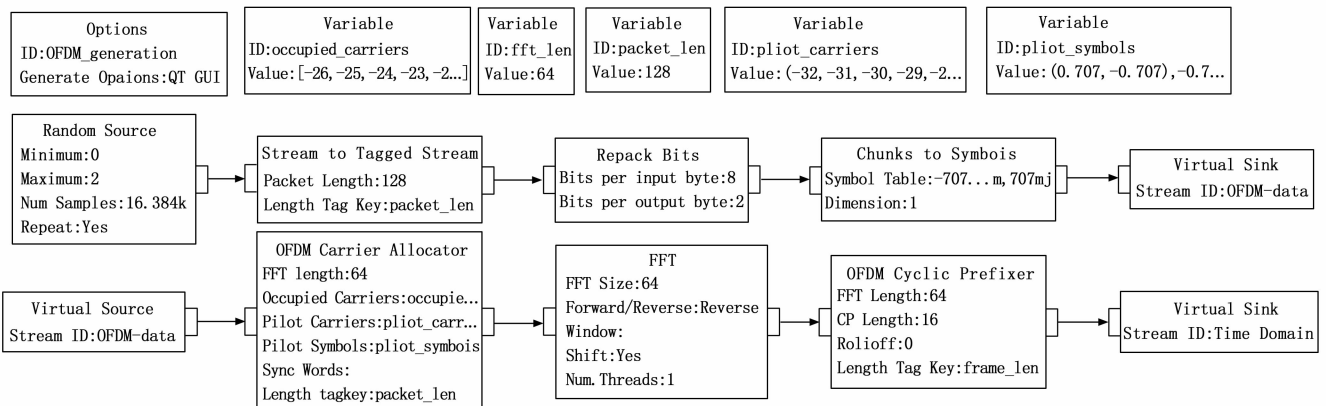


图 2 OFDM 信号产生流图

是 connect 指令连接的模块位置。LS 信道估计公式为:

$$\hat{H}_{LS}(k) = \frac{Y_p(k)}{\hat{X}_p(k)} \quad (4)$$

Y_p 为解调后导频位置处的数据, X_p 为 QPSK 映射后添加的导频数据。 \hat{H}_{LS} 为 LS 信道估计模块数据。

ZF 均衡公式为:

$$\hat{X}_{ZF}(K) = \frac{Y(K)}{\hat{H}_{LS}(K)} \quad (5)$$

其中: Y 为解调后每个 OFDM 符号信号数据, \hat{X}_{ZF} 为 ZF 均衡模块数据, 也是神经网络模块的输入数据。

2.3 深度学习模块 GPU 实现

2.3.1 网络结构的选择

根据上文描述传统信道估计与均衡模块有效减少了神经网络层数和神经元个数, 因此本文在设计神经网络模块时选择层数较少的全连接神经网络结构, 有效控制了神经网络模块的计算次数。全连接神经网络 (FCNN, full connected neural network)^[18] 的网络结构是从输入层到隐含层, 再到输出层, 层与层节点之间是全部连接的, 但是隐含层之间的节点是无连接的。其中该神经网络需要设计的参数包括神经网络层数、神经元个数、激活函数、学习率、优化器、损失函数等。

本文实现的全连接神经网络层数为 2 层, 分别为输入层与输出层, 中间无隐藏层, 而神经元个数与每次运算处理数据的个数有关, 本文一帧数据包括 2 个 64 位的 OFDM 符号, 其实部加虚部的数据位数为 256 位, 所以输入层与输出层的神经元个数都为 256。

激活函数主要是为了给模型加入非线性因素, 让模型拥有更好的表达能力^[19]。激活函数的选择取决于数据的特性, 本文设计的 OFDM 数据经过 QPSK 调制, 具有双极性, 所以激活函数选择双切正切 (Tanh) 函数。

损失函数的本质是根据真实值和预测值的距离来改变模型的收敛方向。常用的损失函数有均方误差 (MSE, mean square error) 和交叉熵^[20]。优化器的选择与模型优化方式有关, 优化的实质是在损失函数和正则化函数确定的前提下, 使权重重新达到最优。优化算法分为一阶算法和二阶算法, 由于二阶导计算成本高所以二阶算法不常用, 一阶算法中最常用的是梯度下降法。学习率就是使梯度下降的步长, 学习率也是每个优化器的重要参数。常用的优化器有小批量梯度下降优化算法 (SGD)、引入一阶动量的梯度下降算法 (SGDM)、梯度平方根算法 (RMSProp)、自适应动量估计算法 (Adam)^[21]。本文对于优化器和损失函数的选取则基于常规考虑采用 RMSProp 优化器和 MSE 损失函数, 初始学习率设置为 0.001 且每训练 500 轮数据将学习率设置为原来的 1/5, 在训练过程中减小学习率不仅可以加快神经网络的拟合速度, 还可以提高网络参数的拟合精度。

2.3.2 全连接神经网络信道补偿技术模块实现

2.3.2.1 神经网络参数设置及 GPU 配置

本文借助 tensorflow^[22] 进行模块开发和配置 GPU 实现

模块正常运行。首先是全连接神经网络输入层与输出层的构建, 网络层参数的设置需要从 tensorflow.contrib.layers 库函数中导入 xavier_initializer, 通过 tf.Variable 指令设置权重与偏置变量, 变量的矩阵大小对应神经元个数即 256。然后根据公式设置层中变量与输入数据的运算关系:

$$y = x \times \omega + b \quad (6)$$

x 为输入数据, ω 为权重, b 为偏置, 需要注意的是, 在设置运算关系时还要借助 tf.nn.tanh 指令添加激活函数。接下来需要根据 MSE 公式设置损失函数参数:

$$MSE = \frac{1}{n} \sum_{i=1}^n (y_i - y_{pre})^2 \quad (7)$$

其中: y_i 表示实际值, y_{pre} 表示预测值。

然后通过 tf.placeholder 指令设置学习率参数并借助损失函数完成优化器参数设置。

网络参数设置完成后需要将整个神经网络的训练和测试过程配置到 GPU 上。为了实现高效计算, 本文借助 tensorflow 中 config 设置的 allow_growth 选项将网络运算进程配置在所有 GPU 内存中以实现神经网络模块在 JETSON TX2 内 GPU 上的配置运行。allow_growth 选项可以根据运算需要自主分配 GPU 内存, 运算开始时此选项会分配较小的内存, 随着网络运算次数增加需要更多的 GPU 内存, 此选项会扩展 tensorflow 进程所需的 GPU 内存区域。

2.3.2.2 神经网络训练与测试过程的 GPU 实现

完成 GPU 运行环境的配置后接下来是训练数据与测试数据的选择, 这些 FPGA 模块数据是借助 PCIe 2.0 X4 通道传输给 JETSON TX2 平台。具体数据的产生、网络训练及测试过程如图 3 所示。

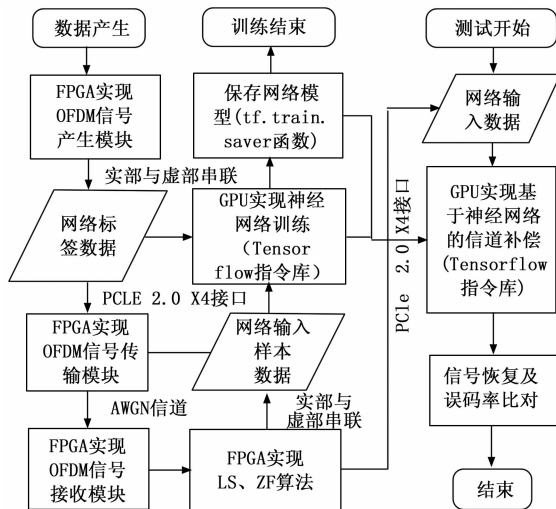


图 3 GPU 实现神经网络信道补偿模块流程图

本文通过多次行 2.2 章节中的 FPGA 实现模块得到大量神经网络的训练数据与测试数据存入 JETSON TX2 平台中, 训练数据包括网络训练标签和网络输入数据集, 训练标签是指分配子载波模块数据, 其数据结构是信号实部与虚部的串联, 网络输入数据集是指 ZF 均衡模块数据, 数据

结构与标签数据相同。测试数据分为网络输入数据集和误码率比对数据，其产生流程、参数设置与训练数据一致，同样从对应 FPGA 模块获取，其中误码率比对数据对应训练数据的训练标签。

本文在 JETSON TX2 平台上存储 $256 \times 1\,000\,000$ 组训练数据， $256 \times 100\,000$ 组测试数据。训练过程的实现为首先通过指令读取训练数据将网络输入集分配在网络的输入端，训练标签分配在网络的输出端，然后再借助 tensorflow 指令设置训练轮数为 5 000，学习率的设置如上文所诉，设置完成后通过指令运行神经网络训练拟合权重与偏置参数。以上为神经网络信道补偿模块的训练实现过程，即神经网络在拟合参数时学习并补偿了信道中未被 LS 信道估计和 ZF 均衡解决的未知误差。当网络训练完成后通过 tf.train.Saver() 等相关指令将网络参数保存在 GPU 中供测试使用，也可不保存参数直接测试网络。测试过程的实现同样先通过指令读取测试数据，将测试网络输入集分配在网络的输入端，运行已训练好的神经网络信道补偿模块得到补偿后的输出端信号数据，将输出端数据与误码率比对数据进行误码率分析显示网络信道补偿模块的实现效果。

3 实验结果及分析

3.1 OFDM 信号收发模块的实现效果及分析

本文在 FPGA 芯片上实现 OFDM 信号收发模块时使用的数据位数为 16 384，信号带宽为 31.25 MHz。在其他模块参数设置不变前提下改变 AWGN 传输信道模块中 E_b/N_0 参数，图 4~6 分别为 E_b/N_0 为 5, 15, 25 时 OFDM 接收信号的时域图和频域图。

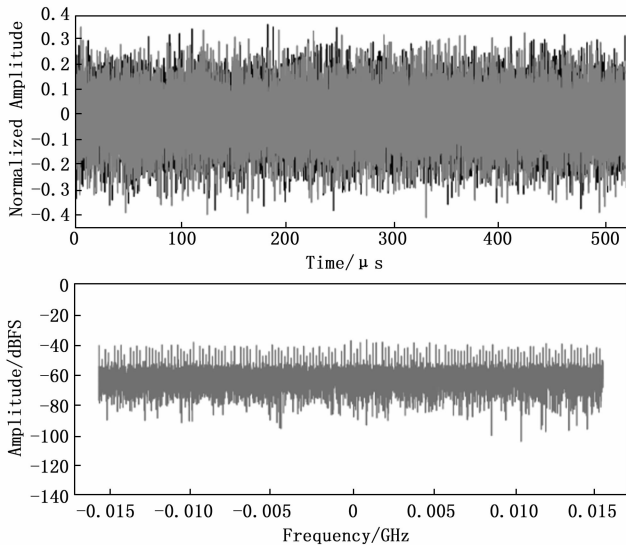


图 4 $E_b/N_0=5$ 时 OFDM 接收信号时域图与频域图

从图 6 来看，OFDM 接收信号模块中输出信号的时域频域图符合 OFDM 信号图像特征，频谱带宽为 31.25 M。其中不同 E_b/N_0 参数值使得 AWGN 信道对接收信号时域部分的影响是不同的，在 E_b/N_0 值较低时信号时域部分受噪声影响较大，当 E_b/N_0 值大于 15 时，噪声影响明显减

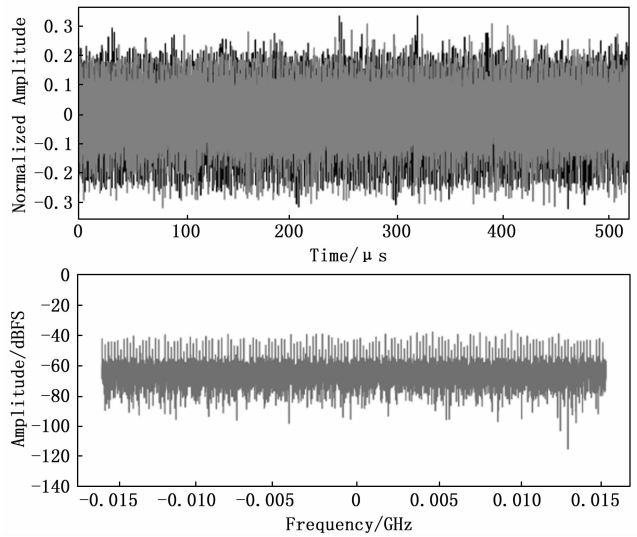


图 5 $E_b/N_0=15$ 时 OFDM 接收信号时域图与频域图

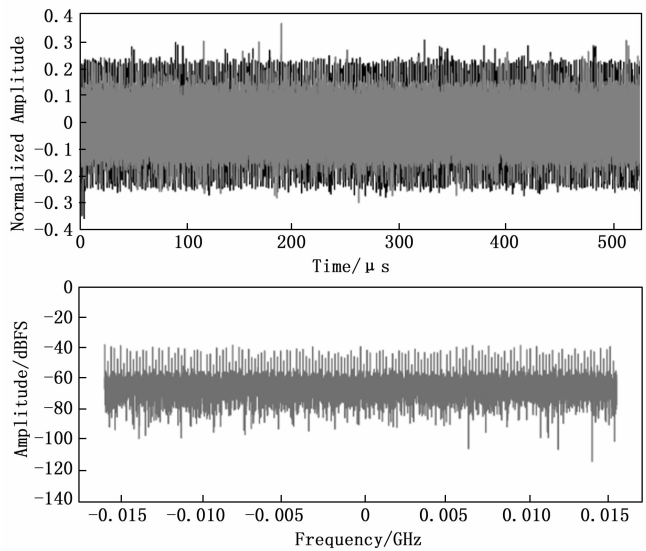


图 6 $E_b/N_0=25$ 时 OFDM 接收信号时域图与频域图

轻，符合 AWGN 信道特性。综上所述说明本文在 FPGA 芯片上有效实现了 OFDM 信号收发及传输模块。

3.2 传统信道估计均衡模块实现结果及分析

针对在 FPGA 上实现传统信道估计与均衡模块的性能测试，本文以模块输出数据与 OFDM 信号产生模块数据进行误码比对，得出如下误码率图：

由图 7 可知，当 E_b/N_0 为 10 时，传统信道估计与均衡模块可实现 10^{-2} 量级的误码率性能，当 E_b/N_0 大于 15 时，此模块可达到 10^{-3} 量级的误码率数据均衡结果，而在实际系统中，此模块性能无法达到 OFDM 信号精确传输的要求，需要对输出信号进行后续信道补偿。

3.3 全连接神经网络模块实现结果及分析

本文通过记录不同 E_b/N_0 参数条件下网络训练过程中训练轮数与损失函数值之间的关系来观察网络的计算复杂度和参数拟合速度，记录结果如图 8~10 所示。

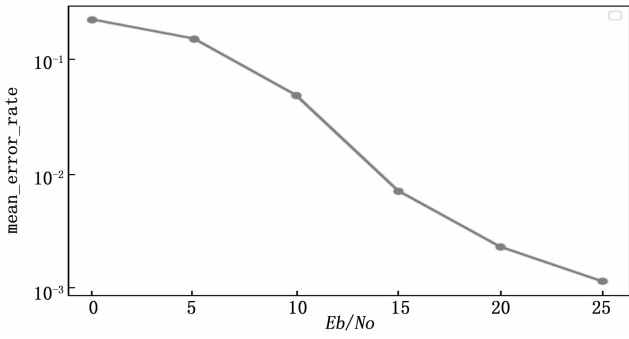


图 7 不同 E_b/N_0 下传统信道估计与均衡模块误码率图

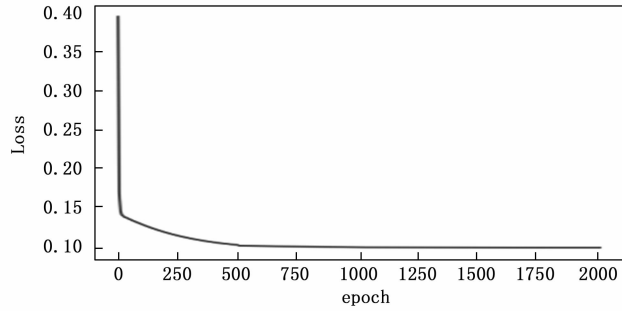


图 8 $E_b/N_0=5$ 时网络训练情况

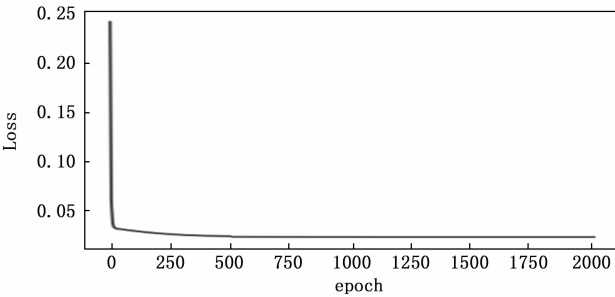


图 9 $E_b/N_0=15$ 时网络训练情况

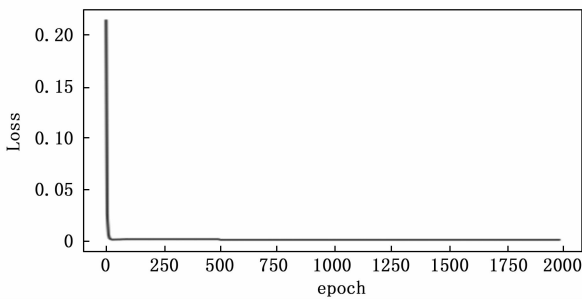


图 10 $E_b/N_0=25$ 时网络训练情况

从图 8 分析得知, 在 E_b/N_0 为 5 时, 全连接神经网络只需 400~500 轮的训练就能达到参数拟合, 在 E_b/N_0 为 15 时, 此网络只需不到 300 轮就能接近拟合, 当 E_b/N_0 为 25 时, 网络只需不到 50 轮就能拟合参数, 这说明此网络复杂度低, 计算量小, 参数拟合速度快, 在硬件实现方面具有结构优势。

本文借助 NVIDIA Jetson 系列边缘盒子配置性能查看工具 jtop 对神经网络模块训练和测试过程中 GPU 的使用情况进行了记录, 记录结果如下:

图 11 神经网络运行之前 GPU 资源使用情况, 未运行网络前 GPU 的使用效率在 0~4% 左右。

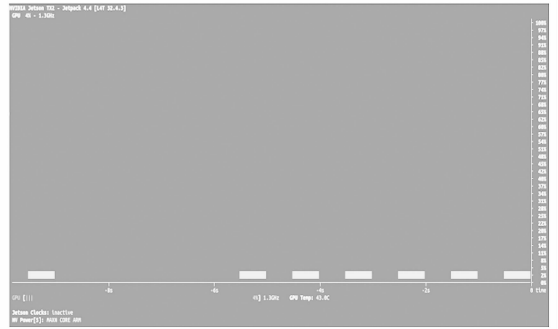


图 11 神经网络未训练时 GPU 使用效率图

图 12 为神经网络训练过程中 GPU 的使用效率情况, 此过程中 GPU 的使用效率从 6% 提升到 99%, 说明网络训练过程占用了 GPU 全部计算资源。

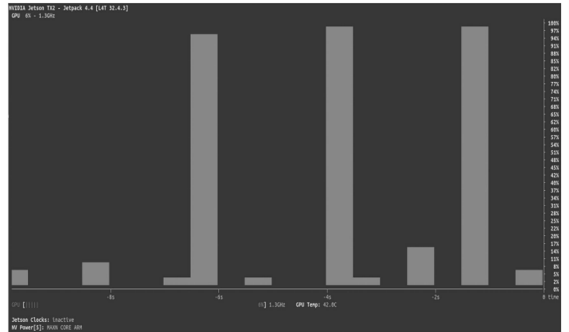


图 12 神经网络训练过程中 GPU 使用效率图

图 13 为神经网络测试过程中 GPU 使用效率情况, 测试过程中 GPU 的使用效率在 4%~76% 之间, 这说明测试过程并不需要占用 GPU 全部计算资源, 同样说明如果将训练好的网络保存在 Jetson TX2 平台上, 再次调用时不需要分配过多硬件资源。

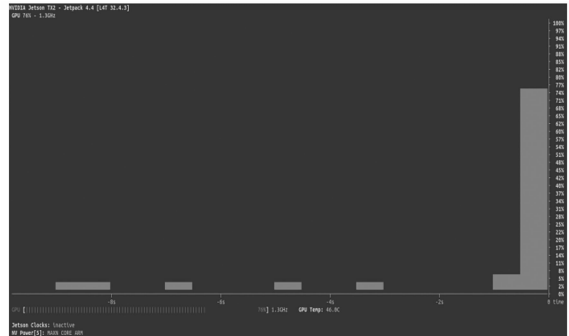


图 13 神经网络测试过程中 GPU 使用效率图

图 14 是不同 E_b/N_0 参数条件下传统信道估计均衡模块与全连接神经网络信道补偿模块的性能对比图。

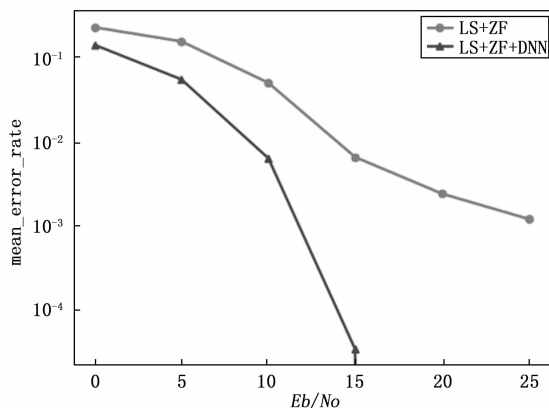


图 14 不同 E_b/N_0 情况下不同模块误码率图

从图中分析得当 E_b/N_0 为 15 时经过信道补偿模块后的数据误码率达到 10^{-5} 量级, 相比传统信道估计均衡模块具有明显的性能优势。从硬件实现方面分析, 由上文可知, 此网络在 GPU 上实现时, 具有网络复杂度低, 计算量小, 参数拟合快的结构优势。此模块还可以通过提前训练将拟合参数保存在 Jetson TX2 平台上供实际系统直接使用, 此过程不需要占用全部 GPU 计算资源。

4 结束语

随着深度学习在嵌入式设备上实现的研究, 简单便携的集成设备已成为基于深度学习无线通信传输系统的实现趋势。结合传统系统以模型驱动为设计理念和基于深度学习以数据驱动为设计基础的 OFDM 信号传输系统实现具有一定的发展前景, 根据不同系统模块计算量大小借助集成设备分配不同硬件计算资源可以高效有序的实现高性能低功耗的基于深度学习信道补偿技术的 OFDM 信号传输系统。

1) 本系统借助 python 对 AIR-T 智能无线电设备进行顶层开发并在 FPGA 芯片上实现了 OFDM 信号产生模块、信号传输模块、信号接收模块, 下一步有望借助 AD9371 收发器实现芯片数据无线信道传输。

2) 本系统基于传统信道估计与均衡模块所需少量运算资源的考虑在 AIR-T 的 FPGA 芯片上实现 LS 信道估计模块、ZF 均衡模块, 通过模块数据的误码率性能分析, 传统信道估计与均衡模块性能无法满足实际传输系统需求, 需要借助神经网络信道补偿模块完成进一步的性能提升。

3) 本系统基于深度学习模块所需大量运算资源的考虑在 AIR-T 的 GPU 上实现了全连接神经网络信道补偿模块, 通过观察分析得出此网络复杂度低, 计算量小, 参数拟合速度快, 这也说明 LS 信道估计与 ZF 均衡模块有效降低了网络训练时的运算次数。从测试性能方面分析, 经过全连接神经网络信道补偿模块后的数据误码率比经过传统信道估计均衡模块后的误码率提高 2 个量级, 具有明显的性能优势。另外测试过程并不需要占用 GPU 全部计算资源, 这说明如果将训练好的网络保存在 GPU 所在平台上, 再次调用时并不需要分配过多硬件资源。

参考文献:

- [1] 徐浙君, 陈善雄. 基于深度学习的弱纹理图像关键目标识别定位方法 [J]. 计算机测量与控制, 2022, 30 (2): 186-191, 200.
- [2] 曹梦硕, 韩军, 陈宝文. 基扩展模型下基于深度学习的双选信道估计方法 [J]. 计算机测量与控制, 2020, 28 (10): 205-210, 215.
- [3] 梁晓芳, 张臻, 张东磊, 等. 一种通信信号的自动调制识别技术研究 [J]. 电子技术与软件工程, 2016 (13): 68-68.
- [4] LI H, FAN X, JIAO L, et al. A high performance FPGA-based accelerator for large-scale convolutional neural networks [C] //2016. 26th International Conference on Field Programmable Logic and Applications (FPL), IEEE, 2016: 1-9.
- [5] SOLTANI S, SAGDUYU Y E, HASAN R, et al. Real-time and embedded deep learning on FPGA for RF signal classification [C] //MILCOM 2019-2019 IEEE Military Communications Conference (MILCOM). IEEE, 2019: 1-6.
- [6] XUE J, LI J, YU D, et al. Singular value decomposition based low-footprint speaker adaptation and personalization for deep neural network [C] //2014 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP), IEEE, 2014: 6359-6363.
- [7] HAN S, MAO H, DALLY W J. Deep compression: Compressing deep neural networks with pruning, trained quantization and Huffman coding [J]. arXiv preprint arXiv: 1510.00149, 2015.
- [8] HINTON G, VINYALS O, DEAN J. Distilling the knowledge in a neural network [J]. Computer Science, 2015, 14 (7): 38-39.
- [9] QIU J, WANG J, YAO S, et al. Going deeper with embedded FPGA platform for convolutional neural network [C] //Proceedings of the 2016 ACM/SIGDA International Symposium on Field-Programmable Gate Arrays, 2016: 26-35.
- [10] 邓畅, 陆骏, 李广. 深度学习在云和终端上的混合分布式部署研究 [J]. 智能物联技术, 2019, 51 (1): 12-17.
- [11] 高榕, 张良, 梅魁志. 基于 Caffe 的嵌入式多核处理器深度学习框架并行实现 [J]. 西安交通大学学报, 2018, 52 (6): 36-41, 113.
- [12] 王得光, 杭子钧, 黄友, 等. 基于 CPU-FPGA 异构多核平台的卷积神经网络并行加速研究 [C] //第二十二届计算机工程与工艺年会暨第八届微处理器技术论坛论文集, 2018: 103-112.
- [13] O' SHEA T J, ROY T, CLANCY T C. Over-the-air deep learning based radio signal classification [J]. IEEE Journal of Selected Topics in Signal Processing, 2018, 12 (1): 168-179.
- [14] 金涛, 郭鸿韬, 刘浩宇, 等. 基于嵌入式平台的路面信息监测系统 [J]. 数字通信世界, 2019 (5): 14-15, 19.
- [15] 王莹莹, 周鹏, 牟军. Xilinx 公司 FPGA 配置方法分析及实现 [J]. 电子质量, 2020 (8): 20-23.

(下转第 162 页)