利用数据稀疏性的LSTM加速器设计

高 琛,张 帆,高彦钊
(信息工程大学,河南郑州 450002)

摘 要: 针对长短时记忆神经网络(Long Short-Term Memory,LSTM)模型计算开销大、冗余计算较多的问题,本文提出一种利用输入数据稀疏性的LSTM加速器设计方案.本方案基于Delta网络算法,对输入序列的稀疏性进行构建,在避免数据不规则加载的前提下,对冗余矩阵向量乘法运算进行过滤;针对矩阵向量乘法计算模式进行建模,寻找最高效的并行阵列计算架构设计.在MNIST标准数据集上的实验表明,当Delta网络算法的过滤门限不超过0.5时,LSTM神经网络算法检测准确率不变,计算性能提高了21.53倍.

关键词: 长短时记忆神经网络;现场可编程逻辑门阵列;稀疏性;矩阵向量乘法
中图分类号: TP391.1 **文献标识码:** A **文章编号:** 0372-2112 (2021)02-0209-07
电子学报URL: http://www.ejournal.org.cn **DOI:** 10.12263/DZXB.20190773

Design of LSTM Accelerator by Utilizing Data Sparseness

GAO Chen,ZHANG Fan,GAO Yan-zhao
(*University of Information Engineering*,*Zhengzhou*,*Henan* 450002,*China*)

Abstract: Aiming at the problem of high computational overhead and redundancy in LSTM model,this paper proposed a design method of LSTM accelerator based on data sparsity. This scheme uses Delta network algorithm to mine the sparsity of input data,and filters the multiplication of redundant matrix vectors without irregular loading of data. The calculation mode of matrix-vector multiplication is modeled to find the most efficient parallel array computing architecture design scheme. The experimental results on MINIST dataset show that when the filter threshold of Delta network algorithm is less than 0.5,the detection accuracy of LSTM neural network algorithm remains unchanged,and the computational performance is improved by 21.53 times.

Key words: long short-term memory;field programming gate anay;sparseness;matrix-vector multiplication

1 引言

随着机器学习的不断发展,层出不穷的智能算法被应用在人工智能领域,其中比较有代表性的就是递归神经网络(Recurrent Neural Network,RNN).它凭借出色的时序学习能力,在机器学习中的应用越来越广泛,特别是在处理序列学习任务中性能更为优异,例如语音建模[1,2],机器翻译[3]和情景分析[4]等.但由于RNN本身结构缺陷,模型学习过程中存在严重的梯度消失或梯度爆炸现象,导致模型无法处理长序列学习任务,因此研究学者提出了长短时记忆神经网络[5](Long Short-Term Memory,LSTM).

LSTM通过增加门控单元控制节点间的信息流动,避免了RNN的梯度消失或爆炸现象.但门控单元的设计带来了大量复杂矩阵向量乘法运算,导致计算开销进一步增加,无法满足实时性要求.近些年,基于硬件平台对LSTM进行加速运算成为国内外学界的研究热点.常用的硬件加速器主要包括GPU、FPGA和ASIC,其中FPGA由于其特有的可配置、低功耗、可重构等性能,在LSTM硬件加速领域得到了广泛应用.但随着实际网络应用需求不断增长,数据集规模呈现海量化趋势,LSTM模型复杂性也随之增加,使得FPGA难以实时应对各种计算需求,因此在设计硬件加速器时,过滤冗余计算以降低计算时间开销显得尤为重要.

为使LSTM计算开销降低,压缩LSTM参数规模、过滤冗余计算显得十分重要,近些年来不少研究学者对此进行了研究.Li等人[6]使用块循环矩阵编码算法,对LSTM模型的权重矩阵进行了压缩重构,同时结合快

收稿日期:2019-07-08;修回日期:2019-11-11;责任编辑:王天慧
基金项目:国家自然科学基金(No.61572520);国家自然科学基金创新研究群体资助项目(No.61521003);国家核高基重大专项核高基项目(No.2017ZX01030301)

速傅里叶变换降低了计算复杂度. Kang 等人[7]提出一种负载均衡感知的网络剪枝方法,通过设置门限去除冗余节点连接,以降低权重矩阵规模. 但权重矩阵稀疏化会引起内存数据访问不规则问题,进而导致内存数据读取过程繁琐. Ali 等人[8]使用紧凑型列向量(Compressed Column Bit Vector, CCBV)稀疏索引方法,将LSTM 稀疏权重矩阵的所有非零数据映射成列参数形式,有效提升了稀疏矩阵索引效率.

为了满足 LSTM 在实际应用中的实时性需求,避免网络稀疏性引起的不规则数据加载问题,本文提出了一种基于 Delta 网络算法[9]的 LSTM 加速器设计方案. 首先,本文利用输入序列的时间依赖性,用相邻时刻向量差 $\Delta \boldsymbol{x}_t$ 构造稀疏输入序列,过滤冗余矩阵向量乘法运算,降低 LSTM 前向传输运算的计算复杂度. 其次,本文结合 FPGA 硬件资源,针对 LSTM 矩阵向量计算模式进行建模,以理论上最优的计算模式将 LSTM 进行了并行化处理. 标准数据集上进行的多次实验结果显示,本方案在保持原始 LSTM 神经网络算法识别准确率的基础上,将计算速度有效提升了 21.53 倍.

2 LSTM 模型结构

RNN 是在自然语言处理(Natural Language Processing,NLP)领域应用比较广泛的一个典型模型[10],它是一种循环连接的全连接神经网络,可以对序列数据进行学习,并保存上下文之间的联系. 但 RNN 在训练时存在严重的梯度消失或梯度爆炸现象,通过增加门控单元可以解决这个问题,模型性能也由于门控单元的加入而得到提升.

LSTM 是一种典型的门控 RNN 模型,最早由 Sepp Hochreiter 和 Jürgen Schmidhuber 于 1997 年提出,它有效解决了传统 RNN 不能建立长时间依赖及梯度消失、爆炸现象,其结构如图 1 所示. 算法扩展了一个基本的 RNN 隐层节点,增加了三个乘法单元(称为输入,输出和忘记门)用来控制信息的流入、流出及保存. LSTM 的状态更新可用式(1)~式(6)表示.

$$\boldsymbol{f}_t = \sigma(\boldsymbol{W}_{xf}\boldsymbol{x}_t + \boldsymbol{W}_{hf}\boldsymbol{h}_{t-1} + \boldsymbol{b}_f) \tag{1}$$

$$\boldsymbol{i}_t = \sigma(\boldsymbol{W}_{xi}\boldsymbol{x}_t + \boldsymbol{W}_{hi}\boldsymbol{h}_{t-1} + \boldsymbol{b}_i) \tag{2}$$

$$\widetilde{\boldsymbol{c}_t} = \tanh(\boldsymbol{W}_{xc}\boldsymbol{x}_t + \boldsymbol{W}_{hc}\boldsymbol{h}_{t-1} + \boldsymbol{b}_c) \tag{3}$$

$$\boldsymbol{c}_t = \boldsymbol{f}_t \odot \boldsymbol{c}_{t-1} + \boldsymbol{i}_t \odot \widetilde{\boldsymbol{c}_t} \tag{4}$$

$$\boldsymbol{o}_t = \sigma(\boldsymbol{W}_{xo}\boldsymbol{x}_t + \boldsymbol{W}_{ho}\boldsymbol{h}_{t-1} + \boldsymbol{b}_o) \tag{5}$$

$$\boldsymbol{h}_t = \boldsymbol{o}_t \odot \tanh(\boldsymbol{c}_t) \tag{6}$$

在 LSTM 中,忘记门 $\boldsymbol{f}_t$ 决定丢弃的历史信息程度,输入门 $\boldsymbol{i}_t$ 决定要接收的新信息程度. 通过 $\boldsymbol{f}_t$、$\boldsymbol{i}_t$ 控制信息流入,并创建新向量 $\boldsymbol{c}_t$,用于下一次计算新状态. 输出门 $\boldsymbol{o}_t$ 确定最终的输出 $\boldsymbol{h}_t$.

与传统 RNN 不同的是,LSTM 从 $t-1$ 时刻到 t 时刻,增加了门控单元来控制信息的流入与流出,并引入了向量 $\boldsymbol{c}_t$ 来保存当前状态信息,用于计算下一时刻的输出. 而传统的 RNN 总是用一个新向量来替代当前状态,并没有保留上一时刻的相关信息,这使得 LSTM 模型预测精度更准确.

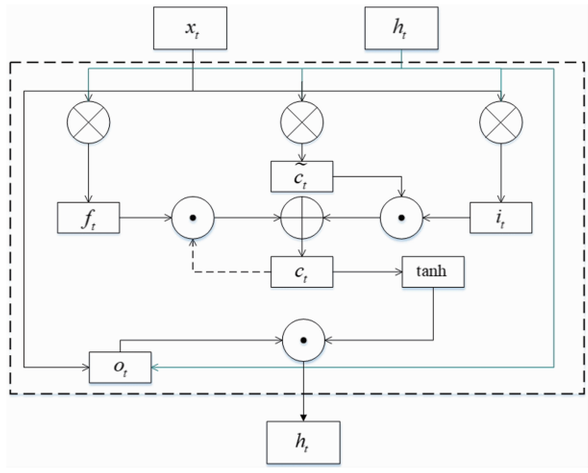


图1 LSTM模型

3 LSTM 加速算法

为了满足日益增长的性能需求,LSTM 计算所用的数据集、隐层节点规模不断增加,随之而来的大规模矩阵向量运算导致计算开销过大,这不利于 LSTM 应用于实时性要求较高的嵌入式系统中. 近些年来不少研究学者使用网络压缩算法,如稀疏矩阵、网络剪枝等,对神经网络权重参数的稀疏性进行构建,通过压缩 LSTM 规模、规避冗余计算的方式提升了 LSTM 的计算速度.

3.1 问题描述

通常情况下,使用稀疏矩阵及网络剪枝算法压缩 LSTM 规模后,权重参数矩阵分布呈不规律形式. 以 LSTM 中最典型的矩阵向量乘法为例,从式(7)稀疏权重矩阵与稠密向量计算公式可以看出,稀疏权重矩阵非零元素随机分布,这导致从内存中加载数据时,需要读取权重矩阵非零元素的大量精确行列指示信息. 另一方面,当没有对稀疏权重矩阵进行编码时,矩阵每行、每列非零元素数量也呈随机分布,导致 $\boldsymbol{W}\boldsymbol{x}_t$ 计算过程的计算量不具备一定规律性,与硬件加速平台的并行处理特性契合度较差.

$$\boldsymbol{W}\boldsymbol{x}_t = \begin{bmatrix} w_{11} & & w_{13} & & \\ & & w_{23} & & w_{51} \\ & w_{32} & & w_{34} & \\ w_{41} & & w_{43} & w_{44} & \\ & & & & w_{55} \end{bmatrix} * \begin{bmatrix} a \\ b \\ c \\ d \\ e \end{bmatrix} \tag{7}$$

为了降低 LSTM 计算开销,解决上述问题,本文从

模型计算所需的参数角度进行了分析. 一个完整的 LSTM 神经网络,其计算所需参数包括输入序列 $\boldsymbol{x}_t$ 及权重参数 $\boldsymbol{W}_i(i\in\boldsymbol{x},\boldsymbol{h})$ 两部分,其中输入序列规模往往远大于权重参数规模. 因此除了考虑压缩权重参数,挖掘输入序列稀疏性对模型计算速度提升也起着至关重要的作用.

在 LSTM 神经网络的计算过程中,零元素不影响最终计算结果. 图 2 给出了某输入序列的数值分布,零元素的占比高达 91.1%,这说明构建、挖掘输入序列的数据稀疏性,理论上也可以降低 LSTM 计算开销.

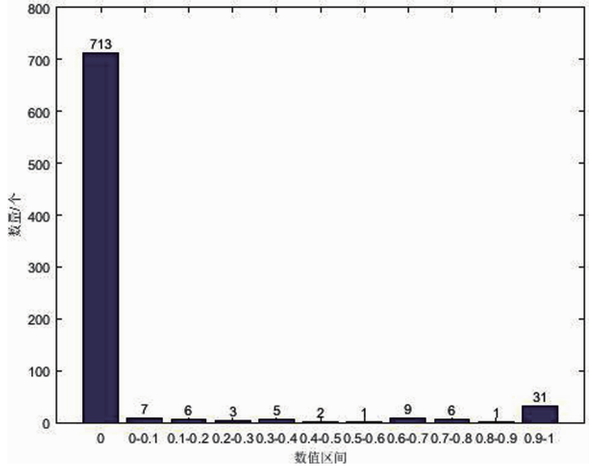


图2　某序列数据值分布

3.2　Delta 网络算法

研究[9]提出了 Delta 网络算法,通过利用输入向量的时间依赖性,用相邻时刻向量差 $\Delta\boldsymbol{x}_t$ 构造输入数据稀疏性. Delta 网络算法如式(8)与式(9)所示:

$$\boldsymbol{y}_t=\boldsymbol{W}\boldsymbol{x}_t \tag{8}$$

$$\boldsymbol{y}_t=\boldsymbol{W}\Delta\boldsymbol{x}_t+\boldsymbol{y}_{t-1} \tag{9}$$

其中 $\Delta\boldsymbol{x}_t$ 是相邻两个时刻输入到神经网络中的向量差 $\boldsymbol{x}_t-\boldsymbol{x}_{t-1}$,$\boldsymbol{y}_{t-1}$ 是上一时刻网络输出,当相邻时刻输入数据相似度较高时,通过这种方法可以构建输入数据稀疏性,避免造成数据信息的丢失.

从图 3 中可以看出,使用 Delta 网络算法构造稀疏输入序列 $\Delta\boldsymbol{x}_t$ 后,0 元素所在列的乘法运算可以直接跳过不再执行. 当输入向量的稀疏性较大时,Delta 网络算法可以大规模降低矩阵向量乘法运算的运算量及数据加载量. $\boldsymbol{W}\Delta\boldsymbol{x}_t$ 计算结果加上上一时刻的输出结果 $\boldsymbol{y}_{t-1}$,即可得到新的输出结果 $\boldsymbol{y}_t$.

3.3　Delta LSTM 构建

考虑到现有加速算法没有对 LSTM 神经网络输入序列稀疏性进行挖掘,且算法本身包含大量矩阵向量运算的特性,本文将 Delta 网络算法引入 LSTM 神经网络中,用以加速 LSTM 神经网络的计算. 使用 $\Delta\boldsymbol{x}_t$、$\Delta\boldsymbol{h}_t$ 重建后的

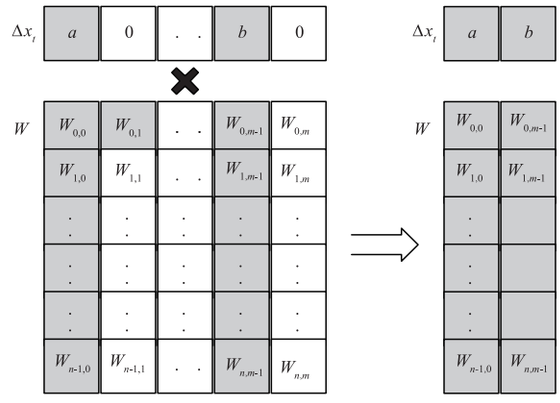


图3　Delta网络算法计算示意图

LSTM 模型推理计算如式(10)~式(15)所示:

$$\boldsymbol{f}_t=\sigma(\boldsymbol{W}_{xf}\Delta\boldsymbol{x}_t+\boldsymbol{W}_{hf}\Delta\boldsymbol{h}_{t-1}+\boldsymbol{b}_{ft-1}) \tag{10}$$

$$\boldsymbol{i}_t=\sigma(\boldsymbol{W}_{xi}\Delta\boldsymbol{x}_t+\boldsymbol{W}_{hi}\Delta\boldsymbol{h}_{t-1}+\boldsymbol{b}_{it-1}) \tag{11}$$

$$\widetilde{\boldsymbol{c}_t}=\tanh(\boldsymbol{W}_{xc}\Delta\boldsymbol{x}_t+\boldsymbol{W}_{hc}\Delta\boldsymbol{h}_{t-1}+\boldsymbol{b}_{ct-1}) \tag{12}$$

$$\boldsymbol{o}_t=\sigma(\boldsymbol{W}_{xo}\Delta\boldsymbol{x}_t+\boldsymbol{W}_{ho}\Delta\boldsymbol{h}_{t-1}+\boldsymbol{b}_{ot-1}) \tag{13}$$

$$\boldsymbol{c}_t=\boldsymbol{f}_t\odot\boldsymbol{c}_{t-1}+\boldsymbol{i}_t\odot\widetilde{\boldsymbol{c}_t} \tag{14}$$

$$\boldsymbol{h}_t=\boldsymbol{o}_t\odot\tanh(\boldsymbol{c}_t) \tag{15}$$

其中:

$$\boldsymbol{b}_t=\boldsymbol{W}_x\Delta\boldsymbol{x}_t+\boldsymbol{W}_h\Delta\boldsymbol{h}_{t-1}+\boldsymbol{b}_{t-1} \tag{16}$$

$$\boldsymbol{b}_0=\boldsymbol{b}(\boldsymbol{b}\in\boldsymbol{b}_f,\boldsymbol{b}_i,\boldsymbol{b}_c,\boldsymbol{b}_o) \tag{17}$$

从式(10)~式(15)可以看出,类比 LSTM 模型,Delta LSTM 计算所需向量由 $\boldsymbol{x}_t$、$\boldsymbol{h}_{t-1}$ 变为向量差 $\Delta\boldsymbol{x}_t$、$\Delta\boldsymbol{h}_{t-1}$,常量偏差 b 变为变量 $\boldsymbol{b}_t$,用以对序列信息进行保存. Delta LSTM 使用交叉熵损失函数用来衡量预测概率分布与真实目标之间的差异,使用梯度下降算法,通过大量训练样本对网络权值进行更新,学习率固定为 0.01.

当序列数据相似度较高时,使用 Delta 网络算法可以过滤冗余计算,降低神经网络计算开销,加速 LSTM 的运算过程. 另一方面,从图 3 中可以看出,在 Delta LSTM 算法中,权重矩阵参数以 $\Delta\boldsymbol{x}_t$ 非零元素下标为指示信息,按列参数形式进行索引,具备一定规律性,相比稀疏矩阵非零参数减少了行信息指示. 从图 3 右侧矩阵形式可以发现,经过 Delta 算法优化后,$\boldsymbol{W}\Delta\boldsymbol{x}_t$ 计算过程仍为稠密矩阵向量运算过程,比较适合在硬件加速平台上进行并行化处理.

3.4　理论分析

Delta 网络算法对输入序列相邻时刻差异性要求较高. 假设输入向量维度为 $1\times n$,输入权重矩阵维度为 $n\times n$,β、λ 为稀疏输入向量 $\Delta\boldsymbol{x}_t$、$\Delta\boldsymbol{h}_t$ 中非零元素的占比,是一个常数. 对于 LSTM 中某一个矩阵向量乘法过程 $\boldsymbol{y}_t=\boldsymbol{W}\boldsymbol{x}_t$,其计算复杂度为 $n^2+(n-1)n$,计算所需参数量为 n^2+n. 使用 Delta 网络算法后,矩阵向量乘法 $\boldsymbol{y}_t=\boldsymbol{W}\Delta\boldsymbol{x}_t+\boldsymbol{y}_{t-1}$ 的计算复杂度为 $\beta n^2+(\beta n-1)n+n$,计

算所需加载的参数量为 βn^2+2n.

当 β、λ 为 1，相比于原始矩阵向量乘法计算，Delta 网络算法 y_t 的计算复杂度增加了 n. 因此当输入序列差异性较大，即 β、λ 的值较高时，需要设置门限 θ 过滤冗余数据，构建 $\Delta \boldsymbol{x}_t$、$\Delta \boldsymbol{h}_t$ 的稀疏性，保证神经网络计算复杂度的降低. 以 $\boldsymbol{y}_t=\boldsymbol{W}\boldsymbol{x}_t$ 为例，理论上计算速度提升倍数为 $2n^2-n/2\beta n^2$，数据加载速度提升倍数为 $n^2-n/\beta n^2+2n$，当矩阵维度 n 增加的时候，计算速度及数据加载速度提升都接近于 $1/\beta$.

通过上述分析可以看出，使用 Delta 网络算法后，计算速度及数据加载速度的提升与 $\Delta \boldsymbol{x}_t$ 非零元素占比密切相关，即与输入序列数据的相似度密切相关. 随着 $\Delta \boldsymbol{x}_t$ 中非零元素占比降低，内存数据加载速度及计算速度都会得到提升.

4 设计和实现

4.1 设计

任何加速器架构都必须依赖于具体的硬件平台，例如 CPU、GPU 及 FPGA. Roofline 模型[11]常用来衡量硬件加速器与硬件加速平台的契合程度.

Roofline 模型使用计算通信比(Ratio of Computation to communication，CTC)来评估模型性能，CTC 可以使用计算量和访存量来进行计算. 其中：计算量指输入单个输入向量，模型进行一次完整的前向传输所发生的运算次数，单位是 OPs；访存量指的是输入单个输入向量，模型完成一次完整的前向传输所发生的内存访问量，单位为 Byte；使用计算量除以访存量就可以得到 CTC.

如图 4 所示，合理的加速器设计方案都要低于 CTC * 访存带宽和算力屋顶曲线中的最小值，且带宽(斜率)及 CTC(横坐标)都是影响硬件加速器性能的因素. LSTM 中矩阵计算占据大部分计算资源及能量消耗，所以加速器的性能可以由矩阵运算部分的性能来衡量.

从式(10)～式(15)可以看出，LSTM 的推理运算主要是权重 $\boldsymbol{W}_x\in\boldsymbol{R}^{N\times M}$、$\boldsymbol{W}_h\in\boldsymbol{R}^{N\times N}$ 与向量 $\Delta \boldsymbol{x}_t\in\boldsymbol{R}^{1\times M}$、$\Delta \boldsymbol{h}_t\in\boldsymbol{R}^{1\times N}$ 的乘法运算，在 FPGA 上可以使用图 5 所示的并行阵列计算架构来对复杂矩阵向量乘法运算进行加速. 为了方便对图 5 结构计算步骤进行观察，本文将此并行阵列计算架构的计算流程映射至图 6 中.

从图 6 中可以看出，每个时钟周期，并行阵列计算架构执行 $\boldsymbol{w}_{xi}\in\boldsymbol{R}^{q\times p}$、$\Delta \boldsymbol{x}_i\in\boldsymbol{R}^{1\times p}$ 的矩阵向量乘法运算，输出为 $\boldsymbol{y}_i\in\boldsymbol{R}^{q\times 1}$ 的临时向量，$\boldsymbol{y}_i$ 与先前缓存中存储的计算结果做累加运算，用以生成最终的输出向量 $\boldsymbol{y}_t$. 为了设计更高效的并行阵列计算架构，本文基于 Roofline 模型，利用单时钟周期内执行的计算量及所需加载的数据量建立 CTC 计算函数，以寻求最优的数据计算模式，即读入单位数据量可执行的计算量越多，认为并行阵列计算架构的计算模式越好. 由于 FPGA 芯片资源限制，并行阵列计算架构 p、q 的值不能无限大，文献[12]考虑了 Xilinx ZynqZc706 FPGA 板上的可用资源，给出了片上并行度上限(即可设计的并行阵列计算架构的规模)：$p\times q=128$. 从图 6 的矩阵向量计算模式我们可以得到，在单时钟周期内，图 5 所示的并行阵列计算架构执行的计算量为

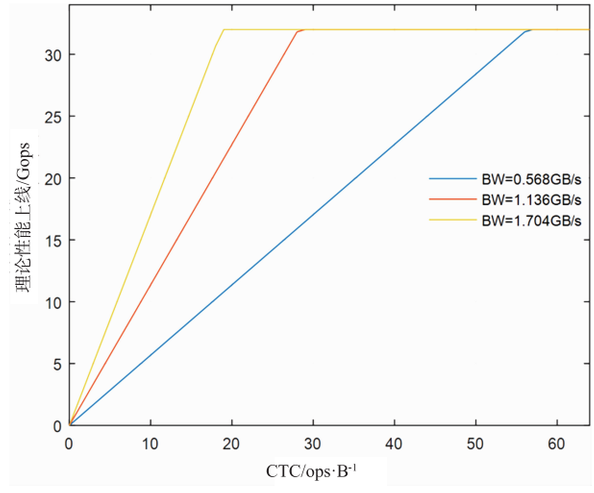


图4 Roofline模型

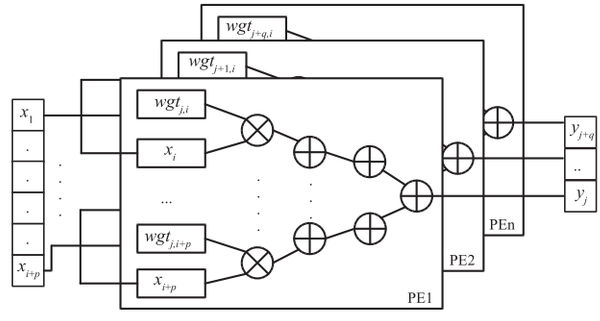


图5 并行阵列计算架构

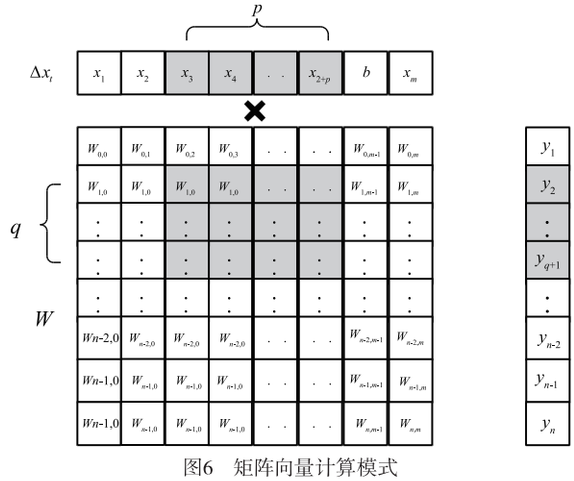


图6 矩阵向量计算模式

$$p \times q + (p-1)q + q \tag{18}$$

其中第一项为 $\boldsymbol{w}_{xi}\Delta\boldsymbol{x}_t$ 中的乘法操作次数,第二项为加法操作次数,第三项为临时计算结果 $\boldsymbol{y}_t$ 与先前累加结果的累加运算操作次数. 在单时钟周期内并行阵列计算架构计算所需加载的参数量为

$$\alpha(p \times q + p) \tag{19}$$

其中第一项为计算所需的权重数据,第二项为计算所需输入向量,α 为数据位宽(bit)转字节(B)的转化系数,是一个常量. 使用计算量除以数据量,我们可以得到并行阵列计算架构的 CTC:

$$\mathrm{CTC} = \frac{p \times q + p \times q}{\alpha(p \times q + p)} = \frac{256}{\alpha(128 + p)} \tag{20}$$

图 5 中 q 个处理单元(Process Element,PE),构成算法 I 的并行计算模式. 从公式中可以看出,随着 p 减小,CTC 的值不断变大. 当 CTC 最高时,矩阵向量计算模式为 $p=1$、$q=128$. 因此最终设计的 FPGA 矩阵向量计算模式如算法 1 所示.

算法 1　并行矩阵向量计算流程

并行矩阵向量计算模式:$p=1$,$q=128$
读入序列向量 $\boldsymbol{x}_t$
构造稀疏性 $\Delta\boldsymbol{x}_t$、$\Delta\boldsymbol{h}_{t-1}$并缓存
读入权重矩阵 $\boldsymbol{W}(\boldsymbol{W}_{\Delta f}$、$\boldsymbol{W}_{\Delta i}$、$\boldsymbol{W}_{\Delta c}$、$\boldsymbol{W}_{\Delta o})$
for $i=1$ to N loop{　//遍历 Δx_t 中的元素
for $j=1$ to $(N/128-1)$ {
　//每次以 $q=128$ 遍历矩阵 $\boldsymbol{W}_\Delta$ 的第 i 列
　//all PEs do in parallel
　$\boldsymbol{y}_t = \boldsymbol{W}_\Delta[(j\times q):(j\times q+q),i] \times \Delta\boldsymbol{x}_t[i]$
$\boldsymbol{y}_t[(j\times q):(j\times q+q)] = \boldsymbol{y}_t[(j\times q):(j\times q+q)] + \boldsymbol{y}_i$
End loop}
End loop}

4.2　实现

本文使用 Vivado HLS 2016.4,选取 Xilinx Zynq ZC706 FPGA 对基于 Delta 网络算法的 LSTM 加速器进行了综合仿真,并使用了 MNIST 数据集[13]对所设计 LSTM 加速器的基本功能进行了验证. Vivado HLS 能实现 C 代码到 RTL(寄存器传输级)级别的转换. 为了使得神经网络契合硬件平台,获取尽可能高的计算性能,LSTM 隐层节点数量取值 128,选择表 I 中的最优矩阵向量计算模式,利用 HLS 编译指令,将 $\boldsymbol{W}[j,i]$ 与 $\Delta\boldsymbol{x}_t[i]$ 的循环计算展开($j\in[1,128]$),实现 128 个 PE 并行处理,以求在 FPGA 片上资源限制下,实现矩阵向量最大并行度的计算.

加速器的峰值计算能力是衡量加速器性能一个重要指标. 在理想情况下,加速器峰值计算性能可以通过每个时钟周期可执行的计算(乘法或者加法)次数来衡量,其中矩阵向量运算部分的峰值计算量最能反应 LSTM 加速器计算性能,可由式(21)决定[14],其中 P 为并行阵列计算架构中 PEs 的数量,f 为时钟频率,系数 2 表示每个 PE 在一个时钟周期内执行一次乘法和一次加法操作.

$$PERF_{\mathrm{perk}} = 2 \times P \times f \tag{21}$$

假设时钟频率为 125MHz,不考虑带宽的限制作用,在数据能够充分加载的情况下,使用此并行阵列计算架构,理论上峰值吞吐量可达到 32GOPs(每秒完成 32G 次计算).

5　实验

为了评估 LSTM 加速器的性能,本文在 PC 端基于 Tensor Flow 搭建了标准 LSTM 模型. 系统使用 Intel CORE i5 CPU,工作频率 2GHz,内存 8GB,操作系统为 Win10 Pro. 测量算法执行时间的编译器为 spyder 3.2.4,其中配置了 Tensor Flow 开发环境用于 LSTM 模型构建及预训练.

FGPA 版本的 Delta LSTM 加速器工作时钟频率为 125MHz,θ 取值为 0,采用表 I 所示的并行矩阵向量计算模式对 $\boldsymbol{W}\boldsymbol{x}_t$ 进行加速运算.

5.1　性能评估

性能评估所用的 MNIST 数据集,是一个由 6 万张训练图片和 1 万张测试图片构成的标准手写体数字数据集,每张图片由 28×28 个像素组成. 由于图片数据按 28×28 的二维像素矩阵存储,因此在进行计算时,以行像素作为序列输入(每一行像素作为一个 $\boldsymbol{x}_t$),一张图片认为是一个 $T=28$ 的 $\boldsymbol{x}_t$ 序列. 实验首先将像素数据进行了归一化处理,其次使用训练集对模型进行预训练,最后随机选取 500 张测试集图片对所设计的算法及计算架构进行了 100 次实验评估.

我们分别统计了 100 次测试实验中 CPU 版本及 FGPA 版本的执行时间及识别准确率,结果如图 7 所示. 可以看出,处理同样 500 张图片序列数据,使用 Delta 网络算法简化后的 LSTM 加速器,平均检测准确率在 97% 左右,与原始 LSTM 神经网络算法持平,这意味着 Delta 网络算法并没有降低 LSTM 神经网络在 MNIST 标准数据集上的性能. 由于每次测试所用图片都是随机选取,所以输入序列相邻时刻的非零元素占比不同,导致在 100 次实验中,检测准确率及计算时间会有些许波动.

从图 7 中对计算时间的统计可以明显看出,相比于 CPU 版本的标准 LSTM 神经网络,处理 500 张图片像素数据,FPGA 版本的 Delta LSTM 神经网络所消耗的时间仅仅只有大约 3.6ms 左右,计算速度提升了约 21.53 倍. 这主要是因为 MNIST 数据集的数据相似度高,使得 Delta 网络算法可以构造具备较高稀疏性的 $\Delta\boldsymbol{x}_t$,以过滤

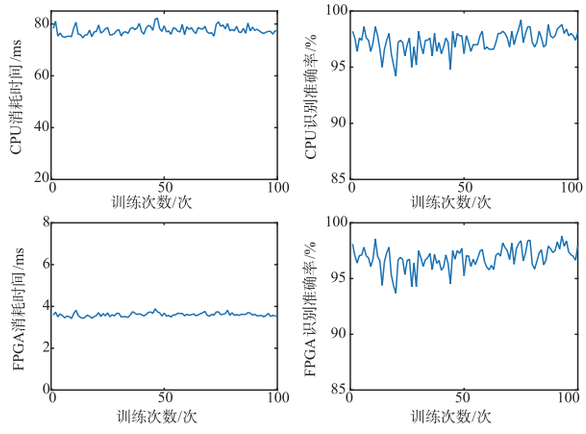


图7 计算时间及准确率对比，增大纵坐标

冗余矩阵向量运算. 另一方面由于本文设计了并行矩阵向量计算模式，充分利用 FPGA 片上资源将 LSTM 循环迭代的矩阵计算进行了并行化处理，以资源消耗为代价提高了模型计算性能.

在 125MHz 时钟频率下的实测结果显示，LSTM 加速器实际平均计算性能约为 3.484GOPs，处于 Roofline 模型的带宽瓶颈区域，远低于理论计算性能 32GOPs，这主要是由于 FPGA 实验板带宽限制，导致数据加载速度无法匹配并行阵列计算架构的计算速度；另一方面由于 LSTM 循环连接的特性，$\boldsymbol{h}_t$ 的计算必须依赖于上一时刻 $\boldsymbol{h}_{t-1}$ 的计算结果，这间接降低了 LSTM 前向传输运算的计算速度.

5.2 检测率评估

为了评估门限值对检测准确率的影响，本文通过设置不同的 θ，对输入序列 $\Delta\boldsymbol{x}_t$ 中的冗余数据进行过滤，以构造不同程度序列稀疏性，实验结果如图 8 所示.

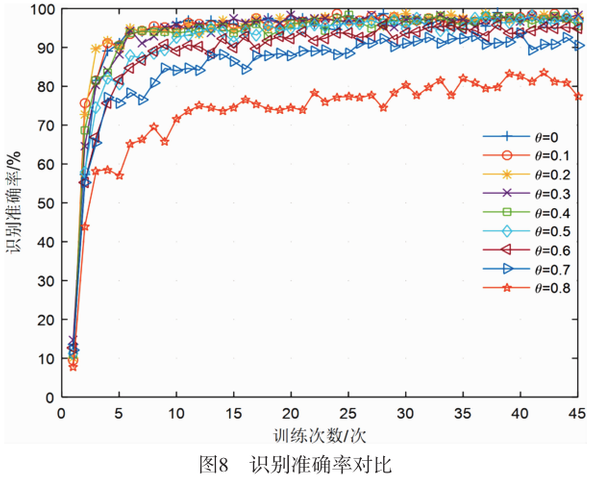


图8 识别准确率对比

由于所训练的模型在第 45 次训练以前就已经趋于稳定，为了便于区分不同门限值 θ 对模型性能的影响，图 8 只显示前 45 次模型训练后的实验结果. 从图中可以看出，当门限值 θ 取值小于 0.5 时，Delta LSTM 神经网络对 MNIST 测试集图片的检测准确率几乎没有降低；当门限值 θ 取值 0.5 时，模型收敛速度开始降低，但平稳后的模型检测准确率并没有呈现明显降低趋势；当 θ 取值大于 0.5 时，模型收敛速度及检测准确率都有所降低.

从图 8 中可以看出，只要选择合适的门限值 θ 对输入序列进行过滤，并不会降低 LSTM 神经网络的性能. 另一方面门限值 θ 的引入还可以提高输入序列的稀疏性，对降低 LSTM 神经网络的计算开销具有重要作用.

6 总结

本文提出了一种基于输入数据稀疏性的 LSTM 加速器设计方法，在避免参数不规则加载的前提下，对 LSTM 前向传输运算进行了加速. 本文首先利用相邻时刻的向量差 $\Delta\boldsymbol{x}_t$ 构造出输入数据的稀疏性，以简化矩阵向量乘法运算，降低 LSTM 前向传输运算的计算复杂度；其次结合 Roofline 模型，设计了高效的矩阵向量加速计算方案，对 LSTM 进行了并行化处理. 在 MNIST 标准数据集上的实验结果表明，本方案可以在不降低原始 LSTM 神经网络算法检测准确率的前提下，将计算速度有效提升 21.53 倍；当过滤门限小于 0.5 时，Delta LSTM 几乎不会对模型的识别准确率造成影响.

本文基于 MNIST 数据集，对 Delta LSTM 模型的计算性能及检测准确率进行了实验评估，验证了模型在图像分类场景中应用的准确性. 为了评估 Delta LSTM 模型在其他场景中的适用性，下一步本文拟基于音频、文本等数据集，对 Delta LSTM 模型性能进行实验验证.

参考文献

[1] SAK H, SENIOR A, BEAUFAYS F. Long short-term memory based recurrent neural network architectures for large vocabulary speech recognition[J]. arXiv Preprint, 2014, arXiv:1402.1128.

[2] MIKOLOV T, KARAFIÁT M, BURGET L, et al. Recurrent neural network based language model[A]. Eleventh Annual Conference of the International Speech Communication Association[C]. Chiba, Japan: ISCA. 2010. 1045 – 1048.

[3] CHO K, VANMerriënboer B, GULCEHRE C, et al. Learning phrase representations using RNN encoder decoder for statistical machine translation[J]. arXiv Preprint, 2014, arXiv:1406.1078.

[4] BYEON W, BREUEL T M, RAUE F, et al. Scene labeling with LSTM recurrent neural networks[A]. Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition[C]. Boston, Massachusetts, USA: IEEE. 2015. 3547 – 3555.

[5] HOCHREITER S, SCHMIDHUBER. J. Long short-term memory [J]. Neural Computation, 1997, 9(8): 1735 – 1780.

[6] LI Z, WANG S, DING C, et al. Efficient recurrent neural networks using structured matrices in FPGAs [J]. arXiv Preprint, 2018, arXiv: 1803.07661.

[7] HAN S, KANG J, MAO H, et al. Ese: Efficient speech recognition engine with sparse LSTM on FPGA[A]. Proceedings of the 2017 ACM/SIGDA International Symposium on Field-Programmable Gate Arrays [C]. Monterey, California, USA: ACM, 2017. 75 – 84.

[8] ALI S M, WANG S J, MA N, et al. A bandwidth in-sensitive low stall sparse matrix vector multiplication architecture on reconfigurable FPGA platform[A]. 2017 13th IEEE International Conference on. IEEE[C]. Yangzhou, China: Electronic Measurement & Instruments (ICEMI), 2017. 171 – 176.

[9] NEIL D, LEE J H, DELBRUCK T, et al. Delta networks for optimized recurrent network computation[A]. Proceedings of the 34th International Conference on Machine Learning [C]. Australia: JMLR, 2017. 2584 – 2593.

[10] HOPFIELD J J. Neural networks and physical systems with emergent collective computational abilities[J]. Proceedings of the National Academy of Sciences, 1982, 79(8): 2554 – 2558

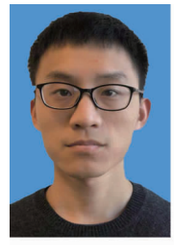
[11] Williams S, Waterman A, Patterson D A, et al. Roofline: An insightful visual performance model for multicore architectures[J]. Communications of the ACM, 2009, 52(4): 65 – 76.

[12] AYAT S O, MOHAMED K-H, AB R A A H. Optimizing FPGA-based CNN accelerator for energy efficiency with an extended Roofline model[J]. Turkish Journal of Electrical Engineering & Computer Sciences, 2018, 26(2): 919 – 935.

[13] LECUN Y, BOTTOU L, BENGIO Y, et al. Gradient-based learning applied to document recognition[J]. Proceedings of the IEEE, 1998, 86(11): 2278 – 2324.

[14] 宋翔,周凡,陈耀武,等. 基于 FPGA 的实时双精度浮点矩阵乘法器设计[J]. 浙江大学学报(工学版), 2008, 42(9): 1611 – 1615.
SONG X, ZHOU F, CHEN Y W, et al. Design of real time double precision floating point matrix multiplier based on FPGA[J]. Journal of Zhejiang University, 2008, 42(9): 1611 – 1615. (in Chinese)

作者简介



高　琛　男, 1994 年出生, 山东济南人. 2017 年毕业于电子科技大学电子工程学院, 同年进入国家数字交换系统工程技术研究中心攻读硕士学位. 主要从事 FPGA 硬件加速及人工智能芯片的有关研究.
E-mail: 616414829@qq.com

张　帆　男, 1981 年出生, 河南郑州人, 博士、硕士生导师. 分别于 2004 年、2007 年和 2012 年在国家数字交换系统工程技术研究中心获学士、硕士及博士学位, 现为国家数字交换系统工程技术研究中心副研究员. 主要研究方向为主动防御、芯片设计技术、高性能计算.
E-mail: 13838267352@qq.com

高彦钊　男, 1984 年出生, 河北平山人, 博士. 2007 年在北京航空航天大学获学士学位, 2009 年及 2014 年在国防科技大学获硕士、博士学历, 现为国家数字交换系统工程技术研究中心助理研究员. 主要研究方向为高性能计算.
E-mail: buaagaoyz@sina.com