计算机科学
COMPUTER SCIENCE

基于FPGA并行实现SVM训练的可重构计算系统

彭卫东, 郭威, 魏麟

引用本文

彭卫东, 郭威, 魏麟. 基于FPGA并行实现SVM训练的可重构计算系统[J]. 计算机科学, 2024, 51(11A): 231100120-7.

PENG Weidong, GUO Wei, WEI Lin. Reconfigurable Computing System for Parallel Implementation of SVM Training Based on FPGA [J]. Computer Science, 2024, 51(11A): 231100120-7.

计算机科学 COMPUTER SCIENCE
http://www.jsjkx.com
DOI:10.11896/jsjkx.231100120

基于 FPGA 并行实现 SVM 训练的可重构计算系统

彭卫东[1]　郭　威[1]　魏　麟[2]
1 中国民用航空飞行学院航空电子电气学院　四川 广汉 618300
2 中国民用航空飞行学院飞行技术学院　四川 广汉 618300
(893745910@qq.com)

摘　要　针对支持向量机在处理大规模数据集时所面临的计算复杂度高和训练时间长的问题,设计了一种基于 FPGA 并行实现支持向量机训练的可重构计算系统,并分析了不同量化方式下的硬件资源消耗与加速性能。通过采用随机梯度下降法训练支持向量机,使得需要求解的维度与样本的维度相关联,相较于传统的基于二次规划的求解方法可以显著降低计算复杂性。同时,利用基于 FPGA 的可重构硬件平台设计了专用并行计算结构以加速支持向量机的训练过程。对设计的完整系统进行了软硬件联合仿真,在 4 个公共数据集上的仿真结果表明,整体模型预测准确率达到 90%以上;在训练阶段,相较于采用相同算法的软件实现,所提出的浮点数表示下硬件实现的单个样本处理时间至少减少了 2 个数量级;定点数表示下硬件实现的单个样本处理时间最大减小了 3 个数量级;与基于二次规划问题求解的硬件实现相比,单个样本处理速度最快提升了 394 倍。

关键词:现场可编程逻辑门阵列;支持向量机;可重构系统;并行计算;随机梯度下降法
中图分类号　TP181;TN791

Reconfigurable Computing System for Parallel Implementation of SVM Training Based on FPGA

PENG Weidong[1],GUO Wei[1] and WEI Lin[2]
1 Institute of Electronic and Electrical Engineering,Civil Aviation Flight University of China,Guanghan,Sichuan 618300,China
2 Institute of Flight Technology,Civil Aviation Flight University of China,Guanghan,Sichuan 618300,China

Abstract　To address the problems of high computational complexity and long training time faced by support vector machines when dealing with large-scale datasets,a reconfigurable computing system for parallel SVM training based on FPGA is designed. The hardware resource consumption and acceleration performance under different quantization methods are analyzed. By utilizing the stochastic gradient descent method for SVM training,the dimensions to be solved are associated with the sample dimensions,significantly reducing computational complexity compared to traditional quadratic programming-based methods. Additionally,a specialized parallel computing structure is designed using FPGA-based reconfigurable hardware platform to accelerate the SVM training process. The entire system is jointly simulated in software and hardware. Simulation results on four public datasets show that the overall model prediction accuracy exceeds 90%. During the training phase,compared to software implementation using the same algorithm,the proposed hardware implementation reduces the processing time for a single sample by at least two orders of magnitude under floating-point representation. Under fixed-point representation,the processing time for a single sample is reduced by up to three orders of magnitude. Compared to the hardware implementation based on quadratic programming problem solving,the processing speed for a single sample is improved by up to 394 times.

Keywords　FPGA,Support vector machine,Reconfigurable system,Parallel computing,Stochastic gradient descent

1　引言

支持向量机(Support Vector Machine,SVM)是机器学习领域中重要的算法之一,通常用于执行分类和回归任务的分析[1]。SVM 因具有对未知数据集较好的预测精度、处理不平衡数据集的有效性以及解决高维度与复杂非线性等学习问题的能力,被广泛应用于图像分类、语音识别、目标检测以及异常检测等领域。然而,SVM 的训练阶段属于计算密集型过程,一般是通过求解二次规划问题来获得支持向量[2],其维度相当于训练样本的数量。由于 SVM 训练阶段的计算成本较高,因此,虽然传统的软件实现具有良好的数值精度,但无法满足一些需要低功耗和实时处理的应用场景下的要求,尤其是对于大规模数据集的训练。

可重构硬件平台具有的高吞吐量以及低功耗的优势可用来加速 SVM 的训练过程,例如现场可编程逻辑门阵列(Field Programmable Gate Array,FPGA)。在一些使用硬件加速

基金项目:民航飞行技术与飞行安全重点实验室自主研究项目(XYKY2023018);民机综合航电技术研究所科研创新团队项目(CZKY2023161)
This work was supported by the Independent Research Project of Key Laboratory of Flight Techniques and Flight Safety(XYKY2023018) and Research and Innovation Team Project of Civil Aircraft Integrated Avionics Technology Research Institute(CZKY2023161).
通信作者:郭威(fpga2516@163.com)

SVM 训练的过往研究中，大部分研究主要集中在使用序列最小优化（Sequential Minimal Optimization，SMO）算法实现 SVM 的训练[3-5]。但是，由于 SMO 算法固有的顺序特性，以往研究设计的 SVM 硬件架构难以实现并行计算或只能实现低效并行[6]。此外，随着使用机器学习技术的嵌入式系统数量的增加以及物联网中边缘人工智能领域的兴起，越来越多的研究将人工智能前沿技术推向网络边缘，以利用数据源附近的物联网设备产生的大量数据进行实时分布式训练[7]，而基于 SMO 算法设计的 SVM 硬件架构无法为大规模数据集的训练提供更快的训练速度，也无法支持物联网设备中的大量工作负载。尽管随机梯度下降法（Stochastic Gradient Descent，SGD）已广泛应用于边缘人工智能领域的分布式机器学习中[8]，但少部分研究关注基于梯度的方法进行 SVM 训练的硬件实现。文献[9]中 Ho 等通过集成 Apache Spark 数据处理框架，利用 FPGA 中的 SGD 实现来加速线性 SVM 的分布式训练，并将该实现用于对 7500 个细胞图像进行分类。与计算机集群上的软件实现相比，其吞吐量最多提高了 2 倍。但其 FPGA 内部为串行计算设计，并且只实现了线性 SVM，无法处理非线性数据集。

为进一步扩展 SVM 的泛化能力，并增强专用硬件架构在未来边缘设备上的适应性，本文设计了一种用 FPGA 加速 SVM 训练的可重构计算系统，并在 FPGA 中实现了基于 SGD 训练 SVM 的并行计算结构。

2 基于 SGD 的 SVM 训练

SVM 是一种有监督训练模型，其训练过程一般是通过最小化一个二次目标函数来找到最优超平面以划分训练样本[10]。给定 N 个训练样本与标签组成的输入数据集 $\{(\vec{x}_n, y_n), n=1,\cdots,N\}$，其中 $\vec{x}_n \in R^M$ 为输入样本，R^M 表示每个样本具有 M 维输入空间，$y_n \in \{-1,+1\}$ 为相应的类别标签。根据分类间隔最大化原则，SVM 的原始优化问题可表示为：

$$\begin{aligned} &\min_{\vec{w},b,\xi_n} \frac{1}{2}\|\vec{w}\|^2 + C\sum_{n=1}^{N}\xi_n \\ &\text{s.t. } y_n[(\vec{w}^{\mathrm{T}}\cdot\vec{x}_n)+b] \geqslant 1-\xi_n \\ &\quad \xi_n \geqslant 0, n=1,\cdots,N \end{aligned} \tag{1}$$

其中，$\vec{w}$ 与 b 分别为最优超平面的法向量与偏移量；ξ_n 为第 n 个训练样本被错误分类后产生的误差项，C 为误差项 ξ_n 的权衡参数，用来确定决策边际最大化与分类误差最小化之间的权衡。为求解此原始优化问题(1)，大量的研究[11-14]集中在利用基于拉格朗日函数和对偶问题的求解思想，采用传统的基于二次规划的方法求解其对偶问题，如使用 SMO 算法训练 SVM。然而，使用基于二次规划的方法解决 SVM 问题时，问题的复杂度与问题的维度相关[15]，当问题的维度很大，即训练样本很多时，计算所需的时间和资源会显著增加，使得 SVM 无法应对大规模数据集的训练。采用基于梯度的方法训练 SVM，可将原始优化问题(1)转化为无约束的经验风险最小化问题[16]，使问题的维度与样本的维度（特征的数量）相关，相比于传统基于二次规划的方法可以显著降低计算复杂性。为有效使用 SGD 训练 SVM，基于铰链损失函数[17]，无约束的经验风险最小化问题可表示为：

$$\min_{\vec{w},b} \frac{\lambda}{2}\|\vec{w}\|^2 + \frac{1}{N}\sum_{n=1}^{N}\max\{0,1-y_n[(\vec{w}^{\mathrm{T}}\cdot\vec{x}_n)+b]\} \tag{2}$$

其中，λ 为正则化参数，引入正则项的目的是防止模型过拟合并提高其泛化能力。为解决问题(2)，采用基于梯度的方法更新迭代过程中的权重向量 $\vec{w}$ 以优化 SVM 的训练过程，若每次迭代过程中只随机选择一个数据 $(\vec{x}_{n_t}, y_{n_t})$，其中索引 $n_t \in \{1,\cdots,N\}$ 为均匀随机过程，以保证训练样本被均匀随机地抽取，则问题(2)被简化为：

$$\begin{aligned} \min F(\vec{w},b) = \min_{\vec{w},b} &\frac{\lambda}{2}\|\vec{w}\|^2 + \max\{0,1- \\ &y_{n_t}[(\vec{w}^{\mathrm{T}}\cdot\vec{x}_{n_t})+b]\} \end{aligned} \tag{3}$$

其中，$F(\vec{w},b)$ 为近似目标函数。对于问题(3)来说，在算法的第 $t+1$ 次迭代中，权重向量 $\vec{w}$ 的更新为：

$$\vec{w}_{t+1} = \vec{w}_t - \alpha\,\nabla_t F(\vec{w},b)|_{\vec{w}=\vec{w}_t} \tag{4}$$

其中，α 为学习率，$\nabla_t F(\vec{w},b)|_{\vec{w}=\vec{w}_t}$ 为近似目标函数 $F(\vec{w},b)$ 的梯度 $\vec{g}$，可近似计算为[18]：

$$\vec{g} = \lambda\vec{w}_t - \begin{cases} y_{n_t}\vec{x}_{n_t}, & \text{if } y_{n_t}[(\vec{w}_t^{\mathrm{T}}\cdot\vec{x}_{n_t})+b]<1 \\ 0, & \text{otherwise} \end{cases} \tag{5}$$

此时，将关注点放回到 SVM 解决二元分类问题上，模型的输出预测分类包括将输入样本 $\vec{x}$ 分类为 $\{-1,+1\}$ 两个可能类中的一个，第 t 次迭代中决策函数可表示为：

$$f(\vec{x}_{n_t}) = \mathrm{sgn}(\vec{w}_t^{\mathrm{T}}\cdot\vec{x}_{n_t}+b) \tag{6}$$

根据式(4)－式(6)可以看出基于 SGD 的 SVM 训练过程中，模型权重向量 $\vec{w}$ 与决策函数 $f(\vec{x}_{n_t})$ 的更新涉及到的计算复杂度较低，有利于 SVM 的硬件化实现。然而，以上的分析基于 SVM 对线性可分离模式进行分类，现实中的训练数据集通常具有非线性可分离性，需要使用核方法[19]，通过一个非线性映射将输入样本空间转换至相应特征空间，使转换后的样本可被超平面分离。核函数的计算将耗费较大硬件资源，这成为在 FPGA 上并行实现 SVM 设计的关键因素。

3 SVM 的硬件并行实现

本文设计的基于 FPGA 并行实现 SVM 训练的可重构计算系统整体框架如图 1 所示，FPGA 通过以太网通信协议与外部主机进行交互，其内部通过 AXI（Advanced eXtensible Interface）总线高速传输数据与控制指令。训练数据集被存储在片外随机存储器内，通过硬件 AXI-DMA 接口，以低延迟的直接内存访问（Direct Memory Access，DMA）的方式将训练数据集传输至 SVM 核心模块。

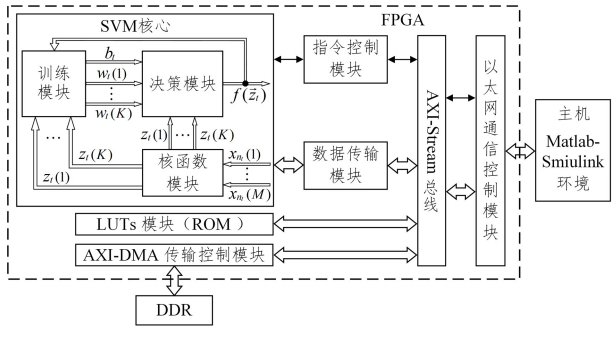

图 1 基于 FPGA 的可重构计算系统整体框架

Fig. 1 Framework of FPGA-based reconfigurable computing system

在 SVM 训练的核心部分，构建了 3 个关键模块：核函数模块、训练模块以及决策模块。为提升整个系统的吞吐量，利用计算过程中固有的数据并行性，将 3 个模块独立设计为并行计算电路结构。同时，各模块之间采用流水线并行处理的方式，进一步优化计算效率。为了在 SVM 的核心模块中实现 SGD，将数据的输入方式从以训练样本个数为输入转变为输入单个样本的所有特征。在每次迭代过程中，核心模块会接收单个样本的全部特征作为输入，并在固定的采样时间内利用基于梯度的方法进行 SVM 的训练，这种方法在处理大规模数据集时，可有效地降低计算复杂性，提高训练效率。对于第 t 次迭代过程来说，若输入样本为具有 M 个特征的向量 $\vec{x}_{n_t}$，核函数模块首先通过核方法将所有特征 $\{x_{n_t}(m)\}_{m=1}^{M}$ 映射到相应特征空间，映射后的特征向量 $\vec{z}_t$ 的所有维度分量 $\{z_t(k)\}_{k=1}^{K}$ 被输入至训练模块与决策模块；其次训练模块将会根据 SGD 更新每个维度分量对应的权重 $\{w_t(k)\}_{k=1}^{K}$ 与偏移量 b_t；最后决策模块输出 SVM 分类结果，并将其反馈至训练模块，用于第 $t+1$ 次迭代的权重更新。

3.1 核函数模块设计

在核函数模块中，采用径向基核函数（Radial Basis Function Kernel，RBFK）将输入样本空间由 R^M 映射到特征空间 R^K，映射后的特征向量 $\vec{z}_t$ 的第 k 个维度分量可表示为：

$$z_t(k)=\exp[-\gamma d_t(k)]=\exp\left\{-\frac{\sum_{m=1}^{M}[x_{n_t}(m)-u_t^{(k)}(m)]^2}{2\sigma^2}\right\} \tag{7}$$

其中 σ^2 为 RBFK 的方差，γ 为 $1/2\sigma^2$，$u_t^{(k)}(m)$ 表示与第 m 个输入特征相关的第 k 个 RBFK 中心，$d_t(k)$ 为每个输入特征与对应的核函数中心之间的第 k 个平方欧氏距离[20]。将式(6)的计算分为两部分，可以看出主要计算量来自于指数函数计算与 $d_t(k)$ 的计算。为减少 FPGA 中的资源消耗和执行时间，本文使用只读存储器（Read-Only Memory，ROM），以查找表（Lookup Table，LUT）的形式实现指数函数的运算，8 位地址输入的 LUT 实现通过多路选择器与两个 4 输入 LUT 的组合来完成，可存储 256 个指数函数值。相比于采用特定硬件电路的实现，将指数函数的值存储在 LUT 的每个地址中对计算延迟的影响更小。此时，式(6)中的主要计算量来自于 $d_t(k)$，即差值平方和的计算，其中 RBFK 的中心是通过采用 K 均值聚类算法[21]选择每个聚类的最终中心作为核函数的中心获得的。为优化整体系统的吞吐量，核函数模块由多个并行计算的核函数映射单元组成，设计的单个核函数映射单元并行计算电路结构如图 2 所示。

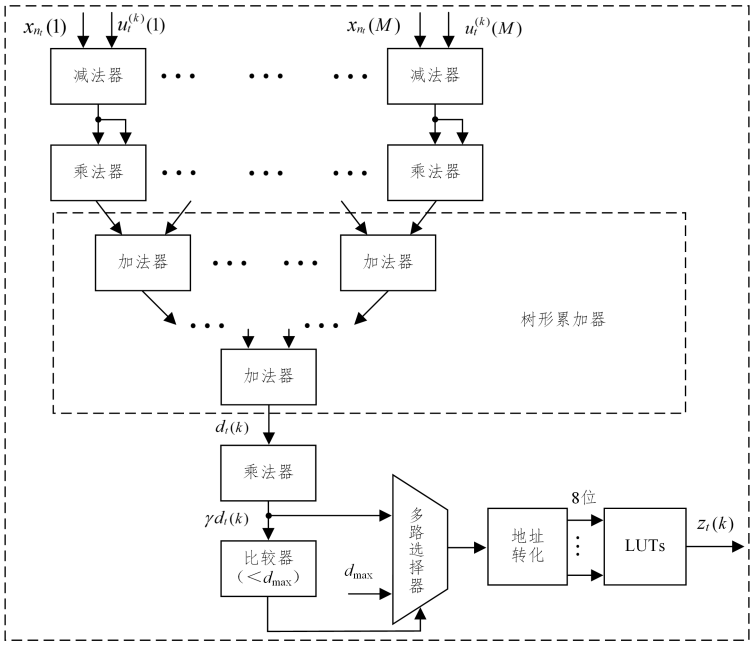

图 2 单个核函数映射单元并行计算电路结构

Fig. 2 Parallel computing circuit structure of single kernel function mapping unit

图 2 显示了一个核函数映射的完整执行过程，所有输入特征与对应的核函数中心之间的减法与乘法均以并行计算的方式执行，同时利用树形加法器结构完成累加求和运算以提高吞吐量，将多个加法操作分层级地组织在一起，其内部相同深度下的每个加法器都在同一个流水线阶段，最终树形加法器输出距离值 $d_t(k)$。随后，$d_t(k)$ 通过一个乘法器与 γ 相乘，其结果被输入至比较器，该比较器内部设置有距离阈值 d_{max}，以防止 $d_t(k)$ 超出 LUT 地址范围。地址转化模块通过将多路选择器输出的距离值按照规定的步长 $d_{max}/256$ 离散化为 8 位地址进行输出，最终 8 位地址被输入至 LUT 以索引相应的指数函数值 $z_t(k)$。

根据以上分析可知，核函数模块的迭代处理时间取决于输入样本维度，而核函数映射单元的数量随核函数的使用个数的增加而增加，每个映射单元内部并行执行的减法器与乘法器数量与输入样本维度线性相关，树形加法器的深度与输入样本维度呈对数依赖关系。当输入样本维度为 M 时，树形加法器深度为 $\log_2(M)$，但其内部加法器数量随输入样本维度的增加而线性增加。

3.2 训练模块设计

在每次迭代过程中，训练模块基于 SGD 更新权重与偏移量，根据式(4)、式(5)可得出在第 $t+1$ 次迭代过程中第 k 个权重 $w_{t+1}(k)$ 与偏移量 b_{t+1} 的更新公式，分别为：

$$w_{t+1}(k)=(1-\lambda\alpha)w_t(k)+\begin{cases}\alpha y_{n_t}z_t(k), & \text{if } y_{n_t}f(\vec{z}_t)<1\\ 0, & \text{otherwise}\end{cases} \tag{8}$$

$$b_{t+1}=(1-\lambda\alpha)b_t+\begin{cases}\alpha y_{n_t}, & \text{if } y_{n_t}f(\vec{z}_t)<1\\ 0, & \text{otherwise}\end{cases} \tag{9}$$

其中，偏移量的迭代更新方式与权重相近。在训练模块中构建多个权重更新计算单元同步并行更新对应于每个维度分量的权重，设计的权重更新并行计算电路结构如图 3 所示。在第 $t+1$ 次迭代过程中，所有权重更新的整个处理过程为，由决策模块反馈的分类输出 $f(\vec{z}_t)$ 与期望分类 y_{n_t} 相乘后被输入至比较器，比较器根据乘积结果判断是否小于 1，若小于 1 则输出高电平至多路选择器，反之则输出低电平。若多路选择器接收到高电平信号，则输出 1，反之则输出 0。此时，每个权重更新计算单元并行执行一系列乘法与减法运算，完成本次迭代过程中所有权重的同步更新，并将此次更新的全部权重存储在各自的寄存器内。寄存器会存储并输出本次迭代更新的权重，并在下次迭代过程中被调用，图 3 中每个寄存器的最终输出为第 t 次迭代更新的权重。

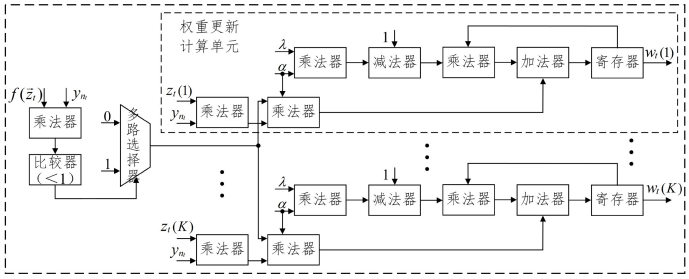

图 3 权重更新并行计算电路结构

Fig. 3 Parallel computing circuit structure of weight updating

因构建有多个并行计算的权重更新计算单元，且每个计算单元消耗的硬件资源与输入 $\vec{z}_t$ 的维度分量个数无关，故训练模块的迭代处理时间保持不变，但整个模块消耗的逻辑元素数量将随输入 $\vec{z}_t$ 维度的增加而线性增加。

3.3 决策模块设计

基于 FPGA 设计的决策模块并行计算电路结构如图 4 所示，该模块中 $\vec{z}_t$ 的每个维度分量与对应的权重并行相乘且与偏移量相加后再累加求和，根据式(6)可得出第 t 次迭代过程的决策函数公式为：

$$f(\vec{z}_t)=\mathrm{sgn}\left[\sum_{k=1}^{K}(w_t(k)\cdot z_t(k)+b_t)\right] \tag{10}$$

为提高吞吐量，累加求和运算的实现是一种流水线设计，采用树形加法器结构，其内部处于相同层级的各相邻加法器之间的输出会被并行地执行求和运算，同时允许在给定阶段处理下一个样本而无需等待后续阶段的完成。随后，比较器接收树形加法器最后一层的累加结果，通过判断累加结果的正负来输出高/低电平信号，sgn 函数通过比较器与多路选择器配合实现。当比较器判断输入为大于 0 时，将输出高电平信号至多路选择器，多路选择器将输出 1 为最终分类结果，否则将输出 −1 为最终结果。同时，分类结果将被反馈至训练模块，用于下次迭代过程中权重的更新。

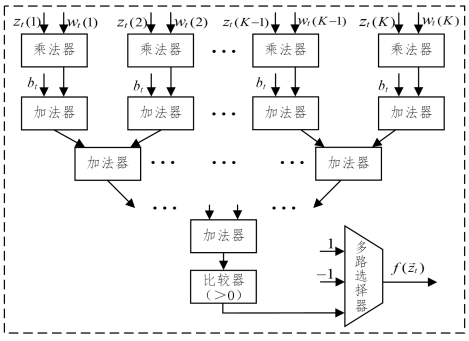

图 4 决策模块并行计算电路结构

Fig. 4 Parallel computing circuit structure of decision module

通过上述分析可知，决策模块的迭代处理时间和消耗的硬件资源数量与 $\vec{z}_t$ 的维度分量个数呈线性依赖关系，其内部乘法器与加法器的数量随 $\vec{z}_t$ 维度的增加而线性增加，且树形加法器内部加法器数量随输入样本维度的增加而线性增加。而其深度与 $\vec{z}_t$ 的维度呈对数依赖关系，当 $\vec{z}_t$ 维度为 K 时，树形加法器深度为 $\log_2(K)$。

4 实验与分析

本文对设计的完整系统进行了软硬件联合仿真，首先对比了软硬件实现下对不同数据集的预测准确率，其次分析了硬件实现的并行加速性能以及资源消耗，最后评估了硬件设计架构的可扩展性。在硬件实现中，采用的硬件平台为 Xilinx 公司 Kintex-7 系列的 XC7K325T FPGA 芯片以及 KC705 评估套件。FPGA 内部的寄存器传输级(Register-Transfer Level，RTL)电路设计通过使用集成在 Matlab Simulink 环境中的 Xilinx 的 System Generator 工具完成，该工具可与 FPGA 交互以进行软硬件联合仿真。系统运行时钟设置为 150 MHz，通过将系统运行时钟频率设置为远大于单个样本输入的时钟频率，保证单个样本处理的实时性。在软件实现中，训练算法通过在 Matlab R2022a 软件中使用 m 语言编写代码完成，使用的通用计算机配置为 8GB 内存，12 核基准速度为 2.5 GHz 的处理器。

实验所采用的 4 个标准数据集均为二分类样本[1)]，如表 1 所列。为实现交叉验证，对数据集进行预处理操作，除数据集 svmguide1 外的 3 个数据集均按 6∶4 的比例随机划分为训练集与测试集，同时数据集内的样本标签均被归一化为 +1 和 −1。对于训练集的模型训练，权重向量 $\vec{w}$ 初始化为 0，偏移量 b 初始化为 1，算法的迭代次数固定为 12 000 次，通过在 Matlab 内实现 K 均值聚类算法获得 RBFK 的中心，并将获得的 mat 类型文件转化为 csv 类型文件以供硬件使用。对于硬件实现，采用单精度浮点数与定点数两种不同的量化方式，其

1) https://www.csie.ntu.edu.tw/~cjlin/libsvmtools/datasets/

中定点数表示首位为符号位，具有 4 位整数位与 20 位小数位。对于软件实现，只采用单精度浮点数的表示方式。

表 1　4 种标准数据集

Table 1　Four standard datasets

数据集	总样本个数	训练集样本个数	测试集样本个数	特征个数
fourclass	812	512	340	2
svmguide1	7089	3089	4000	4
breast-cancer_scale	683	410	273	10
vehicle_scale	846	508	338	18

4.1　预测准确率验证

将训练完成的模型用于测试集的预测，使用混淆矩阵计算软硬件实现下模型的预测准确率，通过将各数据集按照理想标签类别与模型预测类别绘制为矩阵的形式，得到了理想情况和预测结果分别为正例和反例时对应的数量，即真正例（TP）、假正例（FP）、真反例（TN）、假反例（FN），则预测准确率可计算为：

$$\text{预测准确率}=\frac{TP+TN}{TP+TN+FP+FN} \tag{11}$$

对于每个数据集，通过交叉验证获得的最佳参数如表 2 所示，其中 K 为通过 K 均值聚类算法找到的 RBFK 的中心个数，即核函数使用个数。图 5 显示了根据计算获得的各数据集在不同实现下的预测准确率。

表 2　各数据集相关参数

Table 2　Parameters related to various datasets

数据集	λ	γ	α	K
fourclass	1.54×10^{-4}	8	0.6	4
svmguide1	2.82×10^{-5}	2	0.6	5
breast-cancer_scale	1.46×10^{-4}	8	0.6	4
vehicle_scale	1.48×10^{-4}	8	0.6	6

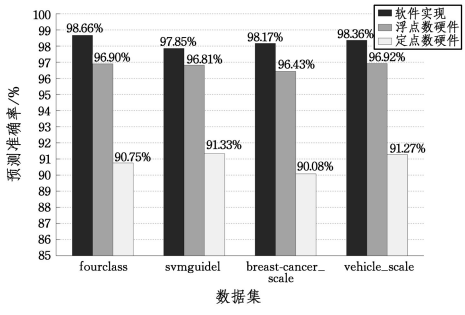

图 5　不同实现下模型对各数据集的预测准确率

Fig. 5　Prediction accuracy of models on different datasets under various implementations

可以看出，使用相同单精度浮点数表示的软硬件实现下，相较于软件实现，硬件实现的模型整体预测准确率降低了 1%左右；而对于不同量化表示下的硬件实现，相较于单精度浮点数表示的模型，使用定点数表示的模型整体预测准确率下降了 6%左右。软件实现的模型预测准确率达到了 97%以上；而硬件实现下，对于使用单精度浮点数表示的模型预测准确率达到了 96%以上，而对于使用定点数表示的模型预测准确率也达到了 90%以上。

4.2　并行加速性能分析

通过对单个样本处理时间 t_s 的分析来评估硬件实现的加速性能，并且通过对比核函数的使用个数 K 来分析硬件资源使用情况。对于硬件实现，因每次迭代过程中只处理一个样本的全部特征，故单个样本处理时间即为系统采样时间，则硬件实现下的训练系统吞吐量为 $1/t_s$，单位为每秒采样次数（Sample Per Second，SPS）；对于软件实现下的单个样本处理时间，由固定迭代 12000 次的总迭代时间求平均获得。对数据集 fourclass 与 svmguide1 进行了软硬件实现下硬件资源使用情况与单个样本处理时间的分析，表 3 与表 4 列出了不同量化方式下核函数使用个数对触发器（Flip-flop，FFs，共 407600 个）、数字信号处理器（Digital signal processors，DSPs，共 840 个）以及 LUTs（共 203800 个）3 种主要硬件资源消耗的影响，表 5 和表 6 列出了不同量化方式下硬件实现相较于软件实现的加速性能。

可以看出，对于两个数据集来说，硬件资源的整体消耗都随着核函数使用个数的增加而增加，定点数表示下的硬件资源消耗明显低于单精度浮点数表示下的硬件资源消耗。随着核函数使用个数增加，两种不同量化方式下硬件实现的吞吐量均逐渐降低，这是由决策模块内部的树形加法器深度增加造成的。并且相比于定点数表示下的硬件实现，单精度浮点数表示下的硬件实现的系统吞吐量受核函数使用个数的影响更为明显。但相较于软件实现，两种量化方式下的硬件实现均将单个样本处理时间 t_s 减少了至少 2 个数量级。以数据集 fourclass 为例，对于单精度浮点数表示下的硬件实现每秒最快可处理大约 850 万个样本，而对于定点数表示下的硬件实现每秒最快可处理大约 1350 万个样本，两种量化方式下的硬件实现均展现出了不错的加速性能。例如处理 1GB 大小的样本，即 2^{30} 个字节，采用单精度浮点数表示时每个样本由 32 比特组成，即 4 个字节，在模型均使用两个核函数的情况下，对于软件实现需要至少 25 个小时左右的时间训练全部样本，而对于硬件实现则可以在 32 秒左右计算全部样本；同时，采用定点数表示时每个样本由 25 比特组成，即 3.125 个字节，对于使用相同核函数个数的硬件实现，可以在 26 秒左右计算全部样本。在使用 2 个核函数时，对比软件实现，对数据集 fourclass 训练时，定点数表示下的硬件实现最大加速倍数达到 4400 倍左右，浮点数表示下的硬件实现最大加速倍数达到 2700 倍左右；而对数据集 svmguide1 训练时，定点数表示下的硬件实现最大加速倍数达到 4700 倍左右，浮点数表示下的硬件实现最大加速倍数达到 3100 倍左右，这种加速性能对于大规模数据集的训练至关重要。

表 3　对 fourclass 数据集硬件资源消耗

Table 3　Hardware resource consumption of fourclass

核函数使用个数 K	浮点数表示下硬件实现			定点数表示下硬件实现		
	FFs	DSPs	LUTs	FFs	DSPs	LUTs
$K=2$	5952	84	11703	2035	58	972
$K=4$	9140	137	22071	3126	98	1855
$K=6$	14739	221	37210	5372	167	3170
$K=12$	25845	396	72646	10624	312	6194
$K=24$	58064	785	146854	26784	650	12506

表 4　对 svmguide1 数据集硬件资源消耗

Table 4　Hardware resource consumption on svmguide1 dataset

核函数使用个数 K	浮点数表示下硬件实现			定点数表示下硬件实现		
	FFs	DSPs	LUTs	FFs	DSPs	LUTs
$K=2$	6274	91	13236	2099	65	1109
$K=4$	9903	148	24797	3582	107	2094
$K=6$	15691	253	41945	5941	187	3614
$K=12$	28817	451	83570	12080	355	7271
$K=24$	66753	824	178209	31685	679	16105

表 5 对 fourclass 数据集硬件加速性能

Table 5 Hardware acceleration performance on fourclass dataset

核函数使用个数 K	Matlab 软件实现		浮点数表示下硬件实现			定点数表示下硬件实现		
	t_s/μs	吞吐量/Ksps	t_s/μs	吞吐量/Ksps	加速倍数	t_s/μs	吞吐量/Ksps	加速倍数
$K=2$	328.3308	3.0457	0.1174	8.5179	2797	0.0739	13.5318	4443
$K=4$	328.1359	3.0475	0.2306	4.3365	1423	0.0771	12.9702	4256
$K=6$	326.4375	3.0634	0.3045	3.2841	1072	0.0831	12.0337	3928
$K=12$	320.5324	3.1198	0.4992	2.0032	642	0.0879	11.3766	3647
$K=24$	319.3743	3.1311	0.9413	1.0623	339	0.0964	10.3734	3313

表 6 对 svmguide1 数据集硬件加速性能

Table 6 Hardware acceleration performance on svmguide1 dataset

核函数使用个数 K	Matlab 软件实现		浮点数表示下硬件实现			定点数表示下硬件实现		
	t_s/μs	吞吐量/Ksps	t_s/μs	吞吐量/Ksps	加速倍数	t_s/μs	吞吐量/Ksps	加速倍数
$K=2$	440.1045	2.2121	0.1394	7.1736	3157	0.0925	10.8108	4758
$K=4$	437.2576	2.2870	0.2621	3.8153	1668	0.0931	10.7411	4697
$K=6$	436.1393	2.2928	0.3745	2.6702	1165	0.0974	10.2669	4478
$K=12$	434.5288	2.3013	0.6085	1.6434	714	0.1061	9.4251	4095
$K=24$	434.2631	2.3028	1.2413	0.8056	539	0.1096	9.1241	3962

4.3 可扩展性评估

受限于所使用的硬件固有的资源限制，无法对大规模高维数据集直接进行训练，因此采用线性回归分析法对所提出的硬件架构进行可扩展性评估。由于硬件实现的系统吞吐量主要受核函数使用个数的影响，因此基于对数据集 fourclass 的实验结果，通过最小二乘法进行数据拟合，分别建立了在不同量化方式下的硬件实现中核函数使用个数 K 与单个样本处理时间之间的线性回归数学模型：

$$t_{\text{float}}=0.0364\times K+0.0692 \tag{12}$$

$$t_{\text{fixed}}=0.0009\times K+0.0744 \tag{13}$$

其中，t_{float} 与 t_{fixed} 分别为浮点数与定点数表示下硬件实现的单个样本预估处理时间，绘制的线性拟合曲线如图 6 与图 7 所示，两种量化方式下求得的决定系数 R^2 分别为 0.99716 与 0.93075。为了分析所提出的硬件实现在对大规模数据集训练时的加速表现，与现有的通过解决有约束的二次规划问题来训练 SVM 的硬件实现进行了比较，其使用的核函数均为 RBFK，如表 7 所列，其中 t_R 表示对比文献中硬件实现的单个样本处理时间。在比较分析过程中，通过将核函数使用个数规定为训练数据集的样本特征个数，对预估的单个样本处理时间的下限进行了分析。

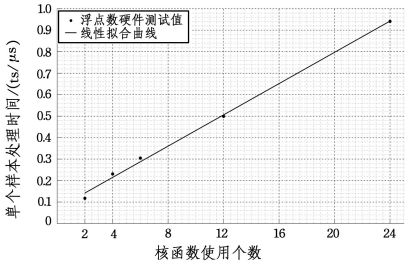

图 6 浮点数硬件实现线性拟合曲线

Fig. 6 Linear regression curve of floating point hardware implementation

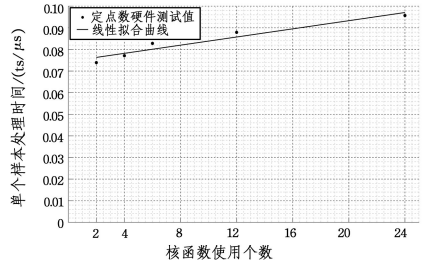

图 7 定点数硬件实现线性拟合曲线

Fig. 7 Linear regression curve of fixed point hardware implementation

文献[22]中提出了在 FPGA 中加速大规模非线性 SVM 训练的硬件实现方案，采用集成学习将训练问题划分为多个子问题进行并行处理。在对 MNIST 数据集(包含 60000 个训练样本和 784 个特征)进行测试时，其最佳 t_R 达到 50 μs。而采用本文提出的硬件实现时，t_{float} 需要 28.6068 μs，具有 1.8 倍的加速表现；t_{fixed} 需要 0.78 μs，具有 64 倍的加速表现。

文献[23]中结合最小二乘支持向量机(Least-Squares Support Vector Machine，LS-SVM)和递归神经网络的串行计算和并行传输的工作模式，提出了一种利用 FPGA 和 SVM 的图像分割硬件架构。其对维度为 294 的 45 个图像样本进行了 4225 次的迭代训练，当 FPGA 时钟频率满足要求时，预估的最佳 t_R 为 12.031 μs。对于采用本文提出的硬件实现，t_{float} 为 10.7708 μs，与对比文献所需处理时间相近；而 t_{fixed} 为 0.339 μs，具有 35 倍的加速表现。

文献[24]中提出了一种用于 SVM 增量训练的硬件并行数据流架构，为执行 KKT(Karush-Kuhn-Tucker)条件中涉及的密集型线性代数计算，使用分块矩阵算法增加并行性。其对维度为 17 的 1902 个样本进行了实验验证，结果显示 t_R 为 35.33 μs。在采用本文提出的硬件实现时，t_{float} 为 0.688 μs，具有 51 倍的加速表现；t_{fixed} 为 0.0897 μs，具有 394 倍的加速表现。

表 7 与现有硬件实现的加速性能比较

Table 7 Comparison of acceleration performance with existing hardware implementations

对比文献	硬件类型	量化方式	t_R/μs	t_{float}/μs	t_{fixed}/μs	浮点数硬件加速倍数
文献[22]	Virtex-6 FPGA	混合	50.000	28.6068	0.7800	1.8
文献[23]	Cyclone IV FPGA	浮点数	12.031	10.7708	0.3390	1.1
文献[24]	Stratix-V FPGA	浮点数	35.330	0.6880	0.0897	51.0

结束语 本文采用软硬件协同设计的方法，在基于 FPGA 的硬件平台上设计了一种用于加速 SVM 训练的可重构计算系统，并通过构建专用的并行计算结构来加速 SGD 训练 SVM 的过程。对设计的完整系统进行了软硬件联合仿真，对比了软硬件实现下的预测准确率，分析了不同量化方式对硬件资源与加速性能的影响，并评估了所提出的硬件实现的可扩展性。实验表明，所提出的设计降低了硬件实现 SVM 训练系统的复杂度，在设置合理参数的情况下，该系统可在保持模型预测准确率的同时提高计算效率，从而加速 SVM 的训练过程。所提出的设计具有良好的线性可扩展性，可适用于未来硬件资源更加充足的硬件平台，以及用于构建处理更大规模问题的分布式训练框架。接下来的研究将关注分布式计算框架中的多 FPGA 系统，并进一步优化在所提出的硬件架构上使用核方法所造成的硬件资源消耗较多的问题。

参 考 文 献

[1] KIRCHNER A，SIGNORINO C S. Using support vector machines for survey research[J]. Survey Practice，2018，11(1)：1-10.

[2] GONG Y，JIA L. Research on SVM environment performance of parallel computing based on large data set of machine learning [J]. The Journal of Supercomputing，2019，75(9)：5966-5983.

[3] FENG L，LI Z，WANG Y. VLSI design of SVM-based seizure detection system with on-chip learning capability [J]. IEEE Transactions on Biomedical Circuits and Systems，2017，12(1)：171-181.

[4] ŻUREK D，PIETROŃ M，WIATR K. Training with reduced precision of a support vector machine model for text classification[C]// Advances in Information and Communication：Proceedings of the 2021 Future of Information and Communication Conference(FICC)，Volume 2. Springer International Publishing，2021：785-798.

[5] FENG L，LI Z，WANG Y，et al. A Fast On-Chip SVM-Training System With Dual-Mode Configurable Pipelines and MSMO Scheduler[J]. IEEE Transactions on Circuits and Systems I：Regular Papers，2019，66(11)：4230-4241.

[6] NORONHA D H，TORQUATO M F，FERNANDES M A C. A parallel implementation of sequential minimal optimization on FPGA[J]. Microprocessors and Microsystems，2019，69：138-151.

[7] DASS J，NARAWANE Y，MAHAPATRA R N，et al. Distributed training of support vector machine on a multiple-FPGA system[J]. IEEE Transactions on Computers，2020，69(7)：1015-1026.

[8] DENG S，ZHAO H，FANG W，et al. Edge intelligence：The confluence of edge computing and artificial intelligence[J]. IEEE Internet of Things Journal，2020，7(8)：7457-7469.

[9] HO S M H，WANG M，NG H C，et al. Towards FPGA-assisted spark：An SVM training acceleration case study[C]// 2016 International Conference on ReConFigurable Computing and FPGAs(ReConFig). IEEE，2016：1-6.

[10] KUMAR B，VYAS O P，VYAS R. A comprehensive review on the variants of support vector machines [J]. Modern Physics Letters B，2019，33(25)：1950303.

[11] DEMIDOVA L，KLYUEVA I，SOKOLOVA Y，et al. Intellectual approaches to improvement of the classification decisions quality on the base of the SVM classifier[J]. Procedia Computer Science，2017，103：222-230.

[12] NIE F，ZHU W，LI X. Decision Tree SVM：An extension of linear SVM for non-linear classification[J]. Neurocomputing，2020，401：153-159.

[13] UTKIN L V. An imprecise extension of SVM-based machine learning models[J]. Neurocomputing，2019，331：18-32.

[14] CHAUHAN V K，DAHIYA K，SHARMA A. Problem formulations and solvers in linear SVM：a review[J]. Artificial Intelligence Review，2019，52(2)：803-855.

[15] HSU D，MUTHUKUMAR V，XU J. On the proliferation of support vectors in high dimensions[C]// International Conference on Artificial Intelligence and Statistics. PMLR，2021：91-99.

[16] SHALEV-SHWARTZ S，SINGER Y，SREBRO N. Pegasos：Primal estimated sub-gradient solver for svm[C]// Proceedings of the 24th International Conference on Machine Learning. 2007：807-814.

[17] HAJEWSKI J，OLIVEIRA S，STEWART D. Smoothed Hinge Loss and ℓ1 Support Vector Machines[C]// 2018 IEEE International Conference on Data Mining Workshops(ICDMW). IEEE，2018：1217-1223.

[18] ABEYKOON V L，FOX G C，KIM M. Performance optimization on model synchronization in parallel stochastic gradient descent based svm[C]// 2019 19th IEEE/ACM International Symposium on Cluster，Cloud and Grid Computing(CCGRID). IEEE，2019：508-517.

[19] ZOPPIS I，MAURI G，DONDI R. Kernel methods：support vector machines[M]// Encyclopedia of Bioinformatics and Computational Biology. Elsevier，2019：503-510.

[20] RING M，ESKOFIER B M. An approximation of the Gaussian RBF kernel for efficient classification with SVMs[J]. Pattern Recognition Letters，2016，84：107-113.

[21] LIM E A，TAN W H，JUNOH A K. Distance weighted K-Means algorithm for center selection in training radial basis function networks[J]. IAES International Journal of Artificial Intelligence，2019，8(1)：54.

[22] RABIEAH M B，BOUGANIS C S. FPGA based nonlinear support vector machine training using an ensemble learning[C]// 2015 25th International Conference on Field Programmable Logic and Applications(FPL). IEEE，2015：1-4.

[23] HAN L，YUE Z，GUO X. Image segmentation implementation based on FPGA and SVM[C]// 2017 3rd International Conference on Control，Automation and Robotics(ICCAR). IEEE，2017：405-409.

[24] SHAO S，MENCER O，LUK W. Dataflow design for optimal incremental svm training[C]// 2016 International Conference on Field-Programmable Technology(FPT). IEEE，2016：197-200.

PENG Weidong，born in 1968，professor. His main research interests include computer measurement and control and computer simulation.

GUO Wei，born in 1998，postgraduate. His main research interests include embedded system and machine learning acceleration.