中国科学院大学

University of Chinese Academy of Sciences

# 博士学位论文

## 大规模脉冲神经网络的转化算法和硬件实现研究


作者姓名：张磊

指导教师：郭崎 研究员 中国科学院计算技术研究所


学位类别：工学博士

学科专业：计算机系统结构

培养单位：中国科学院计算技术研究所

2020 年 6 月

# Research on Conversion Algorithms and Hardware Implementation of Large-scale Spiking Neural Network

A dissertation submitted to the

University of Chinese Academy of Sciences

in partial fulfillment of the requirement

for the degree of

Doctor of Philosophy

in Computer Architechture

By

Zhang Lei

Supervisor: Professor Guo Qi

Institute of Computing Technology, Chinese Academy of Sciences

June, 2020

## 中国科学院大学
## 学位论文原创性声明

本人郑重声明：所呈交的学位论文是本人在导师的指导下独立进行研究工作所取得的成果。尽我所知，除文中已经注明引用的内容外，本论文不包含任何其他个人或集体已经发表或撰写过的研究成果。对论文所涉及的研究工作做出贡献的其他个人和集体，均已在文中以明确方式标明或致谢。本人完全意识到本声明的法律结果由本人承担。

作者签名：

日　　期：2020.5.24

## 中国科学院大学
## 学位论文授权使用声明

本人完全了解并同意遵守中国科学院大学有关保存和使用学位论文的规定，即中国科学院大学有权保留送交学位论文的副本，允许该论文被查阅，可以按照学术研究公开原则和保护知识产权的原则公布该论文的全部或部分内容，可以采用影印、缩印或其他复制手段保存、汇编本学位论文。

涉密及延迟公开的学位论文在解密或延迟期后适用本声明。

作者签名：　　导师签名：

日　　期：2020.5.24　　日　　期：2020.5.24

# 摘 要

脉冲神经网络被学术界誉为第三代神经网络，以其在空间和时间信息处理上的出色能力以及对生物神经网络的模拟能力闻名。但是脉冲神经网络及其硬件应用一直存在训练难、精度低、扩展性差等问题，这些问题限制着SNN应用范围的扩展。解决此类问题将会推动脉冲神经网络算法的发展，同时以低开销著称的脉冲神经网络也将启发低开销的神经网络加速器设计。为此，学术界开始研究将卷积神经网络转化为脉冲神经网络的转化算法，来解决上述问题，并开始在小规模网络上初见成效。本文将研究基于频率编码的大规模脉冲神经网络的转化算法，并在此基础上研究基于时间编码的转化算法，以大幅降低神经网络运算开销。本文的目标是为脉冲神经网络构建高精度、低开销的转化算法，并探索更低硬件开销的神经网络处理器设计。本文的主要贡献如下：

1.研究大规模脉冲神经网络高准确度的算法可行性。本文之前的研究大多关注小规模脉冲神经网络算法的设计，其应用范围一般分布在小规模的智能任务上。另外已有算法本身具有很多影响精度的因素，例如阈值不稳定带来的精度波动、编码方式中的随机效应带来的精度不稳定性以及缺乏稳定有效的阈值策略等。这些影响算法精度的因素都给算法精度评估带来了不稳定性，同时也给大规模脉冲神经网络的高精度实现带来了不确定性。本文在前人提出的转化算法基础上，提出了用于提升大规模脉冲神经网络算法精度的转化算法 DSNN，并实验验证了算法在大规模网络中的有效性。针对之前脉冲神经网络算法的精度不稳定特性，提出了改进的脉冲神经网络计算机制 DSNN-fold。在该机制下神经网络的前向传播过程计算稳定性不再受到阈值不确定性的影响，同时精度损失控制在了1%以内。与此同时，新的算法在硬件上也具有更高的可扩展性。

2.基于时间编码的大规模脉冲神经网络运算开销优化。经过对DSNN系列算法进行理论评估发现，基于该算法的脉冲神经网络计算开销远远大于同等规模的卷积神经网络。因而针对做出的硬件部署和加速器设计将不会取得明显的收益。本文分析得出影响运算开销的一个关键原因在于脉冲神经网络的编码方式。通过利用时间编码替换频率编码并引入带泄露的累积放电模型，本文建立了基于时间编码的脉冲神经网络前向传播机制和编码方案，并在此基础上构建了转

化算法TDSNN。实验证明TDSNN算法能够获得与转化前卷积神经网络相同水平的精度，并实现平均40%的运算量降低，其神经元存储开销降低为16位定点量化后的卷积神经网络的四分之一。

3.基于TDSNN算法的神经网络加速器设计。鉴于TDSNN算法在精度、硬件开销等关键要素上都优于现有的卷积神经网络，本文研究该算法能否为神经网络加速器设计带来新的设计思路。通过对算法进行部署方案的分析，本文得出加速器设计的核心点在于运算单元的设计。本文选择搭载完备指令集Cambricon的架构为参考基准，并研究了适配TDSNN算法的六种运算单元实现方案。通过对TDSNN算法应用前后的方案的实验对比分析，本文选择其中能耗收益最明显的方案为最终方案。本文的加速器设计方案在维持了和原有Cambricon设计下相同水平的加速比的条件下，能够实现在大规模网络上至少32.26%的能耗降低，极端参数下甚至可以达到44.80%的能耗降低。

经过本文的研究，基于频率和时间编码的大规模脉冲神经网络算法已经成为可能，同时高效的时间编码脉冲神经网络算法也将推动设计更低开销的神经网络加速器。

**关键词：** 脉冲神经网络，转化方法，频率编码，时间编码，神经网络加速器

# Abstract

Spiking Neural Network(SNN) is hailed as the third generation neural network by academia, known for its excellent ability in space and time information processing and simulation ability of biological neural network. However, SNN and its hardware application have always had difficulties in training, low accuracy, and poor scalability. These problems limit the expansion of SNN's application range. Solving such problems will promote the development of SNN algorithms. At the same time, SNN with low overhead will also inspire the design of low-overhead neural network accelerators. To this end, academia began to study the conversion algorithm of convolutional neural network into SNN to solve the above problems, and began to show initial progress on small-scale networks. This paper will study the conversion algorithm of large-scale SNN based on frequency coding, and on this basis, study the conversion algorithm based on time coding to greatly reduce the computational cost of neural network. The goal of this paper is to build a high-precision, low-overhead conversion algorithm for SNN, and to explore neural network processor designs with lower hardware overhead. The main contributions of this article are as follows:

1. Study the feasibility of high-precision algorithm of large-scale spike neural network. Most of the previous research in this paper focused on the design of small-scale spike neural network algorithms, and its application range is generally distributed on small-scale intelligent tasks. In addition, existing algorithms have many factors that affect accuracy, such as accuracy fluctuations caused by unstable thresholds, instability of accuracy caused by random effects in encoding methods, and the lack of stable and effective threshold strategies. These factors that affect the accuracy of the algorithm all bring instability to the evaluation of the algorithm accuracy, and also bring uncertainty to the high-precision implementation of large-scale spike neural networks. In this paper, based on the conversion algorithm proposed by the predecessors, a conversion algorithm DSNN for improving the accuracy of the large-scale spike neural network algorithm is proposed, and the effectiveness of the algorithm in large-scale networks

is experimentally verified. Aiming at the unstable characteristic of the previous spike neural network algorithm, an improved spike neural network computing system DSNN-fold is proposed. Under this mechanism, the computational stability of the neural network's forward propagation process is no longer affected by the uncertainty of the threshold, and the accuracy loss is controlled within 1%. At the same time, the new algorithm also has higher scalability in hardware.

2. Optimization of operation cost of large-scale impulse neural network based on temporal coding. After theoretical evaluation of the DSNN algorithms, it is found that the computational cost of the spike neural network is much greater than that of convolutional neural networks of the same size. Therefore, the hardware deployment and accelerator design will not achieve obvious benefits. This paper analyzes that a key reason that affects the operation cost is the encoding method of the spike neural network. By using temporal coding instead of frequency coding and introducing a leaky integrate-and-fire model, this paper establishes a spike neural network forward propagation mechanism and coding scheme based on temporal coding, and builds a conversion algorithm TDSNN on this basis. Experiments prove that the TDSNN algorithm can obtain the same level of accuracy as the convolutional neural network before conversion, and achieve an average reduction of 40% of the amount of calculation.

3. Design of neural network accelerator based on TDSNN algorithm. In view of the fact that the TDSNN algorithm is superior to the existing convolutional neural network in terms of accuracy, hardware overhead and other key elements, this paper studies whether the algorithm can bring new design ideas to neural network accelerators. Through the analysis of the deployment plan of the algorithm, the core point of the design of the accelerator is the design of the computing unit. In this paper, the architecture of Cambricon equipped with a complete instruction set is selected as a reference benchmark, and six implementations of the arithmetic unit adapted to the TDSNN algorithm are studied. Through the experimental comparative analysis of the schemes before and after the application of the TDSNN algorithm, this paper selects the scheme with the most obvious energy consumption benefit as the final scheme. The accelerator design scheme in this paper can achieve at least 32.26% energy consump-

tion reduction on large-scale networks while maintaining the same level of acceleration ratio as the original Cambricon design, and even achieve 44.8% energy consumption reduction under extreme parameters. .

After the research in this paper, large-scale spike neural network algorithms based on frequency and temporal coding have become possible. At the same time, spike neural network algorithms with efficient temporal coding will also promote the design of lower-cost neural network accelerators.

**Keywords:** spike neural network，conversion methods，rate coding，temporal coding，neural network accelerator

# 目 录

# 图形列表

# 表格列表

# 第1章 引言

神经网络的研究可以根据研究思路分为两个方向：一个是以模拟人脑为主要思路设计类似于人脑的更高效神经网络的研究，即脉冲神经网络（Spiking Neural Network，缩写SNN）的研究；另外一个是以计算机科学为基础受生物学启发而诞生的人工神经网络（Artificial Neural Network，缩写ANN）的研究。到目前为止，两个方向的研究在实际应用上的效果差距巨大。人工神经网络随着深度学习的演进，已经逐步投入到了大规模的工业级应用中，这其中卷积神经网络（Convolutional neural Network，缩写CNN）的影响力尤为巨大。其对应的加速器设计的研究也已经被应用到工业上，进一步推动了多种智能任务的落地应用。反观脉冲神经网络的研究，目前仍然受到规模、精度、速度、硬件开销等因素的制约，仍然处于亟需解决这些问题的重要阶段。如果说早期类脑科学和脉冲神经网络的研究推动了卷积神经网络的发展，那么现如今成果颇丰的卷积神经网络能否也启发构建更高效的脉冲神经网络？基于此，本文从实现脉冲神经网络的算法中最具潜力的转化算法入手，在算法上分别对大规模脉冲神经网络转化算法做出了深入的研究，并借助转化算法的优势实现更高效的神经网络加速器。

本章的组织如下：1.1节简述了本文的研究背景；1.2节介绍了本文研究的意义；1.3节交代了本文的研究路线；1.4节展示了本文的研究成果；1.5节介绍了本文的章节组织和内容。

## 1.1 研究背景

本节将分别从算法和硬件两个角度，介绍本文的研究之前学术界在脉冲神经网络和卷积神经网络上研究的背景。

自深度学习诞生起至今，以卷积神经网络为代表的人工神经网络，在诸多复杂的智能任务中取得了突破性的进展，例如计算机视觉领域的图像识别任务Large Scale Visual Recognition Challenge （LSVRC[4]）。如图1.1所示，卷积神经网络（Convolutional Neural Network，缩写CNN）为代表的人工神经网络，其识别精度在近几年逐步提升并超过了人类的识别精度。2012年的 Alexnet

网络[4]以16.4%的错误率推动了卷积神经网络的发展。自此，卷积神经网络经历了2013年错误率为11.2%的 Clarifia 网络[5]，以及2014年的两个非常经典的网络GoogLeNet[6]与VGG-16[7]，到了2015年终于诞生了超越人类识别精度的网络ResNet[8]。网络错误率降低到了3.57%，低于了人类的5%的识别错误率。反观脉冲神经网络算法的发展，在本文的研究之前，直接在ImageNet任务上有明显的成效的要属在2016年Eric的工作[9]，它的工作使用了转化算法结合带泄漏的累积放电神经模型的特性在ImageNet上实现了23.8%的错误率效果。但是基于该工作的神经网络精度与主流卷积神经网络的精度仍然存在差距。

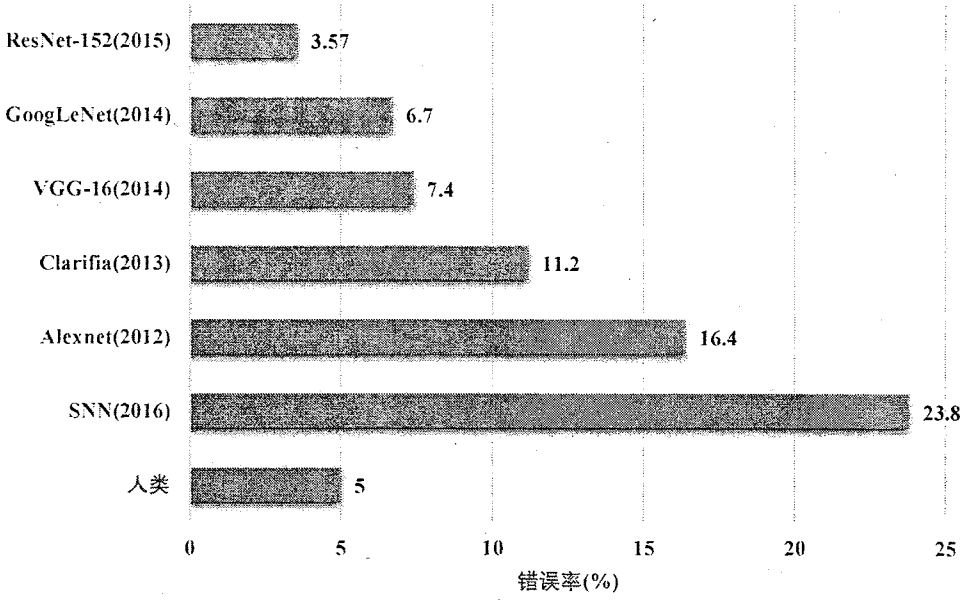


**图 1.1 ImageNet识别任务上精度的发展**

**Figure 1.1 Development of the accuracy on the ImageNet classification challenge**

那么为何脉冲神经网络算法发展至今，没有在类似ImageNet复杂度的任务上有媲美卷积神经网络的表现呢？其本质原因在于脉冲神经网络缺乏行之有效的训练算法。脉冲神经网络发展至今诞生了层出不穷的训练算法，考虑到大部分算法的结果都是在较为简单的数据集例如MNIST[10]上做实验评估，因此此处将每种特定类型的算法在MNIST数据集上的效果呈现在表1.1中。可见不管是由生物启发得到的STDP训练机制，还是直接使用生物学模型结合STDP机制，都很难在MNIST数据集上实现超过99%的精度效果。另外，少数工作参考人工神经网络的反向传播算法，研究得到了适用于脉冲神经网络的训练算法。这些

工作中SNN在精度上能够实现接近99%的精度。在众多的训练算法中，本文发现转化算法具有进一步研究的价值。转化算法已经能够在MINIST数据集上实现99.42%的精度，这与卷积神经网络的最优精度处于相同的水平。这说明转化算法是目前众多算法中唯一一个精度损失相对较小的算法。转化算法的根本目的是通过对已有CNN网络结构和运算的微调整并利用原有的训练算法完成训练来获得神经网络参数模型，参数模型会被进一步移植到对应结构的SNN中。那么转化算法能否在大规模的深层网络中成功实践？本文的第一个研究内容里试图回答该问题，并为大规模脉冲神经网络构建高效稳定的转化算法。本文希望借助转化算法，促使脉冲神经网络的精度达到主流卷积神经网络的水平。

**表 1.1 多种训练方法下的SNN在MINIST数据集上的精度[3]**

**Table 1.1 Accuracy of SNN on MINIST dataset based on various training methods[3]**

| 模型 | 训练方法 | 准确度(%) |
|---|---|---|
| Lee(2016)[11] | Backpropagation | 98.88 |
| Diehl (2015)[12] | STDP (2-layer) | 95.00 |
| Tavanaei(2017)[13] | STDP-based backpropagation (3-layer) | 97.22 |
| Zhao(2015)[14] | Tempotron | 91.29 |
| Zeng(2018)[15] | Equilibrium learning+STDP | 98.52 |
| Esser(2015)[16] | Conversion | 99.42 |

除却算法部分的发展，以卷积神经网络为代表的人工神经网络与脉冲神经网络在硬件领域的发展程度和路线也大不相同。表1.2中展示了近些年来两个方向上的具有代表性的硬件及其参数情况。对于人工神经网络硬件，其设计需要综合考虑其应用范围、可扩展性、面积、功耗等多种因素。DianNao[17]能够完成主流的神经网络运算，如CNN、DNN 等。PuDianNao[18]能够处理包括KNN、K-Means在内的多种复杂机器学习任务，而不局限于正常的神经网络运算。ShiDianNao[19]是面向嵌入式设备的卷积神经网络处理器，它采用脉动阵列的形式完成神经网络的卷积运算，从而提高数据复用，减少访存能耗，实现端到端的神经网络应用。Cambricon-S[20]利用了卷积神经网络中的剪枝技术针对剪枝后的神经网络设计加速硬件。可见人工神经网络加速器伴随卷积神经网络算法的发展也已经发展到非常成熟的状态，并且硬件的发展也促进了一批算法上的革新，例如稀疏化和量化方法的研究。

反观SNN硬件的发展，其中具有代表性硬件是TrueNorth[21]和Loihi[22]。这两个硬件都采用了大规模计算核心互联网络，搭建了由许多脉冲神经网络处理器核构成的加速器。其中它们还对现有的部分SNN训练算法例如STDP（Spike Timing Dependnt Plasticity）做了支持，也即支持片上的在线训练。通过对比硬件参数不难发现，在芯片面积上SNN硬件的面积远大于现有的卷积神经网络芯片。原因主要在于脉冲神经网络的整体运算特性导致了算法在硬件部署时难以做拆分。而映射整个层甚至整个网络就需要大量的计算核心。现有的SNN算法构建的网络大多具有规模小的特点，因而能够完成硬件的部署。但是随着SNN应用范围的扩大，为了满足复杂处理任务的需求，必然需要构建更大规模脉冲神经网络，这将会对目前的脉冲神经网络硬件的设计思路产生冲击。另外一个创新性的研究是Tianjic芯片[2]，它的设计思想是让芯片同时支持SNN算法和现有的主流ANN算法，包括CNN和RNN等。但是该芯片的实验评估网络也大多局限在多层感知机的水平，还远未达到深层神经网络的规模，因此其能否处理像ImageNet这样复杂的任务仍然是存疑的。

**表 1.2 代表性的神经网络硬件**

**Table 1.2 Representative neural network hardware**

| 硬件 | 应用 | 工艺(nm) | 面积($mm^2$) | 功耗(W) |
|---|---|---|---|---|
| DianNao[17] | neural networks | 65 | 3.02 | 0.485 |
| PuDianNao[18] | machine learning | 65 | 3.51 | 0.596 |
| ShiDianNao[19] | CNN | 65 | 4.86 | 0.32 |
| Cambricon-S[20] | sparse neural networks | 65 | 6.82 | 0.821 |
| TrueNorth[21] | SNN | 28 | 430 | 0.3 |
| Loihi[22] | SNN | 14 | 60 | NA |
| Tianjic[2] | SNN+ANN | 28 | 14.44 | 0.95 |

2017年Du等人[23]开展了对硬件实现脉冲神经网络算法和人工神经网络算法比较的研究，他们的工作表明对脉冲神经网络硬化后的硬件在面积、功耗、能耗等多种硬件参数上相比人工神经网络硬件没有明显优势。该工作主要通过对当时主流的脉冲神经网络算法的硬件实现进行对比，来得出上述的实验结论。因此即使在大规模脉冲神经网络精度可行性得到证明的基础上，也不太建议直接针对该算法设计专有的硬件，需要优先解决脉冲神经网络的运算开销问题。

学术界也试图从脉冲神经网络的时间编码方式入手降低脉冲神经网络的运算开销。比较有代表性的工作是Rueckauer[24]在2018年的工作，它建立了一种时间编码方式的脉冲神经网络转化算法，然而在MINIST数据集上的实验效果并未达到基于频率编码的SNN转化算法的水平。不难看出时间编码的算法研究难度更高，但一旦算法上的研究成功将进一步降低SNN的运算开销，并且在编码方式上推动时间编码方式的研究进展。

## 1.2 研究意义

脉冲神经网络具备低运算开销的潜力，但是其存在精度低、难训练、扩展性差等问题，受制于算法上的研究。本文的研究致力于解决脉冲神经网络的已有问题，其研究意义如下：

1.建立大规模脉冲神经网络转化算法，提高脉冲神经网络的实用价值；

2.提取脉冲神经网络低开销运算特性，设计更低开销的神经网络加速器。

## 1.3 研究路线

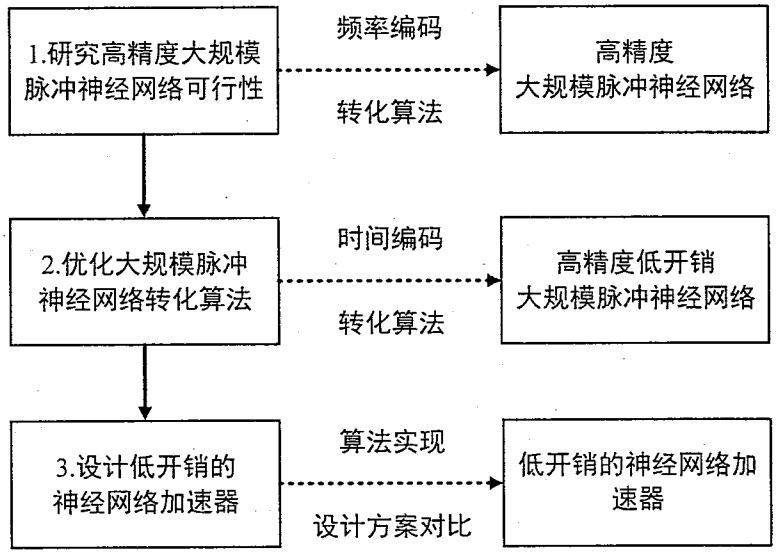


**图 1.2 研究路线图**

**Figure 1.2 Research Roadmap**

本文的研究路线共分为三步，如图1.2所示。

第一步，进行大规模脉冲神经网络精度可行性的研究。本文首先以基于频率编码的脉冲神经网络为研究对象，研究将卷积神经网络转化为脉冲神经网络的转化算法。通过建立有效的转化算法，可以使得转化得到的脉冲神经网络达到转化前的卷积神经网络相同的精度水平。从而验证了高精度大规模脉冲神经网络的可行性。

第二步，针对大规模脉冲神经网络的运算开销进行优化。本文通过替换频率编码为时间编码的方式，建立新编码方式下的转化算法。利用时间编码的优势，新构建的转化算法能够将卷积神经网络转化为高精度、低运算开销的脉冲神经网络。

第三步，设计低运算开销网络的加速器。通过对第二步得到的低运算开销网络进行算法部署方案的分析，本工作挖掘其运算优势并将其应用到神经网络加速器的设计中，以希望降低神经网络加速器的开销。

## 1.4 研究成果

本文的主要研究成果如下：

第一个工作通过对大规模脉冲神经网络精度可行性的研究，建立了适用于大规模脉冲神经网络的转化算法DSNN，转化精度损失控制在了6%以内。在此基础上本文对影响脉冲神经网络精度的参数进行实验分析，得到最优的频率编码策略为均匀分布编码，最优的输出判决策略为最大累积电位判决。与此同时，本文以软硬件协同的方式优化脉冲神经网络的前向计算过程。通过对神经网络可折叠特性的分析，本文建立了基于DSNN的新前向计算方法DSNN-fold。该方法使得脉冲神经网络的精度进一步提升，转化精度损失控制在了4%以内。DSNN-fold还能够降低阈值等参数的确定难度，进一步提高了脉冲神经网络的精度稳定性。

第二个工作是在DSNN和DSNN-fold算法的基础上，通过对脉冲神经网络的运算开销进行实验分析，得出影响其表现的关键点在编码方式上。本文引入了时间编码方式，建立了该编码方式下的前向传播机制和转化算法TDSNN。TDSNN算法得到的脉冲神经网络能够取得和卷积神经网络相同的识别精度，其运算开销能够获得至少40%的降低。通过进一步限制时间编码的时间窗口大小，本文实现了神经数据的压缩存储，相比于主流16位定点的神经元量化方法，存

储开销降低到了其四分之一。

第三个工作承接TDSNN算法的工作，进行了算法硬件部署的研究。本文挖掘了TDSNN算法实现运算开销降低的关键点，对其中关键运算层的硬件实现进行了深入分析，得出硬件设计的关键点在于运算单元设计。本文以多种不同的设计角度研究运算单元的设计方案，得到了多种有效的算法硬件部署方案。其中能耗收益最高的方案能够实现和参考基准设计相同水平的加速比，且实现了至少32.26%的能耗降低，极端参数下甚至可以达到44.80%的能耗降低。本文的工作表明，脉冲神经网络转化算法的研究有效地推动了低开销神经网络加速器的发展。

## 1.5 本文的结构安排

本文共包括七章，具体组织如下：

第一章介绍了脉冲神经网络领域的研究情况，通过与以卷积神经网络为代表的人工神经网络进行对比，对其中的算法和硬件研究现状进行了详细地介绍，并指出了当前的脉冲神经网络转化算法为最有潜力的研究方向。此外，本章也展示了脉冲神经网络转化算法研究的意义。最后，本章介绍了本文的主要研究路线和成果。

第二章介绍了脉冲神经网络的基本背景和本文的相关技术。考虑到脉冲神经网络转化算法既涉及到了转化前的人工神经网络，又涉及到了转化后的脉冲神经网络。因此本章对两个领域的研究背景进行了详细的对比分析，并对算法中涉及到的常见技术进行了介绍。考虑到算法最终能够带来神经网络加速器的设计的优化，本章还对神经网络加速器的研究背景进行了基本的介绍。

第三章介绍了基于频率编码的大规模脉冲神经网络转化算法的研究。我们首先建立大规模脉冲神经网络的基本转化算法DSNN，在此基础上开展了该算法在不同参数确定策略、判决策略、编码策略上的对比研究。然后我们利用人工神经网络的可折叠特性开发了新的前向计算方法DSNN-fold，并与CNN、DSNN以及前人的工作进行了精度效果的对比。

第四章介绍了基于时间编码的大规模脉冲神经网络运算开销优化方法。我们首先根据实验结果判定影响网络运算开销的关键点。然后我们定义逆向编码的时间编码方式和时钟神经元机制，并对转化过程进行了理论推导。最后

我们建立基于时间编码的转化算法TDSNN，并给出实验结果，证明转化得到的SNN是高精度、低开销的。

第五章介绍了基于脉冲神经网络转化算法的神经网络加速器设计。我们首先分析了TDSNN算法的部署方法和运算开销降低的关键点。然后我们根据算法特性从不同角度出发设计加速器的核心运算单元，给出设计方案与算法部署方法。紧接着我们发现在参数取特殊值的情况下会带来设计方案的进一步优化。最后通过与现有神经网络加速器参考基准Cambricon进行对比实验，得出更低开销的加速器设计方案。

第六章对本文工作进行了总结，并指出了将来进一步的研究方向。

# 第2章　神经网络基础

自上个世纪五十年代以来，人工智能（Artificial Intelligence）领域的一个子领域机器学习（Machine Learning）推动了多个领域产生革命性的进展。其中，受神经科学中的脑科学研究启发而诞生的神经网络的研究发挥了重要的作用。脉冲神经网络属于神经网络研究的范围之一，具体地如图2.1所示，该图展示了AI领域的研究的分布图，可以发现脉冲神经网络（Spike Neural Network，缩写SNN）和人工神经网络（Artificial Neural Network，缩写ANN）各自的研究相对比较独立，都是受脑科学启发的机器学习研究的一部分。在最近的十五年里，ANN的研究中还出现了另外一个子分支名为深度学习（Deep Learning，缩写DL），其在任务广度和精度上的表现都代表了目前研究的最高水平[25]。深度学习通过构建高层次的神经网络实现多层级的非线性信息传递，从而获得强有力的特征学习和模式分类能力[26,27]。反观SNN，在1997年它被证明是一个兼具时间与空间信息强处理能力的第三代神经网络[28]。自此以来，SNN的算法发展至今仍然停留在学术研究阶段，未能取得类似ANN的大规模工业级应用的成果。这促使了本章中对SNN发展现状和制约因素的探索研究，本章将对比介绍ANN和CNN的技术现状和研究背景，探索SNN领域最具有潜力的研究方向。与此同时，人工神经网络加速器领域的发展愈趋成熟，探索更低开销的加速器设计已成为其中的重要方向之一。脉冲神经网络领域的发展能否带来新的神经网络加速器设计思路？这个问题促使了本章对人工神经网络和脉冲神经网络硬件的发展的对比分析。本章所涉及到的研究背景与技术趋势分析，为后续的研究做充足的准备工作。

本章的组织方式如下：

2.1节介绍了脉冲神经网络模型和人工神经网络模型的差异以及本文工作涉及到的模型的相关工作。

2.2节介绍了人工神经网络算法的发展趋势，基于此对比分析了脉冲神经网络算法的发展趋势和转化算法的相关工作。

2.3节介绍了人工神经网络硬件的发展情况和脉冲神经网络硬件的发展趋势。

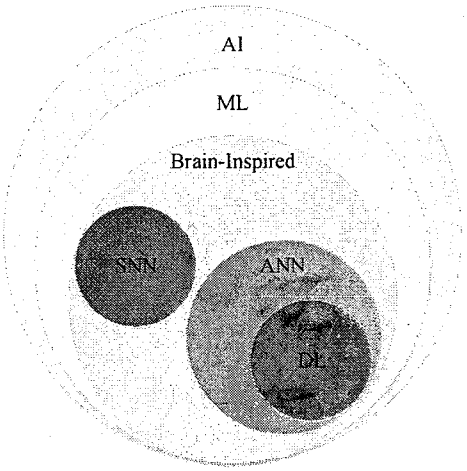


图 2.1 SNN和ANN在AI研究领域的位置[1]

**Figure 2.1 Position of SNN and ANN in the field of AI research[1]**

## 2.1 神经网络模型

本节对比介绍了人工神经网络模型和脉冲神经网络模型，其中异同点的分析是构建转化算法的必要条件。

### 2.1.1 人工神经网络模型

人工神经网络能够获得快速发展归结于其两大特性：一个是它应用广泛的非线性函数近似器的角色[29]；另外一个是其易训练的特性。通过编写程序实现高迭代次数的训练算法，例如梯度反向传播算法，ANN能够在未获知函数具体表示形式及参数先验信息的前提下，实现对任意网络或函数的高精度近似模拟[30]。本节从ANN的最简模型多层感知机(Multi-layer perceptron)入手做模型的基本介绍。

ANN的神经元模型是受脑神经元的工作机制启发而构建的，如图2.2所示。原先在脑神经元中的细胞（Soma）将树突（Denrites）传递过来的信息整合之后，发送给轴突单元(Axon)，轴突做进一步处理之后接着往下一步的轴突顶端输送信息。这一流程在人工神经网络中被类比处理为输入神经元$X_1$到$X_n$，携带权值信息$W_{1j}$到$W_{2j}$，它们做点积之后的结果经过非线性的激活函数处理得到最终的输出结果。每个权值只能连接一个输入神经元与一个输出神经元。为了修正线性点积的偏移，神经网络引入了偏置项的概念。偏置项是输入为1权值为常

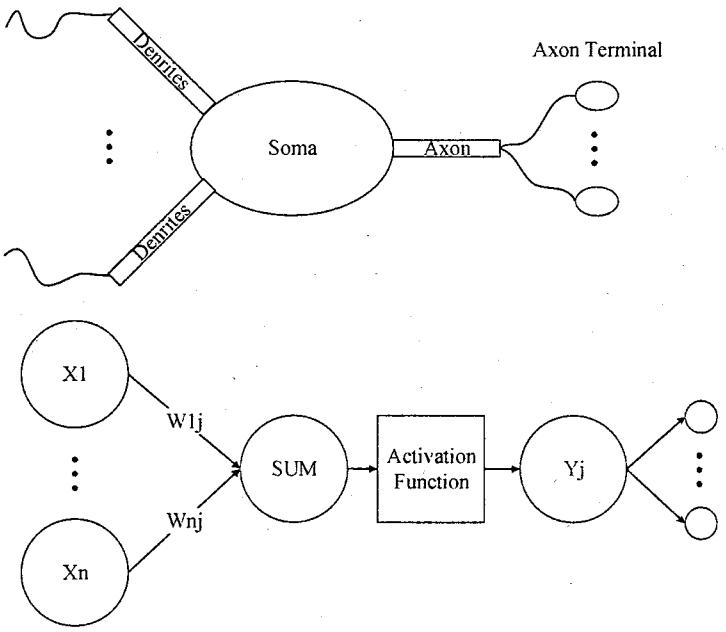


图 2.2 生物神经元与人工神经元

**Figure 2.2 Biological neurons and artificial neurons**

数的连接。最终完整的输出$Y_j$的数学表达式参见2.1：

$$Y_j = f(\sum_i \omega_{ij} \times X_i + b_j) \quad \ldots (2.1)$$

其中激活函数$f$这个非线性函数是ANN扩展性的前提[29]，也是后续转化算法的重点关注对象。 ANN的各层之间可以有各种不同的配置方式，比如在全连接层（Fully-connected Layer，缩写FC Layer）中，每个输入神经元都与每个输出神经元之间存在权值连接，那么对于输入神经元与输出神经元数目分别为$n_1$和$n_2$的全连接层，总的连接数目是$n_1 \times n_2$。本文的研究的重点是卷积神经网络，其连接方式和网络结构遵循同样的规则。另外也存在其它连接和工作方式例如循环神经网络（Recurrent Neural Network）等，这些通常用于其它网络结构和应用例如自然语言处理、语音识别等[26,31–33]。

### 2.1.2　脉冲神经网络模型

脉冲神经网络模型相对而言较复杂，本节从神经元模型、输入信息编码、网络结构、权值模型和一些其它计算机制等方面，对本文中涉及到的脉冲神经网络模型及相关工作做详细介绍。

#### 2.1.2.1 神经元模型

脉冲神经网络的神经元模型并不单一，按照神经元模型按照与生物特性的接近程度总共分为以下几类[34]：完全生物合理模型（Biologically-plausible）、生物特性启发模型（Biologically-inspired）、生物与人工混合模型、生物特性简化模型、McCulloch-Pitts类模型[35]。完全生物合理模型的侧重点在于生物合理性的模拟，基于该类神经元模型构建的神经网络难以应用于现实的智能处理任务。而抽象层次较高的McCulloch-Pitts类模型，会丢失大部分生物神经元的特性，难以应用脉冲神经网络的优势，其中的代表就是人工神经元模型。本文的研究从中选取了最具代表性的累积放电模型（integrate-and-fire，缩写IF）作为研究对象，最主要原因是它以最简洁的方式维持了生物神经元基本特性。

IF神经元和LIF神经元(leaky integrate-and-fire，缩写LIF)都属于从生物神经元的特性启发得到的简单的一类模型。这两种模型分别可以使用公式2.2和公式2.3来表示。两个模型的累积放电过程类似，在输入电流$I(t)$的持续作用下，膜电位$V(t)$随着公式描述的行为逐渐增加。两个模型的区别在于LIF模型增加了电位的泄漏效应，即神经元的电位在没有输入电位的激励下，会出现电位泄漏的现象。 IF类累积放电神经元模型的特点是简单但同时兼具一定的生物神经元特性，IF及其各种变种形式已经被部署到大批的神经计算系统中[36–39]。

$$I(t) = C \cdot \frac{dV(t)}{dt}, \qquad \ldots (2.2)$$

$$I(t) - \frac{V(t)}{R} = C \cdot \frac{dV(t)}{dt}, \qquad \ldots (2.3)$$

基于累积放电类神经元构建的神经元信息传递模型如图2.3所示。突触前神经元收到的输入不再是人工神经元中的数值，而是一个脉冲序列。脉冲序列中的信息包含了时刻t下是否存在脉冲。当突触前神经元收到一个脉冲时，与其连接的突触后神经元将会收到权值连接大小的动作电位，进而发生自身电位的累积。当自身电位$P$超过一个阈值时，将会发送动作电位给其后的神经元，动作电位的形式仍然是一个脉冲。之后自身电位发生复位效果到一个复位电位上，循环往复。图中展示的输入神经元脉冲使用了0,1的二元表示，输出神经元的脉冲也是同样的形式。可见脉冲神经网络中信息将不断以脉冲的形式传递，二值化

的脉冲信息传递方式使得使用模拟或是数字电路硬件实现较简易且开销较低。

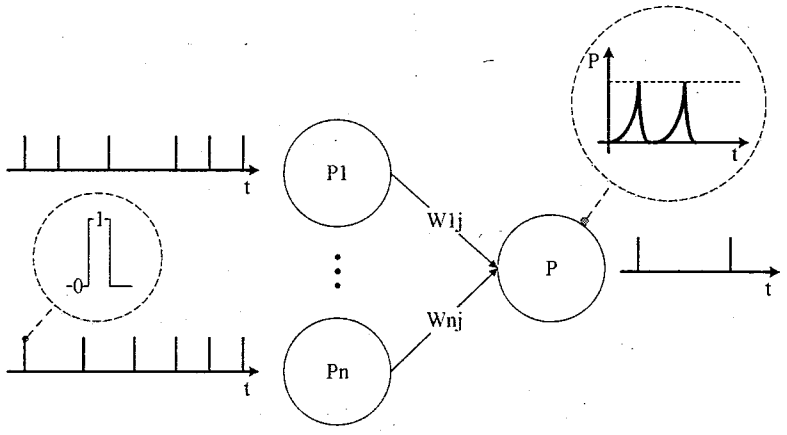


图 2.3 脉冲神经网络神经元模型

**Figure 2.3 Neuron model of spike neural networks**

对脉冲神经网络模型的理论研究也揭示了其巨大的应用潜力。Maass[28]在其工作中证明了对任意函数的模拟，脉冲神经网络都将使用不超过人工神经网络的单元来实现。并且脉冲神经网络在时间演进过程中的计算，将会在处理与时间相关的数据任务中更有效地提取其中的时间信息[40]。因此脉冲神经网络模型是一个充满潜力的网络结构，值得更加深入的研究。

2.1.2.2　信息编码

脉冲神经网络的神经元模型决定了信息在网络中传递的基本方式，而初始脉冲的产生是网络发生前向传播的来源。学术界开展的对信息编码的研究即是为了研究初始脉冲的产生机制，这亦是构建脉冲神经网络的必要部分。与人工神经网络中常见的数据预处理类似，例如对输入数据进行归一化的操作，脉冲神经网络对输入的数据经过信息编码转化为脉冲序列。然而对信息编码机制的研究并非如此简单，直至现在，脉冲神经网络的信息编码方式仍然是热议的研究方向之一[41-43]，这其中最常见的编码方式分为基于频率的编码方式（rate coding）和基于时间的编码方式（temporal coding或time coding）。

频率编码方式的基本原则通常分为平均时间脉冲频率（Spike count rate）、脉冲活动密度（Rate as spike density）和族群脉冲总量（Rate as population activity）三种[44]。基于脉冲活动密度和基于族群脉冲总量两个原则下的编码方式，

通常用于生物中脑科学的神经信息编码，计算复杂度较高。本文从频率编码方式中选取平均脉冲频率编码的方式即是考虑到该种编码的计算复杂度较低，易于部署到神经网络计算中。根据平均脉冲频率产生脉冲序列又通常分为泊松频率编码方式和均匀频率编码方式，这两者分别将输入信息编码为服从泊松分布的脉冲序列和均匀分布的脉冲序列。两种方式下整体序列的平均脉冲频率都与输入强度成正相关。为了维持频率编码表示的数据准确性，整个脉冲序列的计算需维持在一个固定的时间窗口内，增加了计算开销的不确定性。

时间编码将输入信息编码为脉冲的具体时刻，因为舍弃了时间窗口内的平均信息被认为是更高效的编码方式。 S.Thorpe等人的研究表明，人脑中第一次脉冲的时间蕴含有大量的信息。其研究中观察到，短时间内快速的信息处理过程是无法通过脉冲频率编码的方式来实现的[45]。基于此，后续的研究逐步形成了如今多种时间编码方案，例如首次脉冲时间编码（time-to-first-spike，缩写TTFS）和排序编码（Rank Order Coding）。在TTFS编码机制下，神经元的脉冲时间与信息强度成反比，即信息强度越高的神经元越早发放脉冲。而排序编码则将输入编码为脉冲发放的次序，次序决定了神经元动作电位对下层神经元的影响。这两种时间编码方式已经被应用于脉冲神经网络算法中[24]。

此外，也存在同时利用频率编码和时间编码特性的编码方式，典型的代表是脉冲间隔时间编码（inter-spike interval coding，缩写ISI），其通过一定频次的脉冲之间的时间和频次信息同时作用来编码信息[41,46]。科学研究也证明了在大脑中同时存在上述三种编码方式[47]，因此如何利用好编码方式来实现高精度脉冲神经网络仍然是值得研究的方向之一。另外还存在部分仍未被应用于脉冲神经网络转化算法的时间编码方式，例如相位编码（Phase coding）等。因此编码方式的选择空间大，如何针对性地发挥各个编码方式的效果值得进一步研究。

编码方式选择的问题还催生了专门应用于脉冲神经网络的数据集。为了防止信息编码方式的选择问题影响后续网络的构建和实验评估的公平性问题，学术界还针对开发了对应脉冲神经网络版本的数据集，免去研究人员自行选择编码方式的困扰。例如著名的图像识别数据集MNIST就存在一个用于神经计算的版本N-MNIST[48]，该数据集不再是对图像进行逐帧逐像素的存储，而是通过捕捉其中每个像素在每个时间拍之间的变化来形成逐个脉冲事件。每个脉冲事件用40比特的信息表示，其39到22比特表示图像X方向的像素地址，31到24比特表

示Y方向的像素地址，23比特表示当前的极性即脉冲事件，最低的23个比特表示时间戳。通过这种方式得到的数据集，省去了脉冲神经网络构建过程中对原始数据编码为脉冲序列的过程。数据集的表示方式也会在特定场景任务上例如物体追踪任务，相对逐帧图像的表示方式大量节省存储和访存带宽。本文由于关注转化算法将会单独设计信息编码算法，暂时未直接使用该种类型的数据集，但是不可否认该种数据集存在适配脉冲神经网络完成更高效任务处理的潜力。

#### 2.1.2.3　网络结构

构建神经网络的拓扑结构是神经网络算法设计的重要一环，不同的应用场景、任务需求往往对应不同的结构设计。学术界针对包括脉冲神经网络在内的神经网络，其结构设计都是按照如下原则进行[34]：

- 神经元和权值模型的复杂度，太过复杂的模型信息传递时的计算过于复杂；
- 网络拓扑结构的合理性，一些硬件资源有限的脉冲神经网络硬件无法部署大规模的硬件；
- 可训练或是可学习性，网络结构要能够在现有训练和学习算法的基础上训练出收敛的参数；
- 可应用性，网络结构一般随着应用到任务的不同，会根据实际情况调整拓扑结构。

研究人员已经根据上述的约束，针对性地设计出了各种各样的脉冲神经网络结构，比如前馈脉冲神经网络（spiking feed-forward networks）[49]、循环脉冲神经网络（spiking recurrent networks）[50]、深度脉冲神经网络（spiking deep neural networks）[51]、深度信念脉冲神经网络（spiking deep belief networks）[52]等等。除此之外，还有一类非常特殊的脉冲神经网络液体状态机（liquid state machine，缩写LSM）。由于其受到了脑工作原理的启发，其拓扑结构已经不再是经典的层级结构，而是一种神经元之间循环连接的结构，任意神经元既可以接收到输入神经元的动作电位又可能接收到其它层级神经元的动作电位。可见其计算的复杂度之高。而且其结构不同于一般的网络结构，因而难以在现有的主流硬件上运算。现在已经出现针对该结构的硬件设计的研究，主要集中在FPGA的设计上[53–55]，也存在少数利用光学硬件[56]和三维集成电路的研究[57]。这启发我们选择脉冲神经网络结构时也需要考虑实际的运算可行性。本文主要研究网

络都是已有的深度前馈神经网络结构，且结构由转化算法得到，降低了结构设计的难度。其余多种结构下网络的深入研究将作为未来的研究工作的关注点之一。

#### 2.1.2.4 权值模型

除了类似于人工神经网络中的权值模型外，脉冲神经网络中还存在着以模拟生物特性为目标的权值模型。其中最为著名的是突触可塑性机制（spike timing dependent plasticity，缩写STDP）[58]。该机制描述了在两个神经元的活动下，如果突触前膜受到的刺激早于突触后膜时，会引起突触的长时程增强(long term potentiation，缩写LTP)，反之则会诱导出突触的长时程抑制(long term depression，缩写LTD)。由于该机制会导致权值随着激励发生迭代变化，研究人员利用该特性作为训练算法来实现脉冲神经网络的无监督学习。实际上突触可塑性机制可以算作比较粗粒度的生物学规律，近年来也存在对权值模型更细粒度地模拟的研究[59]。针对复杂的生物学权值模型，一些试图追求通用性的脉冲神经网络硬件也支持了对这些特性的模拟[60,61]。基于对权值模型的调研，我们在本文的研究中针对转化算法得到的网络模型不使用此类复杂的权值机制。一是希望在权值机制尽可能简单的情况下实现网络权值模型的转化，二是转化算法的研究重心在于神经网络的前向计算过程，不涉及权值模型的训练。

#### 2.1.2.5 其它计算机制

脑科学存在大量的机制能够启发SNN和CNN的研究人员去模拟建立提高网络表现的人工机制，因此很多机制需要在网络的构建时做取舍。例如在 ImageNet[62]分类任务中，通过建立大规模深层次的卷积神经网络，其本身也是受到了脑科学中多层级的视觉识别皮层的机制的启发[63,64]。在脉冲神经网络中仍然在沿用这种受启发建立的计算机制，WTA（Winner-take-all）[65]机制就是其中的代表。WTA 机制是一种抑制机制，它发生在一层神经元的脉冲计算上，接收到最强动作电位的神经元在一定时间内会抑制其余神经元的脉冲发放，而只允许本神经元发放脉冲。该机制也存在变种形式，比如控制该抑制发生在有限的一簇神经元范围内等。WTA 机制经常被用来做最终的网络输出判决上，以免冗余的输出影响正确的输出判决[66,67]。我们在构建输出判决方式时也对 WTA 机制有所讨论，但最终我们实验发现在大规模脉冲神经网络中实现 WTA 机制会带来

明显的精度损失问题，因而在最终的算法里舍弃了该机制。

综上，在设计脉冲神经网络时，需要同时考虑其模型的合理性和可行性，还需要在生物合理性和运算精度与开销上做权衡，并且在诸如权值模型、网络结构和特殊生物计算机制等设计上也需要做深入的分析处理。

## 2.2 神经网络算法

本节将简要介绍人工神经网络算法的发展趋势和具有启发性意义的工作，以此为参照介绍脉冲神经网络算法的发展趋势和相关工作。

### 2.2.1 人工神经网络算法

人工神经网络算法在深度学习的介入下，已经处于蓬勃发展并投入产业应用的阶段。从网络模型训练方式看，目前人工神经网络的发展发向分为以下几个方面：

**有监督学习**。有监督学习使用预标记的数据通过构建损失函数完成训练，在此基础上建立起的网络已经投入到图像分类、语音识别等重要领域中。这其中比较有代表性的有深度网络（Deep Neural Networks，缩写DNN）[68]，以图像处理闻名的卷积神经网络（Convolutional Neural Networks，缩写CNN），擅长处理语音序列的循环神经网络（Recurrent Neural Networks，缩写RNN）以及其演变形式LSTM类的网络[69]。与此同时从DNN开始发展起了多种关于梯度下降训练算法、学习率设置优化、正则化等训练细节的研究，这些研究推动了多个领域的多种算法的可训练性研究的发展[70]，极大地降低了深层网络的训练难度。

**无监督学习**。无监督学习算法不使用任何预标记的标签数据，能够自主学习到数据中的内在联系或是特征分布，典型的代表是聚类算法、降维算法、生成算法等。人工神经网络基于无监督学习技术也发展出来了相应的算法，例如自动编码器（Auto Encoders）[71]、有限玻尔兹曼机（Restricted Boltzmann Machines）[72]、以及最近几年火热的生成对抗网络（Generative Adversarial Networks）[73]等等。可见无监督学习算法在人工神经网络中也存在大量用武之地。

**半监督学习**。介于有监督学习之间和无监督学习之间还存在一些利用部分标记的数据完成网络训练的算法，例如部分GAN（Generative Adversarial Networks）和DRL（Deep Reinforcement Learning）使用了半监督学习的技术[74]。

本文中使用转化算法针对的对象主要是利用有监督训练得到的卷积神经网络，因此本节对卷积神经网络相关的工作也做了充分的调研。另外在深度学习的研究中存在一个研究方向名为迁移学习（Transfer Learning[75]，SNN转化算法借鉴了迁移学习的很多研究思路，因此本节分析了两者的异同点。

2.2.1.1 卷积神经网络

卷积神经网络在 Fukushima 的研究中被首次提出[76]，但是 LeCun 的研究将其影响力进一步扩大到实际应用中[10]。自此，卷积神经网络能够通过梯度下降法训练得到高精确度的参数，从而在各种识别任务中取得最高水平的精度效果。CNN 的大规模应用还要归功于其独特的网络结构，它在保留部分全连接层的基础上，增加了卷积层、池化层等结构使得网络的参数数目大幅度下降，降低了运算和存储开销，提高了硬件部署的高效性。以下分析了卷积层和池化层的计算方式。

卷积层的运算可以用公式2.4来表示，其中$S(x,y,z)$代表了在输出特征图像（feature map）平面z上坐标为$(x,y)$的输出，输入层共有$CI$个特征图像，输出层共有$CO$个特征图像。权值的坐标有4个维度，除了前两个特征图像内的权值坐标$(i,j)$外，第三维坐标$k$代表了输入特征图像的平面坐标，第四维坐标为输出特征图像的平面坐标$z$，与输出的第三维坐标相同。输入层的三维坐标中$sx$和$sy$分别代表权值窗口在输入特征图像内滑动的步长参数。 $b_z$代表第$z$个输出特征图像的偏置项。整个运算过程的循环有三重，$i$和$j$层的循环计算同一个特征图像内的输入点，循环范围分别为窗口大小$KX$和$KY$。最外层循环$k$负责切换特征图像，循环范围为输入的特征图像个数$CI$。相比正常的感知机运算，卷积将点乘的操作对象由全局的神经元转化为了局部的一簇神经元，且同一输入平面内的权值维持不变。对于输入为$CI \times HI \times WI$、输出为$CO \times WO \times HO$的卷积运算，只需要选取远小于输入大小的窗口大小参数$KW$和$KH$，总权值参数数目为$CI \times CO \times KW \times KH$，而如果是全连接层则参数数目为$CI \times CO \times WI \times HI \times WO \times HO$，可见权值参数数目大幅缩减。

$$S(x,y,z)=\sum_{k}\sum_{j}\sum_{i}\omega(i,j,k,z)\times X(x*sx+i,y*sy+j,k)+b_z \quad \ldots (2.4)$$

池化层的运算可以用公式2.5来描述。和卷积不同，池化层的每一个输出特

征图像都是由对应维度的输入特征图像得到，而不是综合了多个输入特征图像的运算结果。其中*SubSample*操作可以包含最大值、最小值、平均值和线性池化等操作。近几年的部分工作还出现了同一层内的池化运算使用不同大小的窗口的技术[77-79]。池化层的一个主要作用是尽可能提取输入信息的情况下降低特征图像的大小，减少后续运算层的计算压力。

$$S(x,y,z) = Subsample_{i,j}(X(x*sx+i, y*sy+j, z)) \quad \ldots (2.5)$$

卷积神经网络虽然都是使用卷积层、池化层和全连接层等基本组件来组合产生整个完整的网络，但是网络的结构选取也需要遵循??一节提到的神经网络结构的设计原则。学术研究已经陆续推出了多种经典的卷积神经网络结构，它们的结构各不相同。例如只使用简单组件搭建的网络有LeNet[10]、AlexNet[4]、VGG Net[7]、NiN[80]和All Conv Net[81]，图2.4展示了VGG网络的结构图。也出现了一系列高级的变种网络结构，例如DenseNet[82]，FractalNet[83]，GoogleNet[6]，Residual Networks[8]。不管是何种结构中，卷积层、全连接层和池化层的基本结构仍然是最基本的组成部分，而且很多深层神经网络也依然在使用Alexnet或是VGG网络作为其基本组成部分，在此基础上做深层次的扩展。而且作者注意到研究人员为了完成复杂的任务往往会构建深度和广度兼具的神经网络，可见大规模神经网络仍然是目前完成智能任务的主流网络。大规模的卷积神经网络训练难度相比浅层网络明显提高，这里我们不对训练方法做过多的讨论。针对本文中转化算法中涉及到的卷积神经网络训练工作，作者都是采取了主流的CNN 训练平台来实现，因此默认主流的 CNN 训练方法都可以作为转化前网络的训练算法。

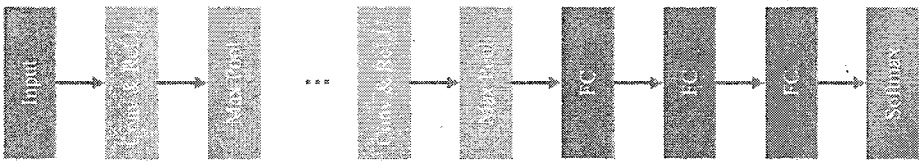


**图 2.4 VGG网络结构图。**

**Figure 2.4 VGG structure diagram.**

考虑到转化算法的关键步骤涉及激活函数的设计，本节也对CNN中的激活函数层做简单介绍。首先在浅层网络中大规模使用的激活函数例如Sigmoid和Tanh等，

参见公式2.6和2.7，它们在小规模网络中被广泛使用，但是这些函数在大规模深层卷积神经网络中的使用频率明显降低。原因是此举在大规模网络中会导致明显的梯度消失问题，为了解决此问题学术界提出使用激活函数 ReLU（Rectified Linear Unit）来做替换，该函数第一次应用于Alexnet中并取得突破性的精度效果，参见公式2.8。自此，研究人员相应开展研究开发了ReLU的多种变体，例如PReLU、leaky ReLU和ELU[84]。

$$Sigmoid(x) = \frac{1}{1+\exp(x)} \quad \ldots (2.6)$$

$$Tanh(x) = \frac{\exp(x)-\exp(-x)}{\exp(x)+\exp(-x)} \quad \ldots (2.7)$$

$$ReLU(x) = max(0,x) \quad \ldots (2.8)$$

本文的研究也利用了ReLU激活函数的特性，并将其应用于转化算法的设计中。但本文没有直接将更复杂的激活函数投入转化算法的设计中。原因有二，一是激活函数种类繁多但适合转化算法的函数并不多。Du等人[23]的工作已经研究了多层感知机的转化算法，其中的网络配置激活函数为Sigmoid，论文的结果显示存在明显的精度损失问题；二是本文的后续研究表明，转化算法的设计依赖于激活函数的设计，预先选取的激活函数难以匹配待设计的转化算法。总体来看，激活函数是CNN与SNN完成转化的基础，对其充分的调研有助于转化算法的设计。

### 2.2.2 脉冲神经网络算法

长久以来训练 SNN 的难度一直居高不下，有两方面原因。一方面，SNN 具备非连续脉冲的特性，大部分连续函数的反向传播算法等传统算法难以发挥作用。另一方面，SNN 的参数相比 ANN 大幅增加，其中包括泄漏参数、阈值、时间窗口、频率编码常数等。诸多待定的参数和参数的微小扰动都对网络的表现影响巨大。因此学术界研究了针对SNN的有监督训练算法和无监督训练算法。

#### 2.2.2.1 有监督学习算法

在有监督算法上，现有的研究主要分为三个思路：

**1.结合STDP机制和有监督学习算法开发训练算法。**代表性研究有ReSuMe[85]和强化学习作用下的有监督学习机制的训练算法[86]。

**2.通过生成式模型产生权值，进而得到网络参数。**代表性研究有2013年提出的生成算法[87]。

**3.模仿人工神经网络的训练算法。**通过建立SNN的误差计算函数，按照反向传播算法的计算方式训练网络的权值参数和其它关键参数[88,89]。

纵观这三个方向上的研究，目前最引人瞩目的要属第三种方式。这种方案由于在 ANN 中存在大量实践，因此可靠性较高。目前研究人员已经解决了其中由非连续行为导致的梯度计算问题，也解决了由脉冲特性导致的误差函数建模问题。这些举措使得SNN在MNIST数据集上实现了99.49%的精度[90,91]，达到了和主流CNN算法相同的精度水平。

即使如此，如今的训练算法仍然还存在这样两个问题：一是上述方法的有效性只在简单的任务上被证实。在复杂任务例如ImageNet上，通过有监督学习建立训练算法并训练得到的SNN，仍然无法获得媲美CNN的精度效果；二是如何权衡生物合理性和实际精度效果。有监督训练算法借鉴了大量的人工神经网络训练算法的特性，导致算法本身的生物合理性随着精度上升而下降。目前来看，生物合理性和精度之间是设计上需要权衡的因素。因此在现阶段，大规模脉冲神经网络亟需研究高精度的训练算法来支撑其应用。

#### 2.2.2.2　无监督学习算法

在脉冲神经网络的无监督学习算法研究中，建立受生物学机制启发的训练算法是现阶段的主要研究方向，例如嵌入了STDP机制的各种训练方法。这类算法利用了 STDP 机制的特性，即突触前后神经元的脉冲时间关系影响权值的变化，如图2.5所示。如果在突触前神经元发放脉冲的短时间内，突触后神经元也发放了脉冲，那么其权值连接将会被加强。反之，如果突触后脉冲的短时间内出现突触前神经元的脉冲，则削弱其连接的权值。加强和削弱的程度与脉冲间隔正相关。

但是到目前为止，无监督学习训练算法都无法使 SNN 达到媲美 ANN 的精度水平。这其中存在如下问题：首先 STDP 机制相比 ANN 训练算法缺乏精确化的数学推导，在算法上仍然属于粗粒度的算法；二是权值的调整只能发生在层

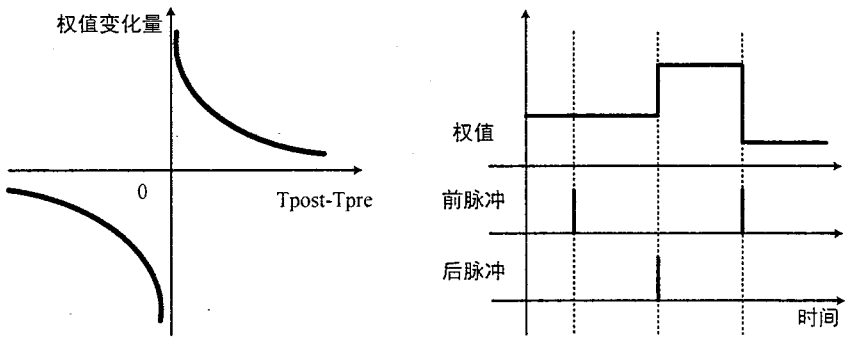


**图 2.5 STDP机制下的权值调整**

**Figure 2.5 Weight adjustment under the STDP mechanism**

内，层与层之间的训练信息传递缺少算法支撑。研究人员正在通过各种途径来试图解决这些问题，例如通过增强学习建立强有力的有监督学习机制[92]，通过建立新的层间信息自组织传递机制等[93]。还有部分研究关注于建立子网络训练算法，该类算法不完全用于训练完整的网络，转而用于建立其中部分层例如特征提取层，代替参数随机初始化方法[94]。例如在Kheradpisheh[95]的工作里，通过建立基于STDP机制的深度卷积网络，在MNIST数据集上实现了98.4%的精度效果。但是相比于主流的 ANN 网络甚至是最优秀的 SNN 网络，其精度仍然存在差距。总的来说，SNN 的无监督学习算法亟待发展，其优势是高度的生物合理性，但是精度效果不容乐观。

### 2.2.3 脉冲神经网络转化算法

在对 SNN 的训练算法做了充足的调研之后，本文发现一个非直接的方法亦可以实现高精度的脉冲神经网络，即转化算法。考虑到 SNN 的训练存在的困难，研究人员提出在已训练完成的 CNN 模型基础上做权值模型的迁移，经过网络结构的微调、重新训练和参数的调整，将参数模型直接迁移到 SNN 中，以实现同等精度水平的脉冲神经网络，这种算法即为脉冲神经网络的转化算法。

脉冲神经网络转化算法的工作可以追溯到Perez-Carrasco在2013年的工作[96]。该工作将 CNN 转化为基于频率编码的脉冲神经网络，通过建立带有泄漏和抑制效果的神经元模型，实现了对传感器捕捉到的动态事件型输入的处理。该工作

重心放在了利用转化算法实现对事件型输入的支持上，其思想启发了后续的工作中利用转化算法实现高精度的脉冲神经网络。

2015年Cao等人提出：ANN中激活函数ReLU和基于频率编码的IF脉冲神经元模型存在等价性关系[97]，该工作基于此等价关系建立了转化算法，在简单数据集上实现了接近CNN水平的精度效果。他们的工作存在一定的局限性，例如必须删除CNN网络的偏置层，并使用平均值池化代替常用的最大值池化策略等。另外，由于网络结构简单，网络精度对参数的变化不敏感，该工作并没有对转化后的脉冲神经网络提出明确的参数确定策略。

针对转化算法存在的问题，已经存在多项工作提出了具体的改进策略。Diehl等人改进了转化算法[98]，提出了权值归一化算法，降低了阈值的处理难度并且提高了脉冲频率的精确度，在MNIST数据集上实现了可忽略的精度损失。Hunsberger等人提出了在训练中加入噪声的方案，加强转化后的SNN对噪声输入的鲁棒性，以使得SNN更加贴近真实的生物场景[9]。Zambrano等人在阈值确定策略上做深入研究，以通过调整阈值来降低基于脉冲神经网络的整体脉冲频率[99]。这些关于转化算法的代表性工作普遍存在一个问题：研究对象均为小规模网络和简单数据集。对于复杂任务上，转化算法是否有效是存疑的，例如应用于ImageNet的Alexnet、VGG网络等。

转化算法的研究还涉及到很多关键性问题需要解决，作者总结了转化算法研究目前存在的挑战如下：

**1.维护计算结果的一致性。**SNN神经元处理非线性运算的行为与ANN不一致，在转化上难以实现转化前后网络的激活输出一致。已有的策略是在转化操作之前对 ANN 网络结构做初步的调整，例如修改激活函数层、删除不兼容计算单元等[96–98]。

**2.运算过程的正确映射。**目前已知的ANN结构众多，存在大量的运算方式无法映射到SNN中。例如一些研究工作正着手解决 ANN 中批处理归一化层和 Inception 层的转化问题[100]。

**3.转化大规模网络。**逐级脉冲信息传递过程会带来误差累积过程，累积误差在层次结构越深的网络中越明显。目前转化算法应用于深层网络中的代表性成果是 Rueckauer 在2018年的工作[100]。该工作针对Alexnet和VGG16等网络实现了转化算法，但是仍然存在8%左右的精度损失。后续又有工作将转化算法应用

到更深层的网络上，例如针对ResNet的转化工作[101]。但这些工作都不可避免地面临不可忽略的精度损失。

**4.降低网络的运算开销。**算法的研究最终都离不开硬件的部署，即使大规模脉冲神经网络精度可靠的前提下，也依然面领着运算开销问题。对于基于频率编码的脉冲神经网络，其运算开销与整体脉冲频率和时间窗口直接相关，降低脉冲频率和时间窗口都会降低网络的精度。因此如何降低运算开销且不明显损失精度是目前学术界的研究方向之一。已经存在利用时间编码低运算开销的特性建立转化算法的工作，例如 Rueckauer 等人提出了一种基于时间编码的转化算法，但是在 MNIST 任务上与最优水平存在2%左右的精度差距[24]。

综上，大规模脉冲神经网络转化算法是难度与潜力并存的研究方向，在此基础上实现低开销的脉冲神经网络更是值得研究的方向。本文的工作将针对这两个目标分别提出解决方案。

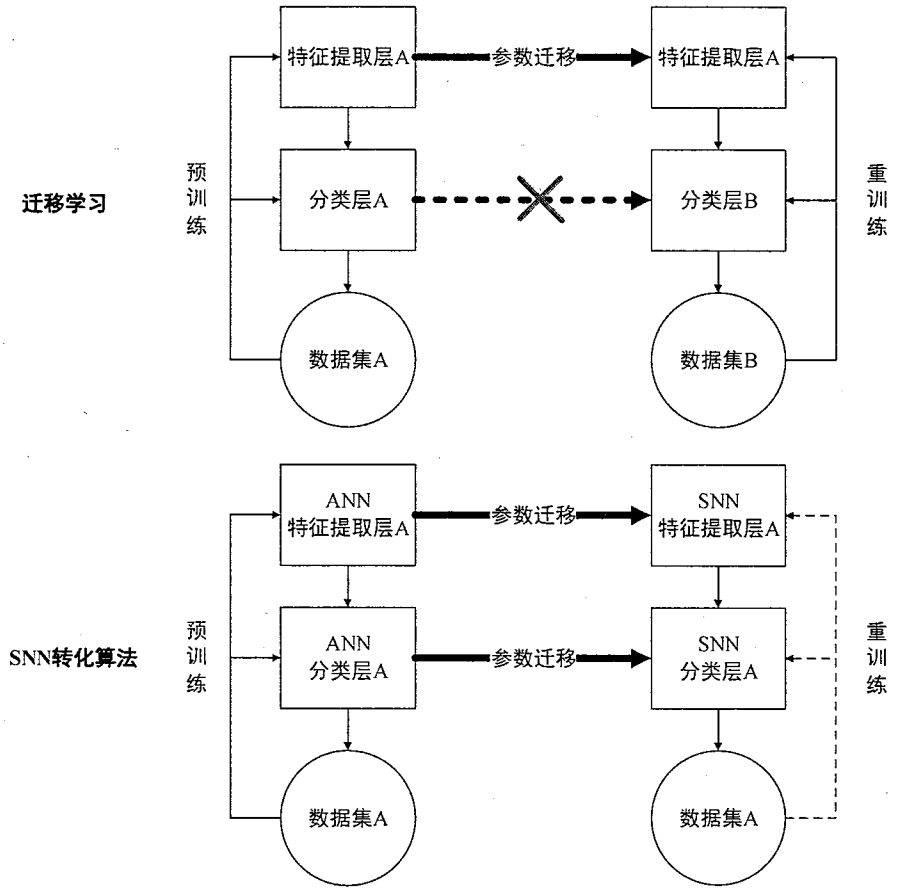


图 2.6 迁移学习与脉冲神经网络转化算法的对比

**Figure 2.6 Comparison of transfer learning and SNN conversion algorithm**

#### 2.2.3.1　转化算法与迁移学习

脉冲神经网络转化算法的基本思想与深度学习中的迁移学习（transfer learning）[75]非常接近，但是二者并不是同一个研究方向。迁移学习的主要思路是在训练得到一个深层的神经网络的基础上，利用现有网络的权值、偏置等参数作为新网络中参数搜索的起始点，训练时间相比随机初始化方法大幅缩减。新网络处理的数据集和原网络数据集相近或任务类型相近。

图2.6对比了迁移学习的过程与脉冲神经网络转化过程。网络A首先通过正常的训练方式，利用数据集A完成预训练得到收敛后的参数模型。在面临新的任务数据集B时，我们使用训练好的特征提取层A的参数初始化B网络的特征提取层。对于B网络的分类层，仍然使用随机初始化方法建立分类层B，因为通常分类层的拓扑结构需要随着任务需求不同而调整。重训练B网络至收敛，B网络的训练时间相比完全随机初始化方法大幅缩短，此过程即迁移学习过程。一般使用迁移学习的场景有以下两种：一种是从头训练一个完整的大规模网络会耗费巨大的运算资源和时间；另一种是新数据集上面临数据样本不足的问题，可以在相似任务上寻找预训练好的模型迁移到新网络上。已经出现了大批关于迁移学习的研究，这些研究已经能够实现不同领域的数据集上做迁移[102,103]。

关于脉冲神经网络算法和迁移学习之间是否存在相互借鉴学习的关系，目前没有任何工作进行过讨论，但是不可否认两者的思想确实存在相似之处。根据图2.6可以发现如下区别：首先，转化算法得到的分类层A在结构上仍然和原始的ANN分类层A相同，而迁移学习中将其替换为了新分类层B；其次，转化得到的网络处理的任务仍然是数据集A，而不是用于处理新任务数据集B；最后，针对转化算法得到的网络，重训练不再是必须项，转化前网络的结构分析调整和降低转化精度损失才是转化算法的核心工作。

总之，脉冲神经网络转化算法和迁移学习的研究不相同点如下：首先，目标对象不同，分别为脉冲神经网络和人工神经网络；其次，转化算法中转化前后的数据集和处理任务不变，算法目的是在同一数据集上达到和转化前网络同样的效果；最后，转化算法的关注点在于弥合脉冲神经网络和人工神经网络的结构差异，而迁移学习更关注新网络在迁移之后的重训练方法。

## 2.3 神经网络硬件

神经网络算法上的研究促进了神经网络应用面的扩展，传统的人工神经网络大多依赖CPU或GPU来完成模型的训练和推理过程。然而，随着网络规模的不断扩大和硬件算力的需求升高，已有的硬件无法满足部署网络的带宽和能耗要求。因此为了更高效地完成神经网络计算，设计专有的神经网络加速器和改进现有的硬件结构成为了必由之路。类似于算法领域的分类，神经网络硬件领域的研究也可以分为人工神经网络硬件和脉冲神经网络硬件两个子领域。本节介绍了学术界在这两个领域上的研究现状，分析了神经网络硬件目前的发展和存在的不足之处。

### 2.3.1 通用计算硬件

随着神经网络硬件需求的增加，原有的通用计算硬件也逐渐开发了支撑神经网络计算的功能。这些常见的通用计算硬件包括通用处理器（Central Processing Unit，缩写CPU）、图形处理器（Graphics Processing Unit，缩写GPU）、数字信号处理器（Digital Signal Processing，缩写DSP）、现场可编程门阵列（Field Programmable Gate Array，缩写FPGA）等。

CPU作为应用广泛的通用处理器，已经开始通过软硬件协同的方式支撑神经网络计算。软件方面，CPU致力于优化目前主流的神经网络计算框架，例如TensorFlow[104]，Caffe[105]等。硬件方面，CPU增加了低精度操作的支持，尤其是其中的向量处理单元（vector processing unit）中增加了对可变精度神经网络运算的功能[106]。

GPU作为一款图形处理器天然支持了高度的并行计算功能，因而擅长处理向量、矩阵类运算。目前主流的计算框架大多支持使用GPU加速神经网络计算，GPU也反向支持了神经网络加速的软件库，例如cuDNN[107]等。除此之外，GPU也开始支持低精度的计算功能，例如半精度浮点计算等。

DSP除了擅长处理数字信号处理任务外，也开始支持神经网络计算的功能，例如Synopsys的EV6x[108]。EV6x处理器集成了标量、矢量DSP和CNN处理单元，可实现高度准确和快速的视觉处理。通过多达四个的矢量DSP与CNN引擎并行工作，EV6x处理器提供可扩展的性能，支持所有视觉算法和CNN图。通过集成可选的IEEE754兼容矢量浮点模块，处理器实现了328G flops单精度浮点操作性

能，半精度浮点性能翻倍。

FPGA与上述几种通用的计算硬件相比，可编程性更强，加速器开发周期更短。例如Zhang等人[109]的工作中，利用快速傅里叶变换来减少卷积层的计算量，实现了CPU和FPGA之间高效的数据通信，通过共享内存系统实现了并发处理，浮点运算次数至少降低了39.14%。

综上，目前通用计算硬件因为需要兼顾其它计算功能，不能实现全部功能部件服务于神经网络加速处理。另外，上述硬件在设计时都考虑了降低运算的总次数，例如其中通过降低数据表示精度等操作。可见神经网络计算硬件的加速比与运算开销都是设计的重要考虑因素。

### 2.3.2　人工神经网络硬件

人工神经网络硬件的出现正是用于专门实现高效的神经网络计算。2014年之前，大部分机器学习加速器都着重于实现算法的计算部分。2014年 Chen 等人设计了一个小尺寸高吞吐量的神经网络加速器 DianNao[17]，。考虑到CNN网络的规模，该加速器特意强调了访存对加速器设计、性能、能耗的影响。在65nm工艺下，DianNao 在3.02$mm^2$上实现了452GOP/s的运算性能，速度达到了一个位宽为128bit、频率为2GHZ 的 SIMD 处理器核的 117.87 倍，但是能耗相比减少了21.08倍。自此开始人工神经网络加速器进入了快速发展阶段，相继诞生了多核神经网络加速器 DaDianNao[110]，以脉动阵列为主体计算结构的ShiDianNao[19]、TPU[111]，拆分优化数据流为目的Eyeriss[112]，以稀疏神经网络为加速器设计对象的 Cambricon-X[113] 、Cambricon-S[20] 等等。人工神经网络硬件和算法协同发展一起推动了其投入多个领域的应用中。

在众多的人工神经网络硬件中，这里选取Cambricon架构[114]做详细的介绍，其成熟的设计非常适合作为加速器设计的参考基准。Cambricon架构的整体结构图如图2.7所示。其中 IDU（Inst Decode Unit）是指令译码部件，主要负责指令的分发控制，原始的 Cambricon 结构里除了正常的指令译码分发外，还包含了标量部件、指令缓存、重排序缓存等重要部件。DMA(Direct Memory Access)部件负责片上的 IO 访问和与片外存储的接口，IDU的指令同样来源于 DMA 部件对指令存储的搬运。VFU（Vector Function Unit）负责片上的向量计算，例如向量乘法、向量加法，甚至是池化层这种非计算致密性层也可以在 VFU 实现。另外 VFU 还支持了包括多种激活函数在内的非线性函数的计算。与VFU相连的是

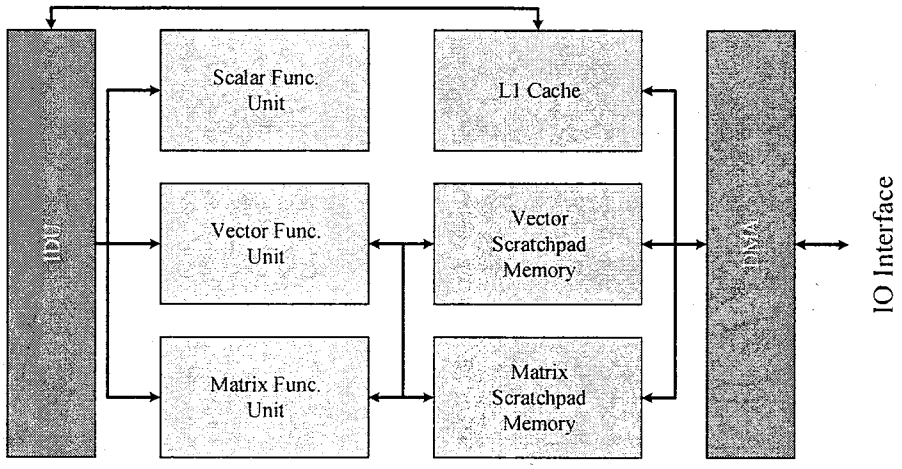


**图 2.7 Cambricon整体结构图**

**Figure 2.7 Overall Structure of Cambricon**

一个专门供向量计算使用的VS Memory（Vector Scratchpad Memory）。该种存储利用了一种交叉互联的电路设计了一个可以实现读写并行访问 SRAM 的访存方式，能够提高该种情形下的读写效率。 MFU（Matrix Function Unit）负责主要的卷积层与全连接层这种计算密集层的运算，其中集成了32个计算块，每个计算块搭配了大小为 24KB 的片上存储。这32个计算块通过 H-tree 完成连接，来完成输入数据广播到每个计算块和收集每个计算块输出的功能。

Cambricon架构的另外一个贡献是定义了适用于神经网络计算的指令集。该指令集包含了四种指令类型：计算、逻辑、控制、数据搬运。该工作指出大部分线性代数库对于神经网络运算不是有效的或者高效的。更重要的是，有很多常用的计算没有被这些库覆盖。比如不支持对一个向量进行元素级的指数操作。而且，它也不支持权值初始化中用到的随机向量产生。所以Cambricon中定义了单独的向量和矩阵运算指令，而不是简单的用现有的线性代数库重新实现这些运算。大部分神经网络计算的过程可以按层拆分，然后都以一种统一的或者说对称的方式来处理。对于这类操作来说，基于向量指令和矩阵指令的数据级并行会比传统的基于标量指令的指令级并行更高效。同时，代码的密度也会更高。Cambricon架构借助该指令集实现了高度的数据级并行。因此Cambricon架构在稠密神经网络加速器中具有代表性，且完备的指令集非常利于算法的实现，适合作为加速器设计的参考基准。

此外，在人工神经网络加速器的设计过程中，伴随产生了许多软件优化手

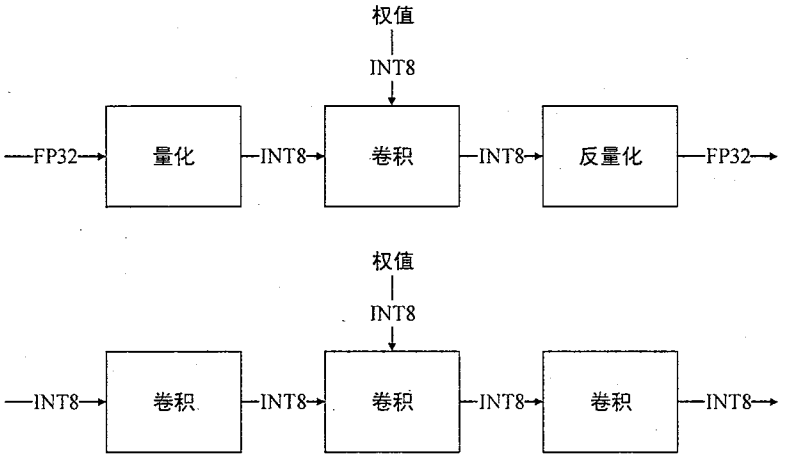


图 2.8 量化神经网络

**Figure 2.8 Neural Network Quantization**

段。为了降低神经网络权值参数的存储开销，已经实现权值共享的卷积运算甚至可以做进一步优化，例如使用更小的滑动窗口[115]。为了同时降低存储和实现高效运算，研究人员又开发出了模型压缩算法，通过剪枝等压缩技术重训练神经网络使得模型压缩到不足原来的十分之一[116]。为了降低硬件的运算开销，在数据表示上做到了极致的研究要属 BNN[117]、TNN[118]和 XNOR网络[119]的研究。这些研究的基本思想是通过将权值、神经元使用二值化表示，显著降低存储开销，同时二值化的运算将会使得硬件设计时使用移位器或逻辑运算器代替乘法器，进一步降低硬件的面积功耗开销。最终 XNOR 网络在 Alexnet 上实现了58倍的 GPU 加速比和32倍的存储开销缩减，但是精度损失达到了2.9%。

另外一个值得一提的技术是量化神经网络技术，该技术没有二值化技术那么极端，但是实际效果得到了保证，目前已经投入到大规模工业级应用中。例如将神经元和权值分别使用8比特定点数替代原始的32比特浮点数，存储开下缩小为原来的四分之一。且运算部件上8位定点的运算器开销远低于32位浮点型，如图2.8所示。量化的方法有很多种，最简单的是图上的两种，一个是添加量化和反量化算子做部分的转化，另外一种是全局网络的精度同一。量化之后的神经网络精度损失不会超过1%，能够继续完成对精度有要求的任务。后续学术界又开展了针对量化神经网络的研究，进一步完善算法，在尽可能高地获得高压缩比的同时保证网络的精度，甚至开发出了特有的训练算法帮助完成空间搜

索[120-122]。本文的硬件设计时也参考了量化神经网络的技术，将量化训练方法嵌入到硬件设计中，以设计更加高效的硬件。

综上，设计更高效的神经网络加速器仍然是当前神经网络硬件研究的主要方向。

### 2.3.3 脉冲神经网络硬件

脉冲神经网络硬件设计的研究具有独特的侧重点。人工神经网络硬件侧重于应对形式多样的大规模网络，因此对硬件的可扩展性、速度在设计要求上较高，对面积和功耗的约束取决于具体的应用。而脉冲神经网络硬件一般面向于嵌入式系统或是类脑的需求，对低功耗的追求较高，但是鉴于目前脉冲神经网络的规模以及训练算法的现状，暂时无法作用于大规模网络，主要能够在保证任务关键参数的前提下，尽可能地设计低功耗的硬件完成训练和推理，例如保证图像处理的帧率。它们大致可以分为通用脉冲神经网络加速器和面向应用的小规模脉冲神经网络处理器。

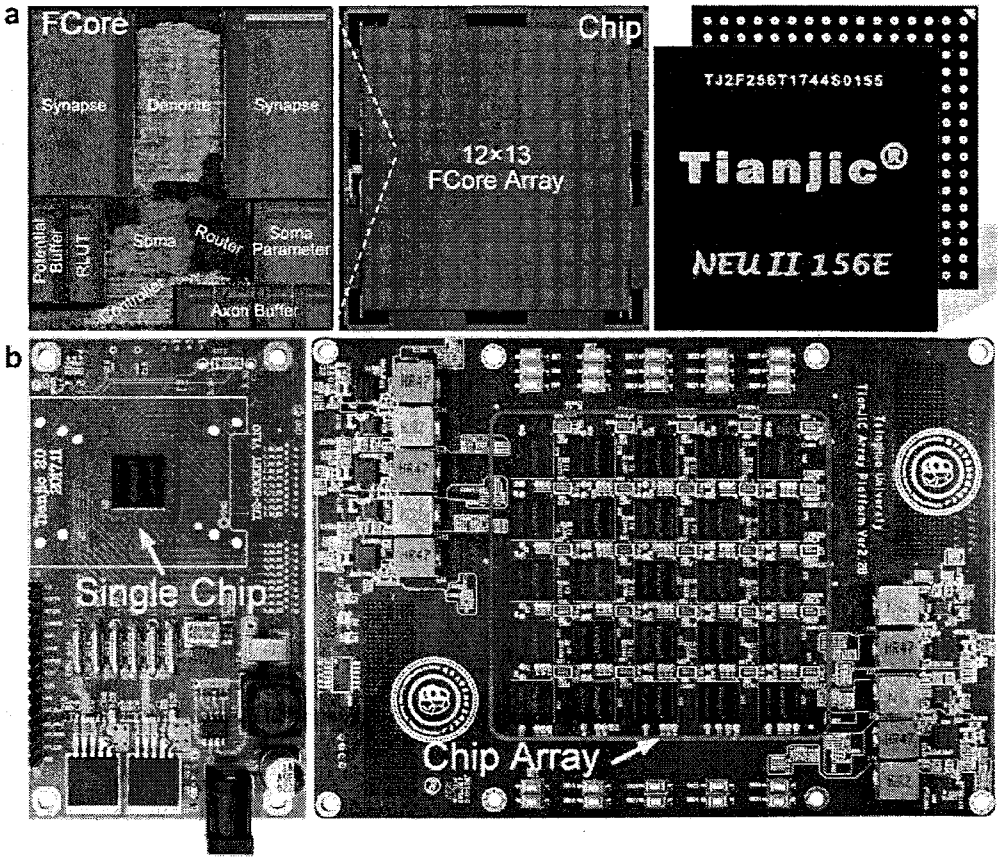


图 2.9 Tijianc芯片与测试板[2]

**Figure 2.9 Fabrication of the Tianjic chip and testing boards[2]**

通用的SNN加速器需要支持多种已有规模的SNN算法，甚至存在支持可配置多种训练算法、在线训练等功能。目前研究人员已经开发出了支持这些功能的硬件，例如利用模拟硬件实现的BrainScaleS[123]和Neurogrid[60]，利用数字电路设计的TrueNorth[21]、SpiNNaker[124]和Loihi[22]等。这其中这其中TrueNorth是最具影响力一款芯片，其设计思想影响了后续多代SNN芯片的设计。TrueNorth主要实现了一个面向大规模脉冲神经网络的低功耗芯片，在计算节点之间利用交叉总线完成网络互联，在数个简单的图像处理任务上实现了接近ANN的精度效果，同时以25mW到275mW实现1200到2600的帧率[125]。但是该硬件对复杂网络的支持性未得到证实。对于Alexnet、VGG这种复杂网络，必须对芯片做数目上的扩展，可见扩展性较差。

面向特殊应用的小规模脉冲神经网络硬件也获得了不小的研究关注，这部分研究侧重于面向专有应用例如图像识别等。此类设计关注的网络规模往往较小，且训练难度较低，大多选取复杂度较低的任务作为实现参考基准集，例如MNIST。目前已经能够实现和CNN相差无几的精度效果，且支持在线训练[126–129]。这部分研究存在的问题是网络规模具有局限性，硬件应用范围不广。另外这部分研究的结果与ANN硬件的效果对比存在不合理性，ANN硬件的可扩展性和复用性在小规模网络上无法体现，因此会出现SNN硬件在各项指标上超过ANN硬件的情况。该类研究未来将逐渐扩大研究的网络规模，开发面向SNN的独有数据集来针对性地实现更高效的专有硬件。

除了上述两类之外，一些新的研究也在对传统SNN硬件的研究产生冲击。一是特殊硬件设备的出现。例如忆阻器（Memristor）、相变内存（phase change memory）、光学器件等。从忆阻器被证明能够完美模拟STDP特性以来[130]，面向忆阻器开发SNN硬件的研究也逐步展开[131]，但是目前未见这些先进器件投入产业化应用。针对这些新兴器件的SNN硬件设计，学术界也仍然在进行论文阶段的研究，未进入实际应用阶段。二是新设计思路的出现。例如以Tianjic[2]为代表的研究，此类研究通过设计一款同时兼容人工神经网络和脉冲神经网络处理的芯片，如图2.9所示。研究显示该芯片能够支持动态化的网络配置，并且在实时物体检测、物体追踪、语音控制和避障、平衡性监测中都取得了令人满意的结果。研究人员认为设计一款通用的芯片将会为迈向通用人工智能做出铺垫，两个领域的加速器设计并存的设计思路对神经网络加速器设计具备一定的参考

价值，但是目前实用价值仍然不高。

## 2.4 小结

神经网络算法和加速器都处于蓬勃发展的阶段，在神经网络算法上取得的成果将会助力更高效的神经网络加速器的设计。本文选取了脉冲神经网络转化算法作为研究对象，算法上的突破既能推动脉冲神经网络的发展，又能启发设计更高效的神经网络加速器。

# 第3章　基于频率编码的大规模脉冲神经网络转化算法研究

脉冲神经网络算法已在小规模的数据集上取得了和人工神经网络相近的精度结果，例如MNIST。这些算法中构建的网络结构一般为小规模脉冲神经网络，其结构不超过小型卷积神经网络和多层感知机的大小。而对于更复杂的识别任务例如ImageNet，人工神经网络的做法是使用层数更深、层间神经元数目更多的卷积神经网络结构。脉冲神经网络如要完成同样复杂度的任务，则需要更大规模的网络结构来支撑。但是对于高精度大规模脉冲神经网络是否可行，学术界尚无定论。解决此问题将提高脉冲神经网络的实用价值，为构建更高效网络算法做铺垫。本章选取在小规模网络中发挥显著作用的转化算法为研究对象，开发适用于大规模网络的转化算法，以解决此问题。

本章的组织如下：3.1节介绍了本章的研究思路；3.2节介绍了构建大规模脉冲神经网络转化算法的全部过程；3.3节介绍了可折叠的大规模脉冲神经网络算法；3.4节对算法进行了实验论证；3.5节总结了本章的工作内容。

## 3.1　研究思路

结合前述相关工作可知，脉冲神经网络转化算法具有精度损失低、训练速度快的优势，且已在小规模网络中取得成功。本章以前人的工作为基础，研究了大规模脉冲神经网络转化算法的可行性，具体研究思路如下：

1.设计大规模脉冲神经网络转化算法。本节首先进行可行性的理论分析，在理论分析的基础上建立转化算法。转化算法分为两个步骤，一是针对CNN结构的调整与训练，二是训练后模型的转化。最后结合网络关键参数的确定策略即可建立整个算法。

2.研究大规模脉冲神经网络的精度稳定性。编码随机性、输出判决方式、阈值等因素都是基于频率编码的脉冲神经网络的不确定因素。对影响精度的关键因素进行研究，将有助于确定最优的参数确定策略。

3.研究可折叠的大规模脉冲神经网络算法。神经网络硬件的有限资源要求网络具备可折叠的特性，这一点对于现有的脉冲神经网络存在实现上的困难。转化得到的脉冲神经网络存在建立可折叠特性的潜力，研究此种计算方法有助

于网络的硬件部署。

## 3.2 构建大规模脉冲神经网络转化算法

本节将通过三个小结阐述大规模脉冲神经网络转化算法的阶段性研究成果，本人将该方法命名为DSNN。

### 3.2.1 理论分析

首先进行转化算法思想的理论分析，并考虑其应用到大规模网络的可行性。转化算法思想的核心在于建立CNN和SNN的内在一致性。尽管 SNN 与 CNN 在神经元模型和训练算法上存在明显区别，但是通过删除脉冲编码的时间信息，简化后的SNN与CNN结构和运算模式就存在了相似性。本文的研究发现：在相同的拓扑结构下，基于频率编码的SNN与CNN的区别仅在于数据表示上。对输入层和层间传递的数据，CNN一般采取高精度的浮点数或定点数表示法。而SNN的频率编码是降低了数据表示的精度，使其可以使用低位宽的定点数来表示。这个结论可以用来解释为何转化算法可以成功实践。这种非精确化的数据表示能否应用到大规模网络中，成为了大规模脉冲神经网络转化算法的关键问题。这一点可以参考前人在人工神经网络上的研究，人工神经网络已经在非精确表示上做出了非常深入的探索。目前甚至已经发展到使用1比特数据表示法，并以此类方法构建了相应的二进制神经网络[132-134]。但是考虑到此类网络在复杂数据集上的精度损失仍然非常明显，必然需要扩展数据表示的精度。为此，学术界又相继开展了低位宽定浮点数表示法，并取得了相应成效。这就使得转化得到大规模脉冲神经网络存在了理论的可能性。

建立转化算法的另外一个理论基础是修正线性函数（Rectified Linear Units，缩写ReLU）。学术界在大规模的人工神经网络上偏向于使用ReLU作为激活函数[135,136]，原因是该激活函数具有加速训练收敛、防止梯度消失等优秀的效果。研究发现，ReLU激活函数能够消除神经元中的负激活，并保留了原始正输出的线性特性。这和脉冲神经网络的IF模型在计算模式上具有相似性。在该神经元模型下，当神经元累积电位超过给定阈值时发送动作电位到突触后神经元，对于低于给定阈值的动作电位将不会对突触后神经元产生影响。对比两种计算模式可以发现，IF模型神经元实现了带有ReLU激活函数的人工神经网络神经元的低精度表示。这个结论建立了人工神经网络转化为脉冲神经网络的理论依据，

后文将以此为基础建立转化算法。

### 3.2.2　CNN结构调整与训练

转化算法的第一部分内容是对CNN结构的调整和训练。在这个过程中需要考虑CNN中无法转化的结构，也需要兼顾原始网络的训练效果。为此，本文设计的整体步骤如图3.1所示：

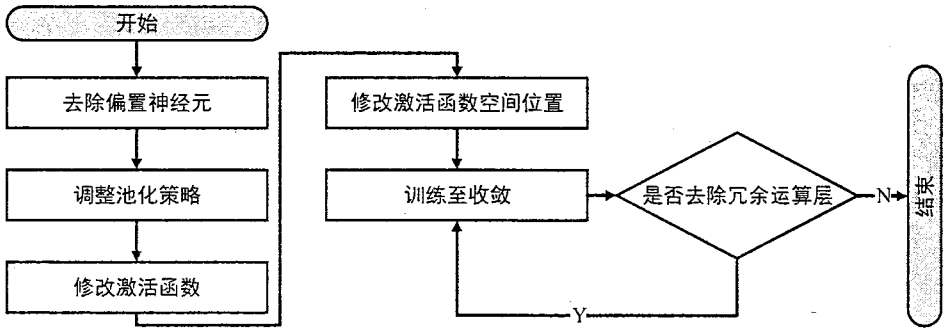


**图 3.1 DSNN算法中的CNN结构调整与训练步骤**

**Figure 3.1 Structure adjustment and training steps of CNN in DSNN algorithm**

第一步，去除原始卷积神经网络中的偏置神经元（Bias）。使用IF神经元模型模拟偏置神经元的代价比较高，且会增加额外的运算开销。因此在能够控制精度损失的前提下，本研究考虑在原始的卷积神经网络结构中去除偏置神经元。

第二步，调整原始卷积神经网络中的池化层（Pool Layer）策略为平均值池化（Average Pool）。通常在脉冲神经网络中实现最大值池化（Max Pool）需要使用多层网络结构，此处为了降低运算复杂度，本研究采取平均值池化策略替代最大值池化。

第三步，替换卷积神经网络中的激活函数为ReLU。考虑到并非所有卷积神经网络都使用了ReLU只作为激活函数，且理论分析一节也说明了该激活函数的不可替代性，本研究将在整个卷积神经网络中使用ReLU激活替换所有的激活函数。

第四步，修改激活函数层的空间位置。考虑到不管是最大值池化还是平均值池化，均不会不会改变神经元输出的正负属性。因此池化层之后的激活函数层并非必须，反而对于计算密集层ReLU激活至关重要。考虑到ReLU激活函

数层和神经元结合之后才能产生合理的转换，本研究在所有的运算层之后添加了ReLU激活函数。这些运算层包括卷积层（Convolution Layer）、全连接层（Full-connected Layer）。

第五步，训练卷积神经网络至收敛。本研究并无训练算法上的创新，网络的训练在常见的神经网络框架中完成，因此训练策略与主流训练算法无异。

第六步，去除冗余运算层。这些冗余运算层包括随机舍弃层（Dropout Layer）、批处理归一化层（Batch Normalization Layer）等。这些冗余运算层能够在卷积神经网络训练的初期，起到加速神经网络训练收敛、拓展参数搜索范围等目的。但是在第五步中神经网络已经训练至初步收敛，这些冗余运算层可以被去除，以降低卷积神经网络的拓扑结构的复杂度。与此同时，脉冲神经网络对于这些层的模拟难度较高且伴随着高运算开销，本研究不考虑对这些冗余运算层做转化。

第七步，再次训练卷积神经网络至收敛。这一步的训练不是重训练，而是在第五步训练出的神经网络权值模型的基础上，针对第六步修正过的卷积神经网络结构来再次训练，直至收敛。原因是去除冗余运算层会出现精度损失的现象，再次训练将会恢复网络的精度。

以上是本研究提出的CNN结构调整和训练过程。对于经过上述步骤的目标网络，其权值模型将会应用于下一节的转化过程，实现完整的网络转化。

### 3.2.3 CNN到SNN的结构转化

在上一节训练得到的CNN模型的基础上，脉冲神经网络的拓扑结即可通过直接映射来得到。基于此，本研究建立转化过程的基本步骤如图3.2所示。

第一步，拓扑结构映射。对于原始CNN中的卷积层、全连接层、池化层，这些结构在映射后的SNN中也存在相对应的层来实现，本研究分别将其命名为SNN-CONV、SNN-FC、SNN-POOL。此处以LeNet网络结构为例说明如何做拓扑结构的映射。如图3.3所示，输入层为$28 \times 28 \times 1$的灰度图，CNN不做特殊处理。在转化后的SNN中增加了输入编码层（Input Encoding），具体编码方式的选择在第三步中将会有详细说明。 CNN中利用卷积层将$28 \times 28 \times 1$的输入转换为$24 \times 24 \times 20$的输出，并使用ReLU作为激活。这些结构将会被一并转化到SNN中的卷积层（SNN-CONV），输入输出层的神经元数目保持不变。对于CNN中的

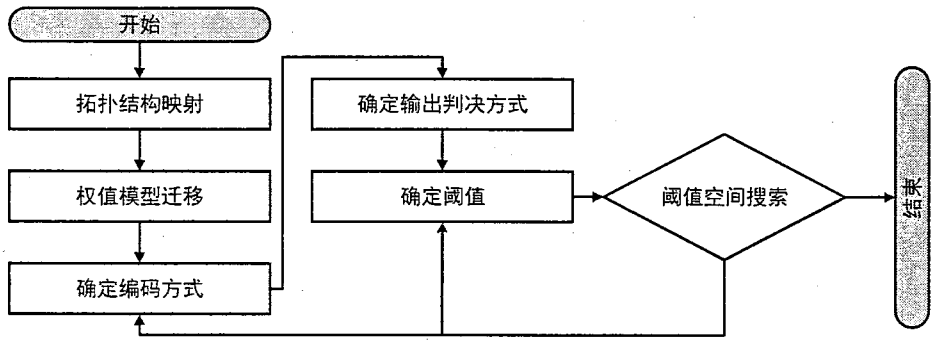


图 3.2 DSNN算法中CNN到SNN的转化过程

**Figure 3.2 Conversion process from CNN to SNN in DSNN algorithm**

池化层，该结构可以直接映射到SNN中的池化层（SNN-POOL）。对于全连接层，映射方式与卷积层类似，即将原先CNN中全连接层和激活函数层一起转化到SNN中的全连接层（SNN-FC）。最后是带有判决功能的输出层，CNN在识别任务中通常使用softmax作为输出判决方式。而转化后的SNN将使用特殊设计的ReadOut层，具体的判决方式将在第四步中详细说明。至此，脉冲神经网络的拓扑结构已构建完成。

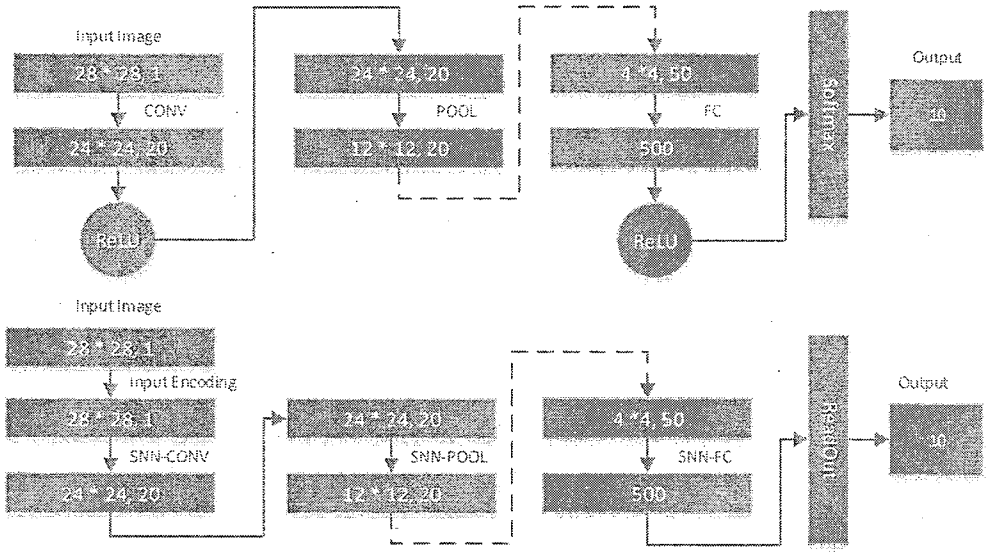


图 3.3 LeNet结构下从CNN到SNN的转化示意图

**Figure 3.3 Schematic diagram of the conversion from CNN to SNN under the LeNet structure**

第二步，权值模型迁移。由于拓扑结构的相似性，原始CNN中运算层的权值参数在映射后的脉冲神经网络中仍然保留，且分布在相同的层间和层内位置。

对于没有权值参数的层，例如平均值池化层，本研究也为其建立了带有权值的突触连接。例如对于滑动窗口大小为$kernel_w \cdot kernel_h$的平均值池化层，映射后的SNN-POOL层中的权值可以确定为$1/(kernel_w \cdot kernel_h)$，窗口内每个权值的大小相等。此时SNN-POOL层也能够实现平均值池化的功能。

第三步，确定脉冲神经网络的编码方式。在脉冲神经网络中，输入层需要经过预设的编码方式将数据从数值形式转换为脉冲形式。本研究选择以脉冲频率编码相应的数值信息，这其中包括两种最常见的编码方式：一致性编码（Uniform Coding[137]）和泊松编码（Poisson Coding[138]）。在一致性编码方式下，输入神经元发放脉冲的频率正比于输入数据的数值，神经元的脉冲间隔相等。在泊松编码方式下，脉冲按照泊松过程的方式发放脉冲，其泊松常数与输入数值成反比。本研究也将对这两种编码方式对精度的影响进行评估。另外，由于神经网络引入了输入数据的中心归一化方法，会不可避免地引入负值输入。对于负值输入，本研究将其处理为负效应脉冲。此类动作电位产生的电位累积与正效应脉冲的效果相反，即正权值连接的动作电位会导致累积电位减少，负权值连接的动作电位会导致电位增加。

**算法 1** DSNN-threshold

```
for layer in layers do
    max_pos_input = 0
    for neuron in layer.neurons do
        input_sum = 0
        for input_wt in neuron.input_wts do
            input_sum+ = max(0, input_wt)
        end for
        max_pos_input = max(max_pos_input, input_sum)
    end for
    layer.threshold = σ * max_pos_input.
end for
search on σ in the set {1, 0.1, 0.01, ...} untial a satisfactory result is obtained.
```

第四步，确定输出判决方式。脉冲神经网络的输出判决方式通常有以下几种：最先发放脉冲，最大脉冲次数和最大累积电位等。最先发放脉冲的方法依

赖准确的脉冲时间计算，不适合于基于频率编码的网络。因此本文对后两种判决方式进行了研究，并通过实验评估这两种策略对精度的影响。

第五步，确定神经元脉冲发放的阈值。阈值确定影响整个网络的精度，微小的阈值浮动会导致精度的大幅度变化。并且由于SNN中每一层有大量的神经元，每个神经元有独立的阈值参数，因此阈值的搜索空间非常庞大。为此我们设计了DSNN-threshold算法1来进行阈值空间搜索，最终选取精度最高的阈值组合来确定SNN的最大精度。

本研究之前，学术界也已经对于阈值确定的方法进行了深入的研究[98]，比较代表性的方法有基于模型的归一化方法（Model-Based Normalization）和基于数据的归一化方法（Data-Based Normalization）。基于模型的归一化方法依赖超长脉冲时间窗口，这意味着较高的运算开销。基于数据的归一化方法需要对整个网络完成整个训练集的遍历计算，并存储所有的激活后结果。最终以激活结果乘以优选的常数作为阈值参数，运算开销随着数据集规模的扩大而增大。这些方法还有共性的问题，它们的有效性只在小数据集上得到了论证。但是小数据集上的简单网络对于阈值变化的敏感度不高，阈值对最终精度的波动影响非常小。因此无法证实这些方法能否扩展到大规模的脉冲神经网络上。本研究综合考虑了前两种方法的优缺点，提出了阈值确定方法DSNN-threshold，在精度和运算效率上做了权衡。

该算法首先对每一层的层内神经元的阈值取值进行了限制，通过限制层内神经元阈值取值相同来缩小阈值的搜索空间，这样只需要为不同层的神经元分别独立设置阈值即可。但是即便如此，阈值的搜索空间还是随着层数增多会逐渐扩大，本算法的计算过程如下。对于每一层的权值，首先计算最大可能的输出累积电位（$input_sum$）。本算法以累积电位乘以一个待定常数$\sigma$作为终的阈值参数。对于$\sigma$的取值，本算法使用了一个有限宽度的取值空间搜索，在集合1, 0.1, 0.01, ...中遍历搜索，选取最高的精度结果下的$\sigma$作为代表值。最终在代表值的附近做近距离搜索来探索网络的最大可能精度。

至此，适用于大规模脉冲神经网络的转化算法DSNN已经建立完成，其精度的评估结果将会在后续实验评估章节进行说明。

## 3.3 可折叠计算优化

基于上一节得到的转化算法DSNN，本节考虑进行可折叠计算特性的引入。可折叠计算特性是设计人工神经网络硬件时所需要考虑的网络特性，此处引入该特性是为了降低硬化神经网络的部署难度，同时降低转化算法中关键参数的确定难度，提高脉冲神经网络的可扩展性。

### 3.3.1 神经网络硬件的折叠思想

神经网络硬件设计存在折叠和非折叠的思想之分。对于小规模的人工神经网络和脉冲神经网络，设计硬件可以采用空间展开的策略，根据每层的最大输入输出神经元数目，计算得到整个网络的最大输入输出计算规模。该结果将会直接决定硬件设计中的寄存器数目、静态随机存取存储器（Static Random-Access Memory，缩写SRAM）端口的上限等硬件参数。这种思想对于小规模网络是可行的，且硬件资源的开销仍然在可接受的范围内。例如针对机器学习测试库UCI[139]设计的多层感知机，其规模为$90 \times 10 \times 10$，最大可能的规模只是$90 \times 10$的运算规模。此时针对该规模部署80个硬件输入寄存器和10个硬件输出寄存器，并针对SRAM设计读数端口数目。运算单元此时配置$90 \times 10$个乘法器和10组华莱士树，每组华莱士树完成80个输入数据的累加。最终设计出的硬件加速器只有$9.02mm^2$[140]。但是近几年的研究表明，神经网络的规模正在逐渐增加。由于规模的不确定性，针对可变的规模设计完全空间展开的神经网络加速器是不现实的。

基于此方面考虑，人工神经网络加速器的大部分工作从开始之初就采用了空间折叠的设计思想[17,141]。在这种类型的设计中，由于硬件资源的限制，输入输出的规模被折叠，单次运算中的输入输出被限制上限为固定值$N_i$和$N_o$。针对每一个卷积或全连接层的运算，需要将输入和输出按照$N_i$和$N_o$进行拆分。每一层的输入输出可能无法通过硬件一次性算出，中间计算结果需要寄存器暂存。参考Du等人[23]的工作可知，使用折叠思想设计的神经网络硬件，虽然无法进行逐个图像或逐层之间的完全流水，但是其层内还是可以进行部分空间的运算流水。这种设计下的硬件具备更高的频率和面积功耗节省。

那么，脉冲神经网络为何不具备可折叠的计算特性？脉冲神经网络的输入存在时间关联性，每个输入神经元在给定的时间窗口内按照既定频率发放脉冲。

如果直接针对网络进行折叠计算，其输出神经元行为与正常整体计算时的行为并不一致。因为小规模的输入神经元容易产生超出阈值的动作电位，而得不到其它输入神经元的及时修正。因此，折叠计算的脉冲神经网络会产生更高频率的输出，破坏网络的精度，详情参考实验与评估章节。

### 3.3.2　脉冲神经网络折叠算法

本节提出了可折叠的脉冲神经网络算法，并命名为DSNN-fold，算法描述参见2。

**算法 2** DSNN-fold

For each layer in SNN :
**Phase 1:** Compute Negative Spikes
**for** each postsynaptic neuron $M_i$ **do**
    **for** each presynaptic neuron $N_j$ **do**
        **if** $N_j$ emits negative spikes **then**
            accumulate potential of synapse weight $-|\omega_{ji}|$ in $M_i$
        **end if**
    **end for**
**end for**
**Phase 2:** Compute Positive Spikes
**for** each postsynaptic neuron $M_i$ **do**
    **for** each presynaptic neuron $N_j$ **do**
        **if** $N_j$ emits positive spikes **then**
            accumulate potential of synapse weight $|\omega_{ji}|$ in $M_i$
            add to the count of spikes of neuron $M_i$ if the accumulated potential exceeds the given threshold.
        **end if**
    **end for**
**end for**

DSNN-fold主要描述了一个两阶段的计算过程，根据输入神经元的脉冲极性来区分两个阶段的计算，如图3.4所示。在第一阶段，控制只有负脉冲效果的神

经元对突触后神经元产生电位累积影响。由于负脉冲效果只会减少神经元的累积电位，在这个阶段突触后神经元不会发放脉冲。在第二阶段，控制只有正脉冲效果的神经元对突触后神经元产生电位累积影响。神经元按照累积放电模型的数学关系处理脉冲，当电位超过给定阈值时发放脉冲到与其连接的下一层神经元。后续每一层的计算按照相同的模式进行。

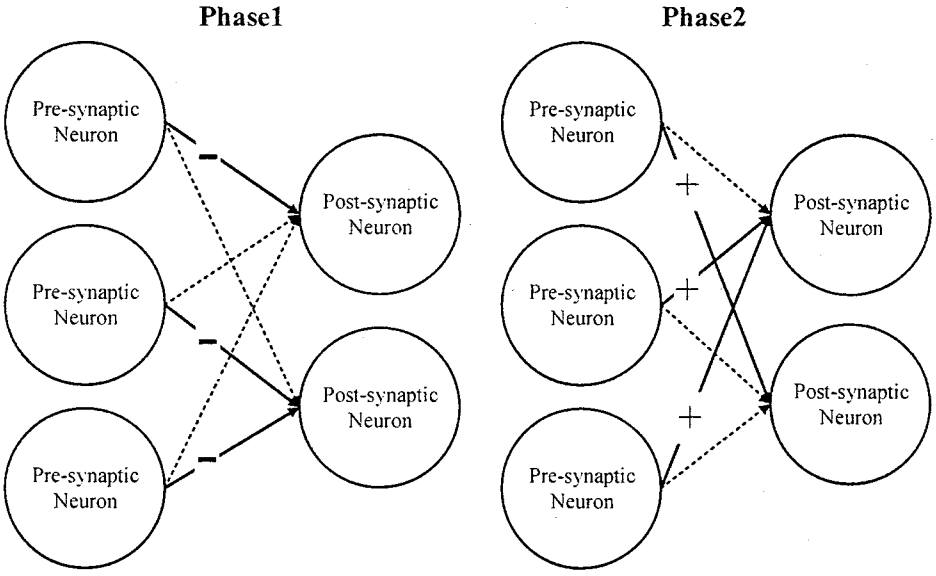


**图 3.4 DSNN-fold中两阶段的计算**

**Figure 3.4 Two computing phases in DSNN-fold**

理论上这个两阶段的计算过程是相互独立的，且在各阶段神经元都被按组拆分，不会影响最终的脉冲放电次数。如果将折叠算法应用到整个网络中，其效果如图3.5所示。首先整个网络的计算可以先按照层(layer)拆分，层与层之间需要存储整个脉冲窗口内的脉冲发放次数。然后层内的运算可以按照上述算法的描述拆分为两个阶段(phase)的运算。更进一步地，每个阶段内的运算可以按照神经元的数目进行进一步拆分，分成多个片段(fragment)的运算，中间的累积电位需要持续存储，对于片段内的计算，理论上就可以利用有限的硬件资源完成运算。

为何两阶段的运算可以有效解决脉冲神经网络的折叠问题？本研究发现，导致传统的基于频率编码的脉冲神经网络的精度不稳定的一个重要因素，是由于无法准确定位正负脉冲的实时效果。若短时间内一簇正脉冲伴随着较强的动作电位，则很可能导致突触后神经元的电位超过阈值发放脉冲。而实际的转化

算法若要求此时的神经元不发放脉冲，则会出现转化精度损失。若突触后神经元脉冲符合转化算法要求，但触发其脉冲的动作电位较大，远超出阈值，则会导致溢出的电位信息丢失，同样会导致精度损失。

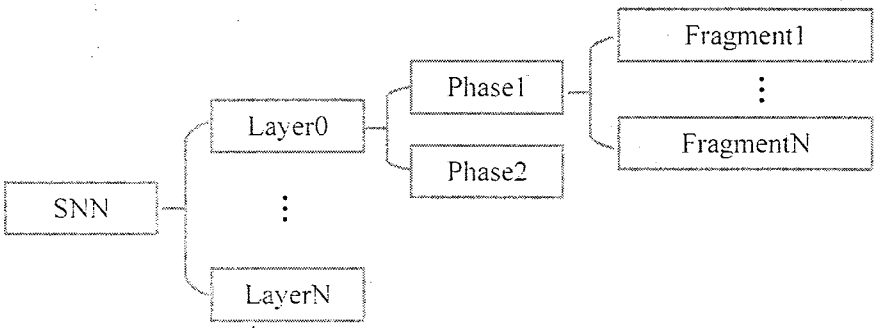


图 3.5 DSNN-fold对整个网络的折叠拆分过程

**Figure 3.5 The splitting process of DSNN-fold on entire network**

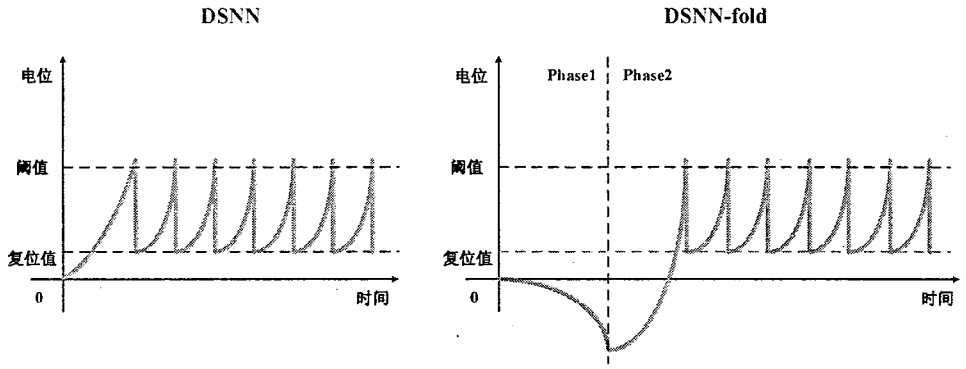


图 3.6 突触后神经元的电位变化曲线对比

**Figure 3.6 Comparison of potential changes of post-synaptic neurons**

本研究提出的DSNN-fold正是考虑了这一点，在实时计算时考虑隔离正负脉冲的影响。使用了DSNN-fold算法之后，脉冲神经网络的前向计算过程发生了时间和空间上的变化。在新算法下，脉冲神经网络中负效果的突触比正效果的突触传递路径短，因而计算发生时间窗口较早，这也间接提供了一个生物学的神经信号传递模型。神经元的典型电位曲线将发生图3.6中的变化。只使用DSNN得到的脉冲神经网络的脉冲尖峰，在时间窗口内处于近似均匀分布

状态，这符合一般情况下基于频率编码的IF神经元的电位变化规律。而使用了DSNN-fold之后，电位变化曲线明显分成了两个阶段。第一阶段电位发生降低直到负脉冲的效应结束；第二阶段只会出现正脉冲效应，累积电位并发放脉冲的过程和DSNN中的类似。

折叠算法不仅提高了脉冲神经网络的可扩展性，还取得了其它方面的收益。在折叠算法下，确定脉冲神经网络逐层的阈值不再需要DSNN中的搜索过程。网络的精度不再随着阈值的波动而出现明显损失，进而整个网络的精度水平得到了提升。

## 3.4 实验结果与评估

本节对本研究进行实验评估，主要分为以下几个方面： 3.4.1节介绍了参考基准的选取； 3.4.2节将本研究与前人研究、卷积神经网络进行精度对比；3.4.3节对影响DSNN算法精度的关键参数、策略进行评估，包括编码策略、判决策略、阈值参数等；

### 3.4.1 参考基准选取

本研究选择四个具有代表性的CNN网络为参考基准，在 Caffe 深度学习平台[105]中完成全部CNN网络的结构调整和训练过程。这四个基准神经网络分别是LeNet[10]、Caffe_cifar10_quick[142]、Alexnet[4]、VGG-16[7]。这四个 CNN 基准网络是为了三个不同的数据集设计的，涵盖的范围从简单任务到复杂任务。

LeNet最早闻名于在MNIST数据集[10]上的优异效果。该数据集由用于训练的手写数字（0-9）的60,000个28 × 28的灰度图像和用于测试的10000个数字组成，是深度学习研究的常用数据集之一。该数据集上构建的网络大多规模较小，在现阶段的研究中它仍然能够在一定程度上快速验证算法的有效性。

Cifar10数据集[142]上共有60000张彩色图像，这些图像是32×32的分辨率，分为10个类，每类6000张图。该数据集的分类难度略高于MNIST，应用在该数据集上的网络通常包括浅层网络和深层网络在内的多种网络结构，比较能代表介于简单数据集和复杂数据集之间的分类任务。本研究选取了在该数据集上取得显著效果的Caffe_cifar10_quick网络来作为参考基准，该网络也是一个卷积神经网络。

Alexnet和VGG-16都是应用在复杂的数据集ImageNet[4]上的深层神经网络。ImageNet ILSVRC-2012 包括1000个类别的高分辨率（224×224）图像，共分为三组：训练（1.3M图像），验证（50K图像）和测试（100K图像）。由于ImageNet的复杂性，使用传统的单一精度的方式难以表征网络的精细化的精度指标，因此学术界一般同时使用top1精度和top5精度来表示一个网络的精度效果。 top5精度和top1精度的区别在于判决输出的最大允许错误范围从1个扩大到了5个。

**表 3.1 参考基准神经网络的深度分布表**

**Table 3.1 Depth distribution table of benchmark neural networks**

| 网络名称 | LeNet | Caffe_cifar10_quick | Alexnet | VGG-16 |
|---|---|---|---|---|
| CNN深度 | 11 | 14 | 20 | 38 |
| SNN深度 | 9 | 11 | 14 | 24 |

在表3.1中我们展示了网络深度在CNN到SNN转化前后的变化。可见，随着CNN网络深度的增加，转化后的SNN深度也相应增加。但是SNN的深度略低于同样结构的CNN，是因为SNN能够将同等网络中CNN的运算层和激活层合并，转化为新的SNN中的运算层。另外，我们发现选取的基准网络的深度涵盖了从11到38的范围，基本覆盖了常用的网络的层数区间，且网络深度由浅及深都有涉及。接下来的几节里我们使用这些网络做为参考基准，从精度、稳定性等角度评估大规模脉冲神经网络的可行性。

### 3.4.2　精度对比

本研究将所有算法在上述参考基准集上的实验精度结果呈现在表3.2中。此处从多个角度一一进行精度效果的评估。

**CNN vs CNN-adjust**。此处使用CNN-adjust来表示对CNN做了结构调整之后并训练至收敛的网络，即DSNN转化之前的网络，而CNN的实际精度我们选取自数据集的官方展示效果。对比表格精度数据的第一列和第二列可知，CNN-adjust相比于CNN的精度损失是微小的，最小为0.01%，最大可以达到2.42%。其中最大精度损失发生在结构调整后的VGG-16网络上，这说明对于层数较深的CNN，结构的调整带来的精度损失开始逐渐显现，但是精度损失仍然在3%的范围内。这也同时说明了在可接受微弱精度损失的前提下，CNN也可以舍弃部分冗余的计算例如偏置和批处理归一化，甚至更换池化策略来换取计算资源的节省。

表 3.2 CNN和DSNN系列算法精度结果表

Table 3.2 Accuracy results table of CNN and DSNN series

| 数据集 | 网络名称 | CNN | CNN-adjust | SNN-pre | DSNN | DSNN-fold |
|---|---|---|---|---|---|---|
| MNIST | LeNet | 99.05 | 99.04 | 99.04 | 98.94 | 99.02 |
| Cifar10 | caffe-cifar10-quick | 75.00 | 74.04 | 74.0 | 73.40 | 73.56 |
| ImageNet | Alexnet(top1) | 57.26 | 55.97 | 51.8 | 54.94 | 55.55 |
| | Alexnet(top5) | 80.2 | 79.09 | 76.2 | 77.25 | 78.76 |
| | VGG-16(top1) | 71.50 | 69.13 | 49.61 | 65.71 | 68.10 |
| | VGG-16(top5) | 90.10 | 89.23 | 81.63 | 87.14 | 88.39 |

**DSNN vs CNN**。由于CNN到SNN的转化过程的精度并不是无损的，因此DSNN 转化得到的脉冲神经网络精度在CNN-adjust基础上又会出现微弱的降低。随着网络深度的加深，精度损失从0.6%逐渐扩大到3.42%，其原因在于脉冲神经网络的信息传递方式上。由于脉冲神经网络每一层之间依靠脉冲传递信息，而脉冲依靠累积电位超过阈值才能发放。由前文的理论分析可知，这种低精度表示法会导致超过阈值部分的电位会因为神经元的复位而丢失。这种损失会随着层数增加愈加明显。另外，由于阈值的确定难度也会随着层数的增加而增大，这给深层脉冲神经网络的精度带来更大的不确定性。

**DSNN-fold vs DSNN**。和DSNN相比，DSNN-fold使得SNN的整体精度得到了提升。与CNN-adjust对比可知，SNN-fold将精度损失缩小到了0.02%~1.03%。与DSNN相比，DSNN-fold精度提升了0.18%~2.39%。可见，使用了DSNN-fold之后，脉冲神经网络的精度得到显著提升。这是因为按照DSNN方式计算脉冲传递，会出现难以预估的正负脉冲交替的情况，而这种交替的情况又和实时的编码方式、触发脉冲的时刻相关，这会显著增加运算结果的不稳定性，也会增加重要参数例如阈值的确定难度。而在DSNN-fold中，这些交替的行为被两个运算阶段隔离，独立的计算使得网络不会出现正负脉冲随机交替出现的行为。该精度结果证明了DSNN-fold能够处理大规模脉冲神经网络，并以更高精度地完成识别任务。

**DSNN-fold vs DSNN-direct-fold**。此处使用DSNN-direct-fold表示对脉冲神经网络直接进行折叠计算的方法，即跳过按阶段拆分的过程直接拆分成每个片段的计算。图3.7呈现了本研究提出的折叠算法与直接折叠方法的精度。由图可

知，直接做折叠的后果是精度出现了明显的降低，在四个参考基准网络上的精度损失达到了2.52%~14.41%。这与本研究的预期是一致的。对脉冲神经网络运算直接做折叠拆分，会导致对突触后神经元的脉冲发放控制存在局部性，即局部的短暂高强度脉冲可能会导致突触后神经元的阈值被突破。而这种情况在非折叠情况下反而不会出现，因为非折叠时脉冲神经网络的整体按照时间窗口逐个单位推进，很难出现局部效应影响全局的情况。另外，DSNN-direct-fold仍然存在阈值确定困难的问题，阈值不确定导致的不稳定精度问题仍然存在。因此，DSNN-fold相比于直接折叠计算方法也存在明显的精度优势。

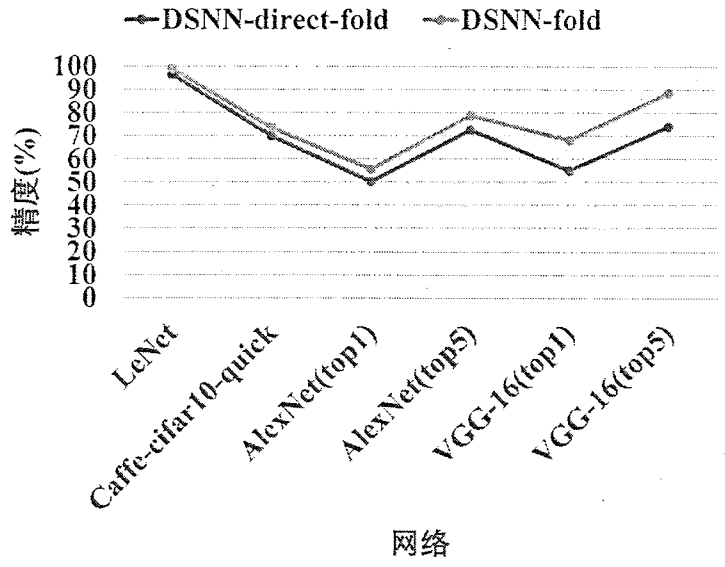


**图 3.7 DSNN-fold和DSNN-direct-fold精度对比**

**Figure 3.7 Accuracy comparison between DSNN-fold and DSNN-direct-fold**

**DSNNs vs SNN-pre/CNN**。最后本研究整体比较了DSNN和DSNN-fold相比于前人工作的SNN和CNN的精度效果，此处使用 SNN-pre 来表示学术界之前的SNN 工作，结果如图3.8所示。显然 CNN、过往研究中的 SNN、以及DSNN系列的算法在小网络和小数据集上的精度差距不大，几乎重合。但是四者的精度差距在复杂任务上开始显现。SNN-pre的整体情况最弱，原因在于之前 SNN 的研究重点关注在小网络上，在大规模网络上的研究并不多，整体的研究成果弱于CNN。本文提出的DSNN和DSNN-fold相对于之前的SNN存在明显的精度提升，已经基本达到了CNN的水平。例如在ImageNet任务上，前人研究中SNN最好精度结果是51.8%(top1)和81.63%(top5)[100,143]，本研究在基础上将精度最大提

升了6.76%。相比于CNN，DSNN-fold将精度损失控制在了2%以内，可以认为高精度大规模脉冲神经网络是可行的。

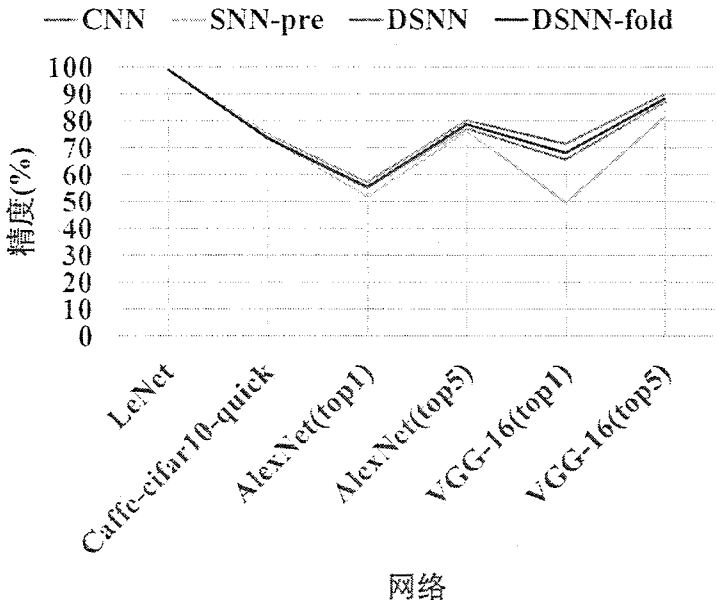


图 3.8 DSNN系列算法与前人工作精度对比

**Figure 3.8 Comparison of DSNN and DSNN-fold with previous work on accuracy**

### 3.4.3 关键参数与策略评估

本节对影响转化算法精度的编码策略、判决策略进行了评估，同时也对阈值这一关键参数的影响进行评估。

#### 3.4.3.1 编码策略评估

前文中提到，输入编码的策略和判决的策略同样会影响DSNN的精度，本节将利用实验数据说明最优的选择策略。

此处选取了均匀分布编码和泊松分布编码两种编码方式进行研究。和前人工作中类似地，本研究切分的时间窗口为500ms，最大脉冲频率为100HZ, 因为这个限制符合当前研究下的生物学客观规律。对比两种编码方式在选择的参考基准集上的精度差异，如图3.9所示。此处使用DSNN-uniform表示均匀分布编码的结果，使用DSNN-poisson表示泊松分布编码的结果。研究发现，两种方式在DSNN上的表现接近，最大精度差异不超过1%。由于产生泊松分布的输入脉冲会增加额外的运算开销，因此在对开销有严格要求的场景下优先选择均匀分

布编码，例如设计硬件加速器时可以按照均匀分布计算。但是泊松分布的输入脉冲更符合生物神经元的实际场景，因此在有比较高的仿生要求或是在做输入鲁棒性测试时优先选择泊松分布编码。本研究并未对DSNN-fold进行两种编码方式上的研究，因为DSNN-fold对编码方式不感知，运算被拆分之后，各个阶段和各个片段的脉冲时刻对最终的精度不会产生影响。

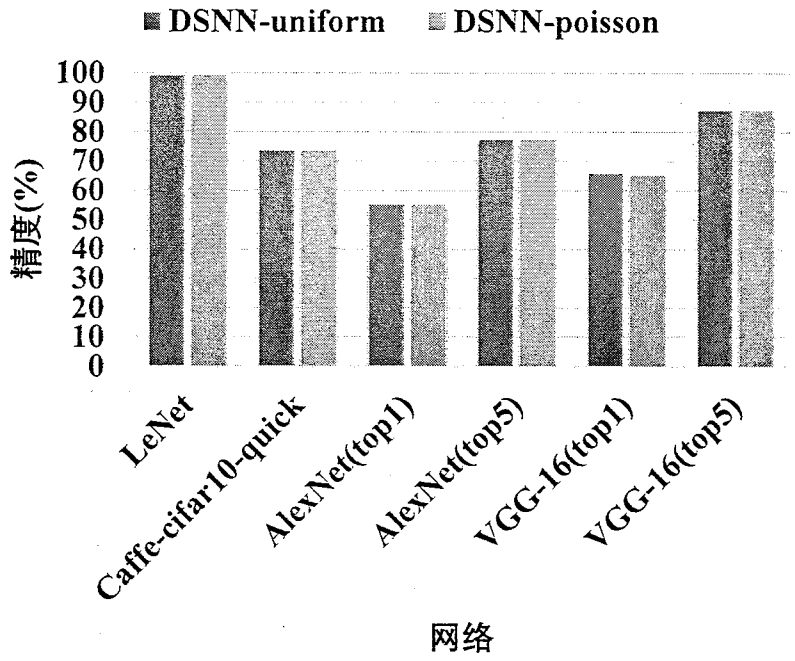


图 3.9 编码方式对精度影响的对比

**Figure 3.9 Comparison of the impact of encoding methods on accuracy**

3.4.3.2　判决策略评估

前文中提到，本研究对判决策略的预期是，最大脉冲累积电位的判决方式优于最大脉冲次数的判决方式。本研究在实验的最初就舍弃了最先发放脉冲的判决方式，因为这种方式在最简单的数据集上精度损失已经超过10%。此处在DSNN和DSNN-fold上比较最常用的两种判决方式：最大脉冲发放次数和最大脉冲累积电位。如图3.10所示，此处分别使用后缀MS和MP来代表最大脉冲发放次数和最大脉冲累积电位。这两种判决方式在小规模网络中精度接近。对于DSNN和DSNN-fold，两个判决方式在MNIST和Cifar10上的最大精度差距都在1%以内。但是当网络的规模扩大并且任务复杂度上升时，两种判决方式的精度差距开始扩大。在DSNN上，Alexnet精度在两种编码方式上的差距分别达到

了17.3%和26.83%，VGG-16的精度差距同样达到了 20% 以上。在DSNN-fold上，两个网络的精度差距也分别达到了15%以上。这说明在大规模脉冲神经网络上，使用最大脉冲发放次数的判决方式已经出现了不可接受的精度损失。原因在于大规模网络对判决层计算的精细程度要求更高，使用最大脉冲发放次数的方式会出现神经元脉冲次数难以区分的情况，阻碍了神经元输出的正确判断。因此本研究得出结论，在大规模的脉冲神经网络上推荐使用最大脉冲累积电位的判决方式。

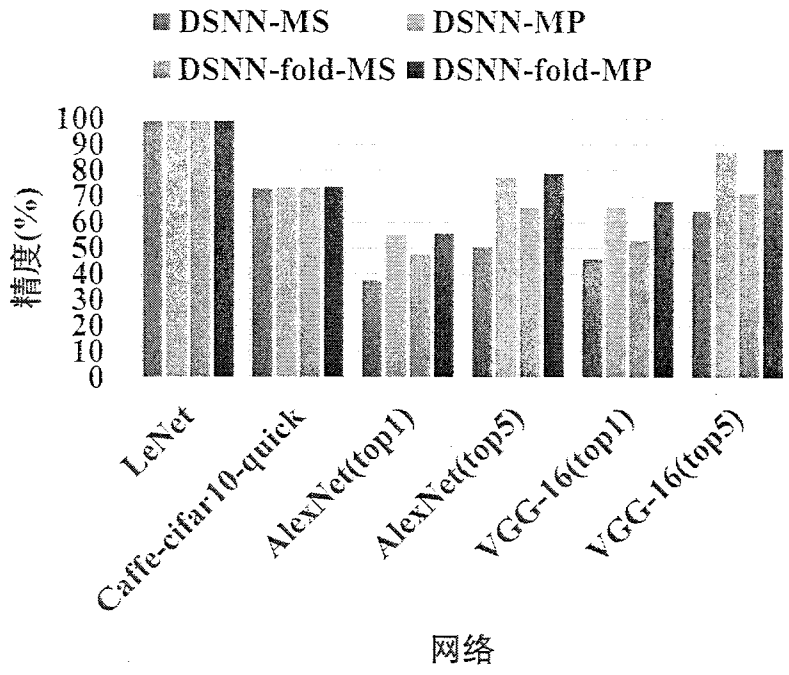


图 3.10 判决方式对精度影响的对比

**Figure 3.10 Comparison of the impact of readout methods on accuracy**

3.4.3.3 阈值影响评估

阈值参数是转化算法中最难确定的参数，本小节进行改参数对网络精度影响的评估。根据前述实验可知，小规模网络对参数的敏感度较低，因此此处选取了复杂网络VGG-16作为研究对象，结果如图3.11所示。实验以最优精度的阈值为中心点，探索25%取值范围的阈值变化对精度的影响，编码方式和判决方式等其余实验策略保持最优的策略。由图可知，随着阈值波动范围的扩大，DSNN和DSNN-fold的精度波动范围大不相同。 DSNN在25%的阈值波动下精度损失已经达到了26.86%，而DSNN-fold在整个过程中的精度损失都未超过1%。

这充分说明了DSNN-fold对阈值参数的鲁棒性大幅提升，在大规模网络中的稳定性要优于DSNN。另外，DSNN的大幅精度损失也说明了其对阈值变化很敏感，在实际应用时阈值的确定难度上远高于DSNN-fold。

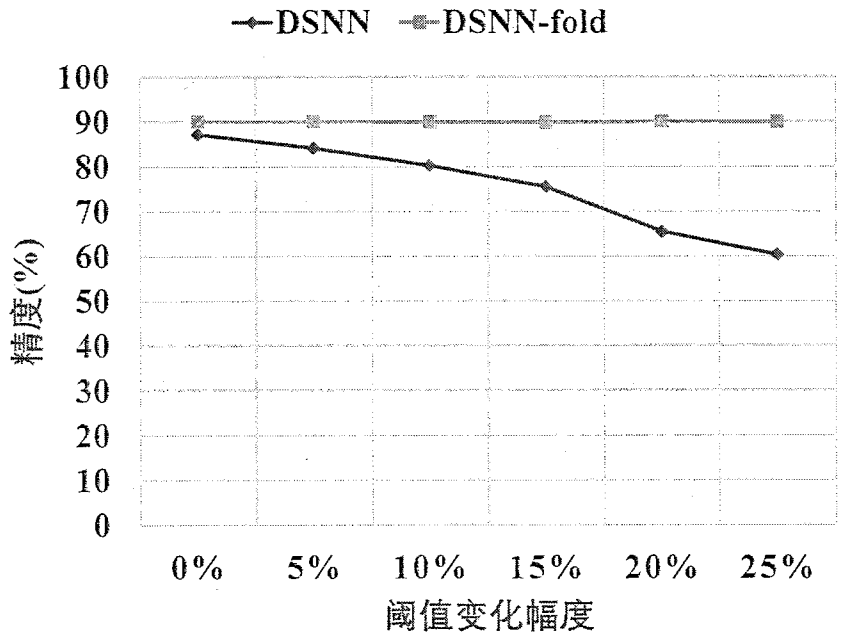


图 3.11 精度受阈值变化幅度的影响对比

**Figure 3.11 Comparison of accuracy affected by the magnitude of threshold changes**

## 3.5　小结

本章主要为了证明大规模脉冲神经网络高精度的可行性和可靠性，分别设计了新的算法DSNN和DSNN-fold。新算法在从简单到复杂的智能分类任务中都取得了和CNN相媲美的精度结果。DSNN-fold相对DSNN算法更适合在对扩展性有要求的硬件中部署，且其精度相对而言更稳定，更不易收到阈值设置的影响。但是在精度得到保证的前提下，运算开销等硬件指标也是衡量为其设计加速器的价值的关键因素。在精度得到保证的前提下，下一章将详细讨论大规模脉冲神经网络的运算开销问题，以及如何针对该问题做优化提出新的脉冲神经网络算法，以更低的运算开销完成智能任务。

# 第4章　基于时间编码的运算开销优化

DSNN系列算法解决了大规模脉冲神经网络难以训练的问题，使大规模网络的精度水平接近于CNN。但是网络实际部署到硬件上仍然存在计算开销上的不确定性，即同等精度的SNN相比于CNN是否存在计算开销上的优势？如果频率编码导致了运算开销上的问题，是否能够在算法上做优化解决这个问题？本章针对此类问题进行了深入研究。

本章的组织方式如下：4.1节通过对比 CNN 和 DSNN 系列算法的运算开销，分析现存基于频率编码的 SNN 算法的问题；4.2节提出了基于时间编码的大规模脉冲神经网络算法TDSNN，以实现运算开销上的优化；4.3节对算法的精度和运算开销进行评估；4.4节对本章的研究进行小结。

## 4.1　研究思路

本研究首先对比了CNN与SNN在典型数据集上的运算开销。此处选择了三个网络LeNet、AlexNet、VGG-16来作为实验对象，比较CNN、DSNN、DSNN-fold 在三个网络上的运算开销。此处定义的运算开销主要包括乘法运算开销和加法运算开销。对于存储开销，此处不做讨论，因为存储开销涉及神经网络的具体量化情况，而运算开销比较独立。在CNN中，神经元和权值的点乘操作既包含了乘法运算又包含了加法运算。而在SNN中，卷积和全连接层中的运算只涉及高频次的加法运算，没有乘法运算。针对非线性函数计算例如激活与泄漏效应，硬件中通常使用线性插值的方式来实现，实现的复杂度为一次乘法和一次加法运算。另外对于频率编码的SNN，其精度与脉冲发放的频次直接相关，而频次又直接影响网络的精度，因此实验时本研究选取以最低运算开销达到高精度水平的网络为评估对象。实验结果如表4.1和表4.2所示，其中（显示原因）DSNNF表示DSNN-fold。

虽然DSNN和DSNNF都利用了SNN天然无乘法的优势，实现了乘法运算的消除，但是由于高频次的脉冲编码，最终整个运算量仍然存在4到5倍左右的增加。另外对比DSNN和DSNNF的结果可以发现，DSNNF可以在更低的频次下实现更高的精度。这说明在运算开销上，DSNNF相对于DSNN存在微弱的优势。

表 4.1 CNN与DSNN运算开销表

Table 4.1 Operation cost table of CNN and DSNN

| Dataset | $CNN_{mult}$ | $CNN_{add}$ | $DSNN_{mult}$ | $DSNN_{add}$ | $\frac{DSNN_{total}}{CNN_{total}}$ |
|---|---|---|---|---|---|
| LeNet [MNIST] | 2239K | 2235K | 0K | 22M | 4.9× |
| AlexNet [ImageNet] | 357M | 357M | 0M | 3998M | 5.6× |
| VGG-16 [ImageNet] | 14765M | 14757M | 0M | 150603M | 5.1× |

表 4.2 CNN与DSNNF运算开销表

Table 4.2 Operation cost table of CNN and DSNNF

| Dataset | $CNN_{mult}$ | $CNN_{add}$ | $DSNNF_{mult}$ | $DSNNF_{add}$ | $\frac{DSNNF_{total}}{CNN_{total}}$ |
|---|---|---|---|---|---|
| LeNet [MNIST] | 2239K | 2235K | 0K | 19M | 4.2× |
| AlexNet [ImageNet] | 357M | 357M | 0M | 3641M | 5.1× |
| VGG-16 [ImageNet] | 14765M | 14757M | 0M | 129852M | 4.4× |

分析发现频率编码是SNN运算开销居高不下的原因，因此研究人员开始研究基于时间编码的SNN算法。时间编码的神经元能够充分利用脉冲的时刻来传递信息，每个神经元使用一种time-to-first-spike (TTFS)[144]的机制发放脉冲，脉冲次数不会超过一次。时间编码大幅削减了因为高频次脉冲带来的高运算开销问题。该编码方式广泛存在于SNN的各项研究中，并且生物大脑的高效神经计算也利用了该种编码方式[145]。但是时间编码在算法上的应用并不顺利，比较有代表性的工作是Rueckauer在2018年的工作[24]，他在其中提出了一个基于稀疏编码的转化方法，成功在MNIST数据集上实现了接近LeNet的精度效果，但是该方法无法扩展到大规模的脉冲神经网络。此类基于时间编码的SNN研究在复杂任务上都存在精度问题，可见该问题的研究存在一定的难度。

基于前人的研究，本研究首先针对时间编码方式做深入的分析，提出新的编码算法；然后对新算法建立脉冲神经网络前向计算机制，并进行理论推导；最后提出CNN到SNN的转化算法，完成CNN到基于时间编码的SNN的转化。

## 4.2 基于时间编码的转化算法

本节介绍了利用时间编码与带泄漏的累积放电模型，建立基于时间编码的脉冲神经网络转化算法。

### 4.2.1　逆向编码

本小节对比介绍了累积放电模型（integrate-and-fire，缩写IF）与带泄露的累积放电模型（leaky integrate-and-fire，缩写LIF），并通过后者建立逆向编码规则。

在IF神经元模型中，输出神经元将会随着输入脉冲累积电位；当累积电位超过输出神经元阈值时，该神经元发放脉冲到下一层神经元。虽然该模型已被广泛使用在前人的转化算法研究中，例如DSNN和DSNN-fold，但是整个转化算法仍然存在未解决的问题，例如神经元阈值的确定问题、高脉冲频率带来的运算开销问题。另外，IF神经元模型无法实现与时间相关的记忆，即神经元接收到一个低于阈值的动作电位时，其电位将会维持该数值直到它发放脉冲为止。这种行为和神经科学记录的神经元行为并不相符。

为了更精确地模拟神经元的行为，学术界提出了带泄露的IF神经元模型[146]。该神经元模型是是另外一个复杂的模型Hodgkin-Huxley模型[147]的简化形式，通常它采用如下公式4.1来定义：

$$I(t) - \frac{V(t)}{R} = C \cdot \frac{dV(t)}{dt}, \qquad \ldots (4.1)$$

其中$R$是一个与泄漏有关的常数，$C$代表膜电容，$V(t)$代表膜电位，$I(t)$代表输入的激励电流。上述的公式描述了单个神经元的行为，此处将其应用到脉冲神经网络上可以得到如下公式4.2：

$$\frac{V(t)}{L} + \frac{dV(t)}{dt} = \sum_i \omega_i \cdot I_i(t), \qquad \ldots (4.2)$$

其中$\omega_i$即是SNN中突触连接的权值大小，$L$是常数$C$和$R$合并之后的常数，仍然反映了神经元的电位泄漏效应。这其中的泄漏效应体现在，如果神经元的膜电位未超过阈值，其膜电位会按照公式中描述的非线性关系发生递减。这种现象存在相应的生物学解释，即如果神经元的膜电位和离子浓度未达到相应的稳态时，离子会不断穿过膜实现离子扩散，对应的膜电位也会逐渐恢复到复位水平。此处将泄漏效应应用到神经网络上，假定在时间$t_1$和$t_2$之间未发生任何脉冲输入，LIF神经元模型下的神经元电位将会按照公式4.3发生变化：

$$V(t_2) = V(t_1) \cdot exp(-\frac{t_2 - t_1}{L}). \quad \ldots (4.3)$$

对于时间编码的神经元，本研究的预期是每个神经元至多发放一次脉冲。记第$i$个与输出相关的神经元脉冲发放时间为$t_i$，可得在全局输入脉冲都到达之后的任意时刻$T_{il}$，此时的输出神经元电位可以使用公式4.4来描述：

$$P(T_{il}) = \sum_i \omega_i \cdot exp(-\frac{T_{il} - t_i}{L}). \quad \ldots (4.4)$$

根据该公式本研究发现，突触前神经元对突触后神经元的电位贡献与突触前神经元的脉冲时间成正相关。这与学术界之前的时间编码规则并不一致。在本研究之前，学术界提出了多种将输入激励编码为单个脉冲时刻的编码方式。例如Van[145]曾提出著名的基于脉冲序（rank order）的编码方式。该编码方式下，输入数据被编码为特定的顺序，可以允许不同的神经元编码为相同的顺序。输入当中强度较高的激励被编码为时间靠前的脉冲，而越晚发放脉冲的输入神经元，其对输出神经元的贡献会被惩罚因子调节。惩罚因子根据顺序，将排序靠后的激励进行指数大小的削减，这导致了越晚发放脉冲的神经元对输出的贡献越低。Van的工作利用这个特性并结合LIF神经元模型的特点，实现了构建特征提取层的目的。该特征提取层能够实现类似高斯差分滤波（Gaussian-difference-filtering）的图像处理效果，非常利于提取图像的特征点做后续的处理。但是基于脉冲序的编码方式仍然未应用于转化算法或大规模网络的构建。

基于上述分析，本研究提出与之前截然相反的逆向编码准则（Reverse Coding）：**输入强度越高，编码时间越晚。**

使用逆向编码规则建立完整的转化算法仍然面临以下挑战：

- **如何在整个脉冲神经网络中保持编码的一致性。**对输入层利用逆向规则之后，后续的各层神经元并不能按照保证同样的规则运作。它们可能出现多次脉冲的情况，也可能脉冲消失导致网络无法完成完整的前向传播。
- **如何消除CNN到SNN转化过程中的精度损失。**DSNN系列算法中精度损失的一大原因是大量参数待确定，例如阈值、频率、时间窗口等。这些不确定的因素使得转化前后产生了难以控制的精度损失。基于新的编码规则，有希望借助理论推导建立低精度损失的转化算法。

• **如何利用神经元单脉冲特性建立高精度网络。**在频率编码的SNN中，神经元依靠高频次的脉冲来近似模拟低精度表示的CNN。而在时间编码中，神经元只允许发放一次脉冲，如何能够维持相同水平的精度存在挑战。另外，时间编码的SNN对原始CNN的结构必然提出了新的要求，如何设计CNN结构成为了本研究的难题之一。

后文里将会通过新前向传播机制和充分的理论推导来解决上述挑战。

### 4.2.2　时钟神经元机制

基于逆向编码规则和LIF神经元模型，并不能按照DSNN系列算法类似的步骤直接完成转化算法的建立。第一个原因是缺少显式的抑制机制，突触后神经元只要累积电位超过阈值，就可以在任何时刻发放脉冲。这会导现脉冲时刻难以控制的情况，且脉冲次数也难以控制在一次以内。另一个原因是无法维持逆向编码的一致性。在除输入层之外的中间层，CNN中输出强度较高的神经元会转化得到累积电位较大的SNN神经元，这种神经元会较早地超出阈值进而发放脉冲，这就违背了逆向编码规则。

为了解决上述问题，本研究提出了时钟神经元机制。在该机制下，转化得到的SNN运算层将会增加一个特殊功能的神经元，本研究将其命名为“时钟神经元”（“Ticking Neuron”）。图4.1展示了时钟神经元加入后的运算层传播机制。整个前向传播过程分为两个阶段，以$T_{il}$为时间点分开。$T_{il}$之前的阶段为抑制阶段，该阶段内突触后神经元不允许发放脉冲即使其电位超过阈值。突触前神经元只允许发放一次脉冲，也即突触后神经元只关心输入神经元的第一次脉冲，忽略后续脉冲带来的影响。该阶段内时钟神经元也不会对突触后神经元产生影响。$T_{il}$之后为激活阶段，时钟神经元开始工作。时钟神经元和输出神经元之间也存在带有权值的连接，权值大小为$\omega_{tick}$。在该阶段，时钟神经元将在每个单位时间发放脉冲。输出神经元在时钟神经元的动作电位作用下将会发放脉冲，单次脉冲之后的电位累积计算可以直接取消。

为了维护逆向编码原则在整个网络中的一致性，在上述机制下本研究还对CNN权值模型进行取反处理。如图4.2所示，对于原始CNN中输出数值较小或是取值为负的神经元，本研究预期将其转化为的较早发放脉冲的SNN神经元。在权值模型取反之后，该类神经元将会在抑制解除的一个单位时间内发放脉冲。对于原始CNN中输出数值为正的神经元，本研究预期将其转化为较晚发放脉冲

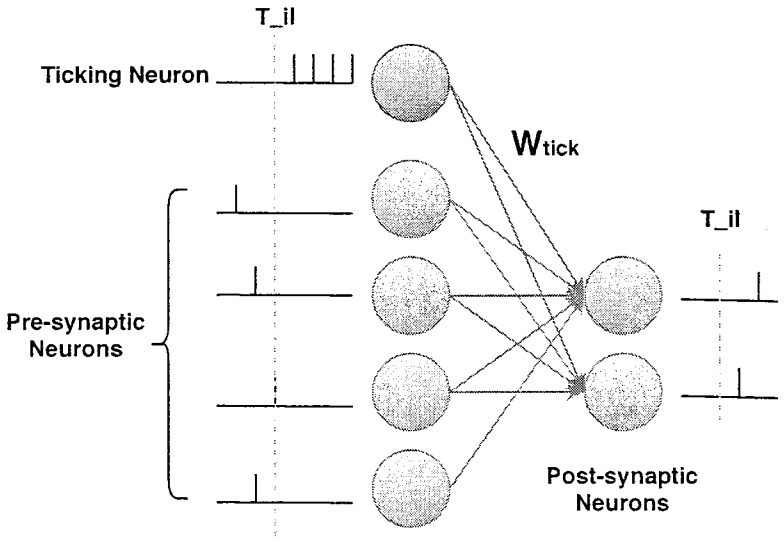


图 4.1 带有时钟神经元的前向传播机制

**Figure 4.1 Forward propagation mechanism with a ticking neuron**

的SNN神经元。在这种情况下，SNN神经元在抑制阶段将会产生负的累积电位。在抑制阶段解除之后神经元不会立即发放脉冲。借助时钟神经元机制，在逐拍的正脉冲推动下，突触后神经元将会超过阈值发放脉冲。并且在抑制阶段结束时，对于原始CNN中输出强度越大的神经元，转化后的SNN神经元累积电位越少，从而需要更长时间以超过阈值发放脉冲。综上，时钟神经元机制和权值模型的取反处理能够维持逆向编码规则的一致性。

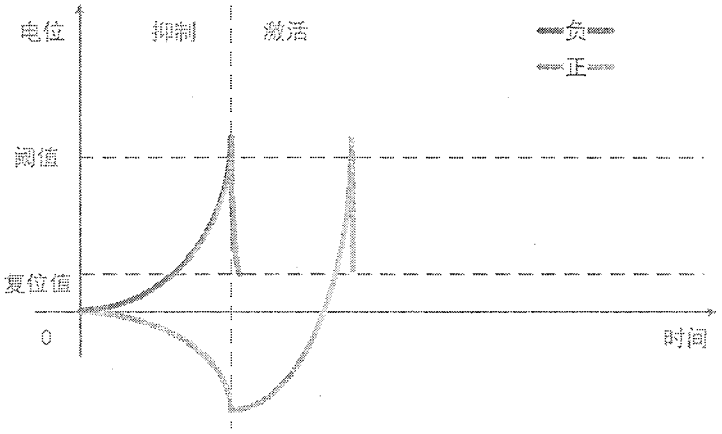


图 4.2 TDSNN神经元电位变化曲线

**Figure 4.2 Potential curve of TDSNN neurons**

此处还涉及到$T_{il}$和$\omega_{tick}$的取值策略问题。对于$T_{il}$，理论上取值必须足够大才能保证所有突触前神经元都发放了一次脉冲，最小可以取到本层神经元中最晚发放脉冲的时刻。而对于权值$\omega_{tick}$，如果取值太小，则在规定时间范围内输出神经元无法完全做到克服泄漏的效应，从而无法完成一次脉冲传递。如果取值太大，则各个输出神经元在抑制阶段一结束将很快发放脉冲，达不到区分各个输出神经元的目的，网络自然也无法完成正常的功能。本研究希望将$\omega_{tick}$表示成一个只和$T_{il}$相关的数，不和其它不定参数产生联系。这么做可以大幅降低权值确定的难度。如果$T_{il}$是个预设的常数，那么$\omega_{tick}$也会是一个运算之前就能确定的常数。最坏的情况下，$\omega_{tick}$需要根据$T_{il}$动态确定，这样其它不定参数不会对权值产生影响，从而降低权值动态计算的开销。

那么这个时钟神经元是否是必要的呢？钟神经元对于构建SNN网络是必须的，因为它保证了所有神经元按照符合SNN的规则发放脉冲。如果没有该神经元，那些电位为负的神经元将会因为泄漏效应逐渐往零电位靠近。如果要使这些神经元在复位过程中发放脉冲，我们就需要设置负的电位阈值，这是和现有的SNN的规则、神经科学的证据冲突的。

抑制机制为何能控制时间刚好是$T_{il}$？又为何时钟神经元在$T_{il}$能够刚好开始工作？本研究在此处通过建立完整的生物依据的网络来给出解释。如图4.3所示，图中蓝色神经元是原始网络结构中的神经元，橙黄色的“Tick”神经元表示时钟神经元。本层的输入神经元的抑制阶段来源于前一层的时钟神经元脉冲，这一阶段刚好持续到时刻$T_{il}$。另外，本层的时钟神经元通过计数网络监测本层其它神经元的脉冲发放情况。这幅图和图4.1的区别是增加了计数网络和抑制关系网络的机制，从而使得时钟神经元机制获得了完全可行的生物学解释，但是实际SNN计算时可以忽略这些冗余的复杂机制。

本节建立了新的基于时间编码的脉冲和神经网络前向传播机制，但是对于阈值、权值等重要参数仍然缺乏理论推导的数学表达式。为了控制转化过程精度损失，这些参数的确定本身需要经过严密的推导，并且需要同时考虑神经网络中多种不同的运算层的情况。

### 4.2.3　理论分析

本节介绍了转化过程的理论推导过程，原始CNN中的网络层将被分为两类处理，一类是有显示的权值的层例如卷积层、全连接层、平均值池化层，另一

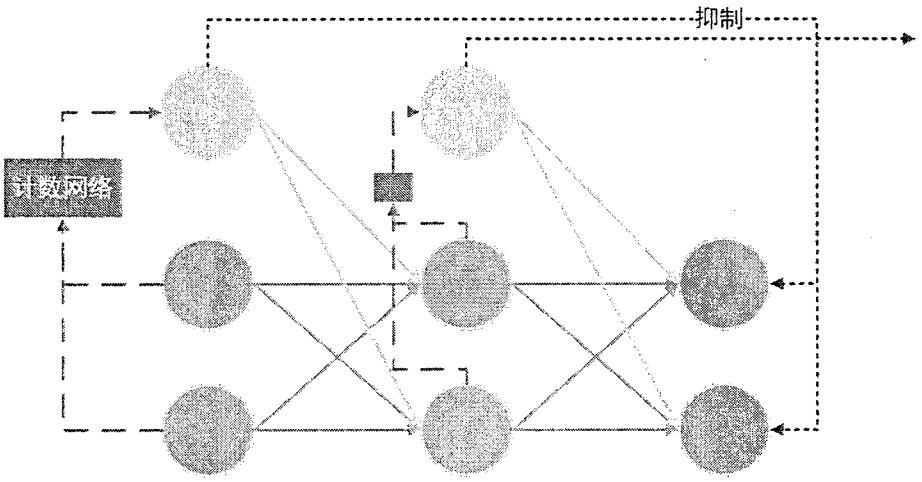


图 4.3 完整的带有时钟神经元的相互作用图

**Figure 4.3 Complete interaction diagram with ticking neurons**

类是无显示的权值的层例如最大值池化层、最小值池化层。

4.2.3.1 权值计算层理论分析

带有权值的密集计算层占据了CNN中的大部分计算量，这部分的转化过程的精度控制最为关键。令突触后神经元的脉冲阈值为$\theta$, 抑制阶段时间为$T_{il}$(可以是突触前神经元所在层的最后一次脉冲时间，也可以是预定的参数)，抑制阶段结束时突触后神经元的电位累积值为$P$。我们来计算在时钟神经元机制的作用下，突触后神经元的脉冲发放时间$T_o$(令突触后神经元的抑制阶段结束时刻为0时刻)。结合公式4.3和4.4可知，$T_o$为满足以下公式的最小整数（此处只考虑时间是离散的，而不是连续分布的）：

$$\sum_{t=0}^{T_o} \omega_{\text{tick}} \cdot \exp(-\frac{T_o - t}{L}) + P \cdot \exp(-\frac{T_o}{L}) \geq \theta, \quad \ldots (4.5)$$

其中$L$是与电位泄漏相关的预设常数，对该公式进行变形可以得到如下公式：

$$T_o = \lceil L \cdot \ln(\frac{(-P) \cdot (1 - \exp(-1/L)) + \omega_{\text{tick}}}{\omega_{\text{tick}} - \theta \cdot (1 - \exp(-1/L))}) \rceil. \quad \ldots (4.6)$$

基于此，令数据编码为脉冲时间的函数为$T(x)$。考虑到对CNN做转化的情

况下，原始CNN中数据为A的数据将会被编码到脉冲时间$T(A)$。那么对于输入神经元脉冲时间为$T(A)$、抑制阶段结束时刻为$T_{il}$的情况，转化后的SNN神经元在抑制阶段累积得到的电位可以表示为：

$$-P = \exp(\frac{T(A) - T_{il}}{L}). \quad \ldots (4.7)$$

由于转化前的CNN中输出为$A$已经无法改变，并且直接将4.7代入4.6无法求解出CNN的输出和SNN累积电位$P$之间的关系。此处本研究考虑借助激活函数来解决这个问题。

$$G(x) = \exp(\frac{T(A)}{L}) \quad \ldots (4.8)$$

结合公式4.6和4.7可知，通过构造公式4.8中描述的激活函数，实际CNN的输出从$A$转化为了$G(A)$。这样在CNN的输出和SNN的累积电位之间建立了如下关系：

$$-P = G(A) \cdot \exp(-\frac{T_{il}}{L}) \quad \ldots (4.9)$$

将公式4.9代入公式4.6可得：

$$T_o = \lceil L \cdot \ln(\frac{G(A) \cdot \exp(-\frac{T_{il}}{L}) \cdot (1 - \exp(-1/L)) + \omega_{\text{tick}}}{\omega_{\text{tick}} - \theta \cdot (1 - \exp(-1/L))}) \rceil \quad \ldots (4.10)$$

基于此，本研究得出适用于转化算法的时间编码函数为：

$$T(x) = \lceil L \cdot \ln(\frac{x \cdot \exp(-\frac{T_{il}}{L}) \cdot (1 - \exp(-1/L)) + \omega_{\text{tick}}}{\omega_{\text{tick}} - \theta \cdot (1 - \exp(-1/L))}) \rceil \quad \ldots (4.11)$$

与此同时根据公式4.8也可以确定新激活函数的表达式4.12。其中对于$x < 0$的部分可以直接限制其激活函数值为1，原因是这部分数据在原始CNN的ReLU激活之后会被取0处理，相当于在时间编码的SNN中编码的时刻为0。这样做还有个好处是消除了SNN中“负神经元”的出现，这种神经元会对神经网络的生物解释性产生冲击，降低网络对生物神经元的模拟难度。

$$G(x)=\begin{cases}\exp(\frac{1}{L}\cdot\lceil L\cdot\ln(\frac{x\cdot\exp(-\frac{T_{il}}{L})\cdot(1-\exp(-1/L))+\omega_{\text{tick}}}{\omega_{\text{tick}}-\theta\cdot(1-\exp(-1/L))})\rceil) & x\geq 0,\\ 1 & x<0.\end{cases} \quad \ldots (4.12)$$

根据上述公式可以发现，在实现整个算法的过程中需要确定的参数包括$T_{il}$、$L$、$\omega_{tick}$和$\theta$。虽然理论上可以做到根据上述公式确定这些参数的取值，但是为了降低参数确定的复杂度本研究还是对上述公式做了进一步的简化。通过约定每一层神经元的阈值为0，公式4.11可以被进一步简化为如下的表达式：

$$TS(x)=\lceil L\cdot\ln(\frac{x\cdot\exp(-\frac{T_{il}}{L})\cdot(1-\exp(-1/L))+\omega_{\text{tick}}}{\omega_{\text{tick}}})\rceil \quad \ldots (4.13)$$

考虑到$\omega_{tick}$的取值可以离线计算得到，因此为了化简时间编码与激活函数的表达式，可以为其建立表达式4.14。最终可以得到简化后的编码函数如公式4.15 所示，激活函数的最简形态如公式4.16所示。

$$\omega_{\text{tick}}=(1-\exp(-\frac{1}{L}))\cdot\exp(-\frac{T_{il}}{L}) \quad \ldots (4.14)$$

$$TS(x)=\lceil L\cdot\ln(x+1)\rceil \quad \ldots (4.15)$$

$$GS(x)=\begin{cases}\exp(\frac{1}{L}\cdot\lceil L\cdot\ln(x+1)\rceil) & x\geq 0,\\ 1 & x<0.\end{cases} \quad \ldots (4.16)$$

可见简化后的表达式简洁明了，具有易于计算和降低参数配置难度的效果。在确定了泄漏常数$L$和预设的抑制阶段时间常数$T_{il}$之后，所有参数和表达式都已经是给定的。

为了限制实际使用时抑制阶段的时间，降低确定参数$T_{il}$的确定难度，本研究还使用了ReLU激活函数的变体形式4.17来做修正，修正后的 $TS(x)$ 和 $GS(x)$ 表达式形式如公式4.18和4.19所示。

$$ReluN(N,x)=\begin{cases}0 & x<0,\\ x & x\leq N,\\ N & x>N.\end{cases} \quad \ldots (4.17)$$

$$TS(x) = ReluN(T_{il}, \lceil L \cdot \ln(x+1) \rceil) \quad \ldots (4.18)$$

$$GS(x) = \begin{cases} \exp(\frac{1}{L} \cdot ReluN(T_{il}, \lceil L \cdot \ln(x+1) \rceil)) & x \geq 0, \\ 1 & x < 0. \end{cases} \quad \ldots (4.19)$$

另外对于在DSNN和DSNN-fold中难以处理的偏置项计算问题，在本研究下将不复存在。在基于频率编码的SNN中，Rueckauer[100]曾提出使用与CNN偏置项成正比的频率来编码，但是实际精度损失的效果明显。这里我们由于使用了时间编码，偏置项的处理问题转化为确定偏置神经元的脉冲时间。偏置项可以看作权值是常数输入为1的神经元，那么根据$T(1)$或$TS(1)$都能轻易确定其脉冲时刻。注意在转化后的SNN中偏置神经元的权值一样需要对原始数值取反。

综上，对于权值计算层可以通过时钟神经元机制实现CNN到SNN的转化。

4.2.3.2　非权值计算层理论分析

非权值层的计算主要体现在最大值池化层、最小值池化层，这种池化层是CNN中最常使用的降采样层。 DSNN系列算法的研究极力避免使用这种池化层，转而采用平均值池化来替代，但是这种操作显然带来了一定的精度损失。然而利用SNN来实现最大值池化层不是一件容易的事，在基于频率编码的SNN中[100]，研究人员提出了一个门控函数的方法来实现，该门控函数只允许最大脉冲脉冲次数的神经元通过，取消所有其它神经元的脉冲传递。但是这个方法存在如下问题：一是不符合频率编码的SNN中的前向网络计算机制，也无法在生物学中找到关键性证据；二是增加了最大值池化层的计算量，原本该层在CNN中的计算非常简单高效，在SNN中模拟的难度反而很大；另外该方法目前只被证明在基于频率编码的SNN上有效，本研究将为时间编码的SNN设计独特的计算机制解决上述问题。

此处将试图使用时钟神经元机制为最大值池化层建立权值连接。令池化层权值大小全部相等为$-\omega_k$（此处权值大小必须为负值参考时钟神经元的机制），滑动窗口大小为N（一般情况下$N = KX \times KY$，$KX$和$KY$分别为$X$和$Y$方向的窗口大小），输入神经元被编码得到的最晚脉冲时刻为$T_m$。为了简化表达式推导，此处也设置神经元阈值为0。由于池化过程取决于输入窗口中各个位置的取值情

况，突触后神经元累积电位的绝对值存在上下界。上界的情况出现在窗口内的所有突触前神经元都在$T_m$时刻发放脉冲，此时累积电位大小满足如下关系：

$$-P = N \cdot \omega_k \cdot \exp((T_m - T_{il})/L) \quad \ldots (4.20)$$

下界的情况出现在窗口内只有一个突触前神经元在$T_m$时刻发放脉冲，其余所有神经元都在0时刻发放脉冲，此时累积电位满足如下关系：

$$-P = \omega_k \cdot \exp((T_m - T_{il})/L) + (N-1) \cdot \exp(-T_{il}/L) \quad \ldots (4.21)$$

根据最大值池化的原理，突触后神经元的脉冲时刻必须为$T_m$。这时只需要约束累积电位在取值上界和下界时，都能够使得脉冲时刻为$T_m$即可。结合公式4.6、4.20、4.21可得：

$$\begin{cases} L \cdot \ln(\dfrac{\omega_k}{\omega_{tick}} \cdot (1 + (N-1) \cdot \exp(-T_m/L)) \cdot \exp((T_m - T_{il})/L) \cdot C_L + 1) > T_m - 1 \\ L \cdot \ln(\dfrac{\omega_k}{\omega_{tick}} \cdot N \cdot \exp((T_m - T_{il})/L) \cdot C_L + 1) \le T_m \end{cases} \quad \ldots (4.22)$$

其中$C_L = (1 - \exp(-1/L))$。由上式可以得到$\omega_k/\omega_{tick}$的约束关系如下：

$$\begin{cases} \dfrac{\omega_k}{\omega_{\text{tick}}} > \dfrac{\exp(T_{il}/L) \cdot (\exp((T_m - 1)/L) - 1)}{C_L \cdot (\exp(T_m/L) + (N-1))} \\ \dfrac{\omega_k}{\omega_{\text{tick}}} \le \dfrac{\exp(T_m/L) - 1}{C_L \cdot N \cdot \exp((T_m - T_{il})/L)} \end{cases} \quad \ldots (4.23)$$

上述公式成立存在支撑条件，即$\omega_k/\omega_{tick}$的左边界必须小于右边界，我们通过求解不等式得到参数必须满足的条件如下：

$$L < 1/\ln(N) \& N \ge 3 \quad \ldots (4.24)$$

考虑到最大值池化层一般情况下大小不会是$1\times1$、$2\times1$和$1\times2$这几种情况，可见约束条件下是可以实现最大值池化层的成功转化。

但是对于非权值运算层应用钟神经元机制的运算开销是巨大的，这相当于把问题复杂化了。为此，本研究考虑一个更简单的替代方式来实现。本研究考虑设置突触后神经元的阈值$\theta$为$N$，同时设置$\omega_k$为1，同时设置突触后神经元

为IF神经元，并放弃使用LIF神经元模型转而采用IF神经元模型。这样突触后神经元要想达到阈值发放脉冲，必须在所有突触前神经元的脉冲到达才会出现。此时输出神经炎电位刚好为$N$，达到阈值发放脉冲。并且此时的时刻刚好为最后一个神经元发放脉冲的时刻$T_m$。由此可知，如果网络结构是权值计算层与非权值计算层交替出现的网络，那么转化后的SNN网络会出现IF神经元层和LIF元层交替出现的现象。对于最小值池化层，可以使用类似的方案来实现。最小值池化在 CNN 中负责选出窗口中的最小值，在SNN中只需要将阈值$\theta$设为1，权值$\omega_k$设为1，这样窗口内第一个神经元发放脉冲就会触发突触后神经元超过阈值发放脉冲，且时刻刚好为最早发放脉冲的神经元的脉冲时刻。

至此，本研究为构建基于时间编码的脉冲神经网络转化算法进行了充分的理论推导，接下来将开始着手建立完整的转化算法。

### 4.2.4　建立转化算法

本节将在前文的基础上建立完整的转化算法，并命名为“TDSNN”。该算法主要涵盖网络结构的训练和转化过程以及网络的前向计算过程。

#### 4.2.4.1　训练和转化过程

整个转化过程分为以下几个步骤，如图4.4所示：

1. 对原始的CNN神经网络进行训练至收敛。这里及以后提到的训练都可以采用CNN中典型的训练算法来实现，例如随机梯度下降法等。

2. 将ReLU激活函数部署在权值计算层之后并训练至网络收敛。由前所述内容可知权值计算层包括了卷积层、全连接层和一些带有权值的池化层等。实际上很多经典的CNN网络例如AlexNet和VGG网络已经采用了这种激活函数的部署方式。而对于最大值池化层或最小值池化层，这类结构之后并不需要部署ReLU激活函数，因为它们不会改变输出的正负性。这一步训练是在第一步训练得到的权值模型基础上进行的。

3. 将新激活函数$G(x)$部署在权值计算层之前并训练至网络收敛。关于$G(x)$的理论推导在上一节已经有详细的说明，这里还是强调一下由于最大值池化层不采用时钟神经元机制进行运算，因此不需要在其之前部署新的激活函数，图上呈现的也是这种情况下的池化层的处理。这一步训练开始时同样沿用上一步结束时的权值模型。

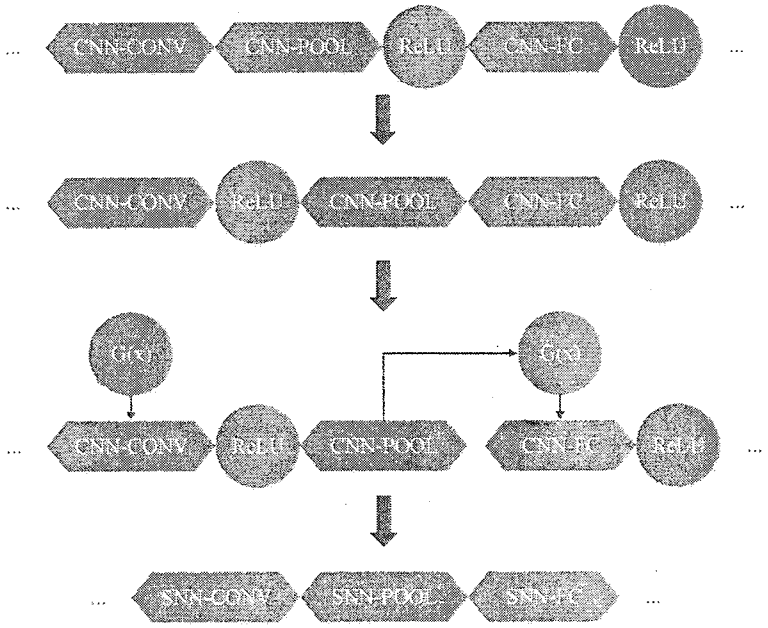


图 4.4 TDSNN算法中的网络结构转化过程

**Figure 4.4 Network Structure Transformation Process in TDSNN Algorithm**

4. 网络结构转化。 CNN中的G(x)、CONV和ReLU三者一起转化为SNN中的卷积层，同时也对池化层和全连接层做了相应结构的转化。池化层如果是最大值池化层或是最小值池化层，其神经元模型需要使用IF神经元，其余层的神经元都使用LIF神经元模型。这里需要注意最后一层输出往往伴随着输出判决，在CNN中利用softmax层来做输出判决，而在SNN中需要采用特殊处理。具体参见下一节中前向传播过程的详细描述。

5. 参数确定。对于CNN中的权值运算层，需要将其权值模型中的参数取反得到转化后的SNN中的权值模型，这里也需要注意最后一层不做取反处理。剩余的重要参数有$L$、$\omega_{tick}$、$\omega_k$、$T_{il}$，其中$\omega_{tick}$和$\omega_k$可以分别参考公式4.14来确定。 $L$和$T_{il}$为经验参数将通过后续的实验来讨论确定方式。

#### 4.2.4.2 前向传播计算

本节将给出转化得到的SNN的完整计算方法，具体包括权值运算层和非权值运算层的计算方法、输入编码层的处理方法、输出的判决方式和网络衔接处的处理方法。

完整的运算过程图如图4.5所示，为了展示每个独特部分的运作情况我们不

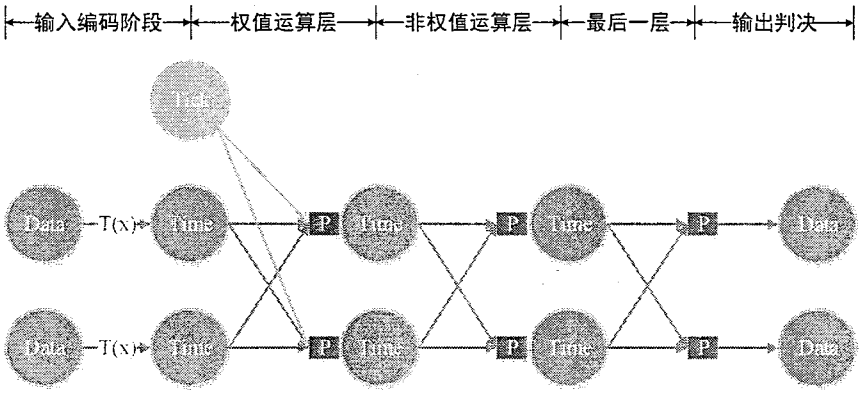


图 4.5 TDSNN算法中的网络计算流程

**Figure 4.5 Network calculation process in TDSNN algorithm**

重复展示同样运作方式的层。

**输入编码阶段**。输入层的输入数据和原始CNN类似是原始的图像数据，需要通过公式4.18编码为第一层输入的脉冲时刻，脉冲时刻控制在0到$T_{il}$范围内。后续的所有层就不再需要调用该函数计算脉冲时刻了，而是遵循脉冲神经网络的机制发放脉冲。

**权值运算层**。权值运算层占据了网络的大部分运算，运算结果分为两个模块存储，分别是电位存储模块和脉冲时间存储模块，运算过程分为两个阶段抑制阶段和时钟神经元激活阶段。在抑制阶段，输入层的脉冲时间存储模块根据存储的时间数据按照对应时刻发放脉冲，同时根据权值连接大小累积电位到输出层的电位存储模块$P$，直到抑制阶段结束。在时钟神经元激活阶段，输出层的脉冲时间存储模块监测神经元的电位。当监测到电位超出阈值时，记录此时的脉冲时刻存储到本层的脉冲时间存储模块。如果下一层也是权值运算层，那么本层的时钟神经元激活阶段也刚好是下一层的抑制阶段，同样会发生电位的累积。其余情况下也可以按照正常脉冲神经网络的计算机制进行计算。

**非权值运算层**。非权值运算层按照正常脉冲神经网络的运算机制运算，即不按照时钟神经元的机制分阶段计算，直接按照IF神经元的脉冲机制。每次脉冲累积对应权值的电位到存储模块，神经元电位超过阈值发放脉冲，记录脉冲的时刻到脉冲时间存储模块。

**最后一层**。最后一层涉及输出的判决，因此比较特殊。随着网络处理的任务不同，最后一层的输出有所差异，总的来说可以先将其转化为正常 CNN 输出相同的数据形式，然后再考虑之后的判决方式选择问题。在这一层，神经元不会发放脉冲，因为权值在转化的时候未取反，最终神经元累积的电位可以和 CNN 的原始输出成比例。

**输出判决**。输出判决方式会根据任务的不同有所差别，本文的实验主要集中在图像分类的任务上，因此根据累积的电位值选取最大值作为输出的分类判定即可。在其它的情况下比如必须输出网络的正确数据时，可以根据公式4.9乘以缩放因子来得到最终输出结果。

综上，本研究建立了基于时间编码的大规模脉冲神经网络转化算法TDSNN。

## 4.3 实验评估

本节将通过实验评估TDSNN算法在精度和运算开销上的效果。除此之外，上文中提到的待确定算法参数（泄漏常数$L$与时间窗口$T_{tl}$）对算法效果的影响也需要在实验中评估。实验中选取的参考基准网络有LeNet、AlexNet和VGG-16，这些网络和数据集的详细介绍参见3.4.1节的描述。

### 4.3.1 精度对比

本小节关注新算法TDSNN和CNN、前人工作中的SNN之间的精度对比，实验结果呈现在表4.3中。

**TDSNN vs previous SNN**。TDSNN在小规模网络LeNet上的精度比历史最高的SNN精度低0.51%. 考虑到同样是转化算法得到的精度，网络最终的精度与选取的CNN精度直接相关。而本研究选取的CNN网络精度也比之前最高的SNN精度低0.28%，且这些损失都控制在了1%以内，因此可以认为精度差距是微不足道的。在大规模网络AlexNet和VGG-16上，TDSNN的精度优势非常明显，精度提升分别达到了4.90%/3.63%和21.26%/8.48%（每个网络的两个精度结果分别是top1和top5的结果）。这说明TDSNN算法相比于之前的脉冲神经网络算法（不包括DSNN系列算法）存在明显的精度优势。

**TDSNN vs DSNN**。本研究对比了TDSNN算法与本文一个工作DSNN系列

算法的精度,发现TDSNN算法仍然存在一定的精度提升。在小规模网络中，精度只提升了0.06%，但是在大规模网络 AlexNet和 VGG-16 上精度分别提升了1.25%/1.07% 和 2.77%/1.72% 。可见在精度上，TDSNN算法相对于DSNN系列算法存在明显优势。

**TDSNN vs CNN**。经过了 SNN 到 DSNN 再到 TDSNN 的研究，SNN已经能够实现和CNN同水平的精度。可以发现，在LeNet和AlexNet上的相对精度损失为0.08%和0.46%/0.5%，这已经是1%以内的微不足道的损失了。另外，本研究还发现了一个期望之外的现象。在VGG-16网络上转化得到的SNN，其精度相对于CNN提升了1.69%/0.83%。本研究把这个现象归因于新的激活函数$G(x)$的引入产生的效果，但是仍然缺乏具体的更深层次的理论解释。此激活函数在CNN的研究中是否具有研究价值需要更深入的研究。

综上，TDSNN算法已经将SNN的精度提升到了和CNN相同的水平。

**表 4.3 TDSNN算法精度实验对比表**

**Table 4.3 TDSNN algorithm accuracy comparison table**

| Network [Dataset] | CNN err. (%) | Previous SNN err. (%) | DSNNs err. (%) | TDSNN err. (%) |
|---|---|---|---|---|
| LeNet [MNIST] | 0.84 | 0.41 [148] | 0.98 | **0.92** |
| AlexNet [ImageNet] | 42.84/19.67 | 48.20/23.80 [149] | 44.45/21.24 | **43.30/20.17** |
| VGG-16 [ImageNet] | 30.82/10.72 | 50.39/18.37 [100] | 31.90/11.61 | **29.13/9.89** |

### 4.3.2　运算开销

在精度得到保证的前提下，本研究评估了TDSNN算法针对运算开销的优化效果。运算开销量体现在乘法、加法以及总运算量，方法同4.1中的描述。LIF神经元模型不同于IF神经元模型，在计算非线性电位泄漏时，每次运算都等价于一次乘法运算。这导致在TDSNN算法里，SNN的乘法总量不可避免地增加了。具体的实验结果参见表4.4。

由表可知，TDSNN算法在三个基准网络上的乘法总量都得到了大幅缩减，但是加法总量都发生了微弱的增加。在这三个基准网络上，CNN的乘法总量分别是 TDSNN 算法的1.17、9.15、9.87倍。由于TDSNN并不像DSNN那样彻底消除了乘法，这就导致了网络的乘法总量只和最大脉冲时间有关，和网络中神经元数目无关。这也导致了乘法总量的削减比例在大网络中更加明显，原因是大

网络相比小网络脉冲时间不变但神经元数目明显增加。TDSNN算法在三个网络上都带来了额外的加法运算量，其总加法运算量分别是CNN中的1.43、1.05、1.05倍。这主要是由时钟神经元的逐拍脉冲所导致。然而，考虑到时钟神经元的最大工作时间也不会超过最大脉冲时间，因此在大网络上加法总量的增加对总运算量的增加也是微乎其微的。

此处还同时考虑了乘法和加法的运算总量。在小网络 LeNet 上，TDSNN算法带来了1.14倍的运算量增加，但是仍然低于DSNN系列的运算量。在大规模网络上，TDSNN的运算量削减效应开始显现。在AlexNet和VGG-16网络上，TDSNN相对于CNN分别实现了41.9%、42.6%的运算量缩减。这证明了TDSNN算法在大规模网络上能够实现明显的运算开销的削减效果。

**表 4.4 CNN与TDSNN算法运算开销表**

**Table 4.4 Operation cost table of CNN and TDSNN**

| Network [Dataset] | $CNN_{mult}$ | $CNN_{add}$ | $TDSNN_{mult}$ | $TDSNN_{add}$ | $\frac{CNN_{mult}}{TDSNN_{mult}}$ | $\frac{CNN_{add}}{TDSNN_{add}}$ | $\frac{CNN_{total}}{TDSNN_{total}}$ |
|---|---|---|---|---|---|---|---|
| LeNet [MNIST] | 2239K | 2235K | 1920K | 3194K | 1.17× | 0.70× | 0.88× |
| AlexNet [ImageNet] | 357M | 357M | 39M | 377M | 9.15× | 0.95× | 1.72× |
| VGG-16 [ImageNet] | 14765M | 14757M | 1496M | 15505M | 9.87× | 0.95× | 1.74× |

### 4.3.3 算法参数影响评估

相比于DSNN系列的算法，TDSNN算法待确定的参数数目大幅减少。基于频率编码的SNN需要确定包括脉冲频率常数、阈值等许多参数，确定这些参数的过程需要大量的工作量，然而最终的精度表现还不稳定。对于TDSNN算法，使用者只需要确定$T_{il}$和$L$两个参数。本研究通过实验评估这两个参数对算法精度和运算开销的影响，通过实验还可以确定在保证高精度和低运算量的情况下的最优参数确定策略。总体来看，$T_{il}$影响总运算量的大小，且$T_{il}$影响中间过程存储的时间数据的大小，因此该参数在保证精度的前提下应该尽可能取较小的值。而参数$L$算表面上对运算量不会产生影响，但是其可能潜在地影响最大脉冲时刻，即$T_{il}$的取值可能会受到$L$的影响。

本研究首先确定$T_{il}$的取值情况。考虑计算机硬件存储$T_{il}$的方法，其按照比特位划分可以分为1bit、2bit、4bit、8bit、16bit、32bit定点数来存储，其余的bit位宽会增加后续硬件设计的复杂度。实验发现1bit、2bit两种情况下无法

保持网络的精度损失控制在1%以内，4bit开始网络精度达到最大精度，之后的8bit、16bit和32bit纯属位宽冗余了，因为脉冲时间存储的位宽存在浪费。可见保持网络精度的最低位宽为4bit，即上限数值为($2^4$-1)。可见相对于原始CNN中使用32位浮点类型存储神经元数据，我们只需要使用4比特存储时间信息即可，神经元的存储开销只有CNN的12.5%。

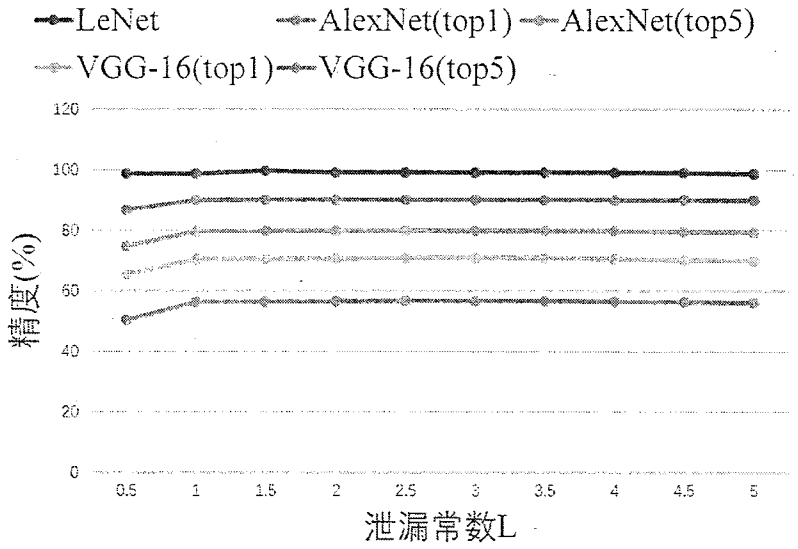


**图 4.6 网络精度与泄漏常数关系图**

**Figure 4.6 Relationship between network accuracy and leakage constant**

保持$T_{il}$不变，本研究通过实验评估参数$L$对网络精度的影响，结果如图4.6所示。研究发现，随着参数$L$的数值从0.5增长到5，LeNet转化后的网络精度只发生了微弱的变化，在98.47%和99.08%之间波动。说明小网络对参数的敏感度仍然很低，参数的微小变化不会对精度产生明显影响。但是在大网络AlexNet和VGG-16中，精度影响开始显现。在$L < 1$时精度已经出现明显的滑坡现象，精度损失远超1%。精度损失已经到了不可接受的程度，说明$L$不可以无限制得小。$L$取1到5之间的数值的时可以保证实验中的精度稳定。

此处增加实验研究$L$取1到5之间这个区间内数值的时候，运算量开销的变化，结果如图4.7所示。由于CNN的开销是恒定不变的，此处使用 SNN 的运算量与 CNN 运算量的比例的变化来呈现这一变化。我们以SNN Ops表示SNN的运算量，CNN Ops表示CNN的运算量，分别统计加法、乘法和运算总量三个量的变化，实验选取Alexnet为参考基准。我们发现随着$L$的增大，运算开销的

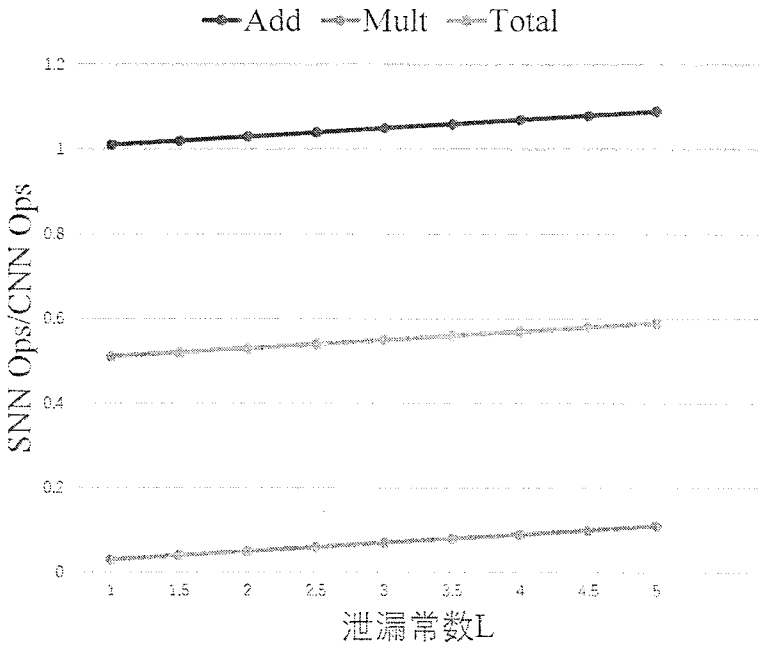


图 4.7 AlexNet运算开销与泄漏常数关系图

**Figure 4.7 Relationship between compute cost and leakage constant in AlexNet**

比例也在逐渐增大。加法的比例落在1到2之间，乘法的比例落在0.03到0.18之间，两者加在一起总运算量落在0.4到0.6之间。最终的总运算量削减波动在区间48.4%到41.7%之间。由公式4.18可知，随着$L$的增加，最大脉冲时间的范围也在扩大，对$T_{il}$的取值要求增大。但是如果我们固定$T_{il}$不变，那么整个网络的运算量也就不会发生变化了。

总的来说，$T_{il}$取值在4比特存储的范围内即可，最便捷的取值是直接定为4比特存储的上界$(2^{16}-1)$。4比特的脉冲时间数据作为每层运算的输入与输出，相比于大多数CNN加速器中使用16位定点的神经元数据，存储和访存开销降低了75%。在$T_{il}$给定的前提下，为了保持最高的精度水平，$L$的取区间1到5之间的值即可。

综上，TDSNN算法中参数对精度和开销的影响得到了充分地评估，同时参数的确定问题得到了解决。

## 4.4 小结

本章在DSNN系列算法的基础上，研究发现基于频率编码的SNN存在明显

的运算开销上的劣势。基于此，本章研究了基于时间编码的脉冲神经网络转化算法。该算法使得转化后的脉冲神经网络，在保持和CNN精度相同水平的情况下，可以实现至少40%的运算量缩减和75%的神经元存储和访存开销节省。研究表明TDSNN算法存在设计低开销的神经网络加速器的潜力。

# 第5章　低开销的神经网络加速器设计

TDSNN转化算法成功实现了卷积神经网络到脉冲神经网络的转化，维持了可接受范围内的精度损失的同时，实现了大幅的访存和运算开销削减。其中，基于时间编码的神经元数据存储能够降低神经元的存储和访存开销。而运算开销的收益能否启发设计更低运算开销的神经网络加速器是个亟待解决的问题。本章的研究将充分利用TDSNN算法的特性，设计更低开销的神经网络加速器。

本章的组织方式如下：5.1节对 TDSNN 算法进行了分析，提取算法的关键特性，并分析了加速器设计的关键点和后续的研究思路；5.2节以 Cambricon 架构的运算单元建立加速器建立参考基准，分析设计了多种运算单元的设计方案与算法部署方案。5.3节讨论了特殊参数场景下的优化效果，多个设计方案都能从特殊参数获益；5.4实验评估了多种设计方案的面积、功耗、能耗、加速比等硬件参数，从而确定最佳的加速器设计方案。5.5节给出本章研究的小结。

## 5.1　研究思路

本节将针对TDSNN算法进行分析，提取该算法带来巨大运算收益的关键特性，然后针对该特性进行加速器设计思路的分析。

### 5.1.1　算法分析

不管是在CNN还是SNN中，网络的核心计算都出现在其中的卷积层和全连接层。根据 TDSNN 算法的描述可知，对于卷积层、全连接层的这种密集计算层在算法上可以分为两个计算阶段，一个是抑制阶段，另一个是时钟神经元激活阶段。根据算法可知，抑制阶段计算得到的第$j$个输出神经元的电位值$P_j$满足公式5.1的描述，而接下来在时钟神经元激活阶段，第$j$个输出神经元脉冲时间$t_j$的计算满足公式5.2的描述。

$$P_j = \sum_{t=0}^{T_{il}} \sum_{i=0}^{N} \omega_{ij} \cdot \exp(\frac{t - T_{il}}{L}) \cdot Spike(i,t) \quad \ldots (5.1)$$

$$t_j = \min_{T} \{P_j + \sum_{t=0}^{T} \omega_{tick} \cdot \exp(\frac{t - T}{L}) \geq \theta\} \quad \ldots (5.2)$$

其中$Spike(i,t)$表示在t时刻第i个神经元有没有发放脉冲，返回值为1表示存在脉冲，为0表示无脉冲。另外$\theta$表示神经元的阈值，在最简单情形下取0。可以发现，在泄漏常数确定的前提下，脉冲时间取值只有16种可能性。因此本研究总结得出结论：

**TDSNN算法在运算开销上的收益来源于对神经元数据进行了有限值量化。**

在上述算法特性的基础上，此处分析了SNN 和 CNN 硬件上部署该算法的方案。对于现有的 SNN 硬件，参考 TRUE NORTH等硬件的实现可以发现，大规模脉冲神经网络的部署依赖大规模的脉冲计算阵列，阵列之间的神经元通过脉冲传递信息。对于大规模脉冲神经网络，现有的 SNN 硬件需要增加额外的运算核来做大小适配，这意味着现有的硬件对于网络规模的可扩展性非常差。假设在硬件资源足够支持网络大小计算的情况下，对于该算法的两个阶段的运算，都必须按照脉冲传递的方式逐个脉冲时间的完成，最长所需脉冲时间步数为$2*T_{il}$。而且两个阶段的无法进行运算上的流水，第二个阶段的运算占有当前的输出神经元以及与其相连接的通路，必须在阶段二结束才可以释放当前的运算通路做下一步的运算。因此直接在SNN硬件上部署TDSNN算法虽然可行，但无法利用该算法的优势。

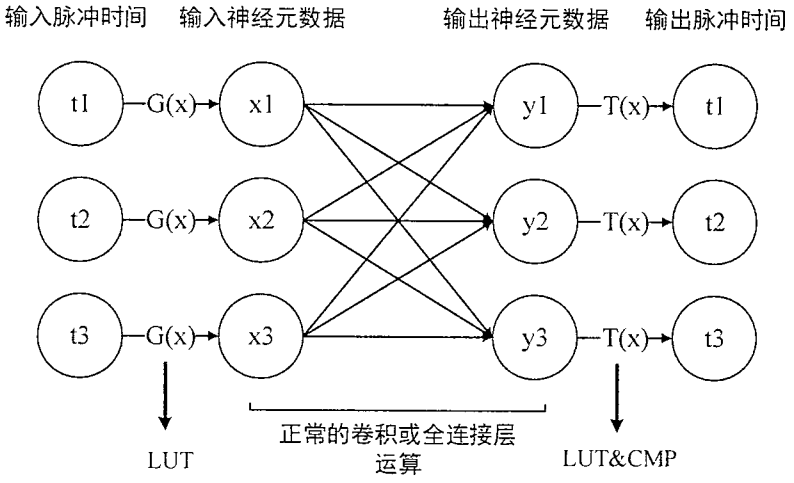


**图 5.1 TDSNN算法核心计算层的部署**

**Figure 5.1 The deployment of the core computing layer of the TDSNN algorithm**

对于CNN硬件，如图5.1所示，按照转化算法的特性，算法部署时无需按照

脉冲神经网络的二阶段运算的方式。反而其运算过程可以等价于图中的运算，首先输入的数据不再是正常的16比特神经元数据，而是4比特的脉冲时间数据。脉冲时间数据无法完成正常的神经元权值的运算，因此需要先完成脉冲时间到神经元数据的转化，由上一章的推导过程可知这一步可以通过公式5.3来实现。但是实际上运算过程没必要像函数描述的那样涉及复杂函数的实现，考虑到脉冲时间4比特的特点，经过函数$G(x)$之后的神经元数据也是有限的。例如可以考虑使用 16 项 16 比特大小的表项存储该结果，即表的构成为$\{\exp(t/L)-1, t \in [0,15]\}$。在使用时直接通过查表即可获得每个时刻t对应的神经元数据，在得到神经元和权值数据的情况下就可以完成正常的卷积层或是全连接层的计算了。为了实现整个网络数据传输神经元数据遵循统一的格式，输出神经元数据需要通过函数$T(x)$转化为脉冲时间数据，参考公式5.4。这一步的非线性函数的运算也可以同样做简化。只需要复用同样的表项，通过将运算结果与表项的比较操作即可获知输出的脉冲时间。从格式上看函数$G(x)$和函数T(x)扮演了原始CNN网络中的激活函数的作用，输出脉冲时间数据也会作为下一层计算的输入脉冲时间数据，这样后续各层的运算也可按照同样方式完成。

$$G(x)=\begin{cases}\exp(\frac{1}{L}\cdot ReluN(T_{il},\lceil L\cdot\ln(x+1)\rceil)) & x\geq 0,\\ 1 & x<0.\end{cases} \quad \ldots (5.3)$$

$$T(x)=ReluN(T_{il},\lceil L\cdot\ln(x+1)\rceil) \quad \ldots (5.4)$$

对于网络中的非核心运算层例如池化层，TDSNN算法已经为脉冲神经网络算法的池化层设计了特殊的计算方法。该方法能够使得脉冲神经网络按照脉冲方式计算池化操作，进而完成在现有 SNN 硬件上的部署。但是抛开脉冲神经网络的时间概念来看，求解一个池化窗口内的脉冲时间的最大值、最小值或是平均值，都可以通过 CNN 的最大值池化、最小值池化和平均值池化来实现。因此 TDSNN 算法中的池化层也可以直接在现有的 CNN 硬件中实现。如图5.2所示，在做平均值池化和最大值池化的操作时，可以类比全连接层的做法，把脉冲时间数据转化为神经元数据再做正常的取平均值和最大值的计算。但是一种更简便的方式是直接对脉冲时间数据做最大值和平均值的运算，其中平均值的运算由于会引入非整数的时间因此在计算输出结果时需要取整，但是实验证明对结果精度不会产生明显影响。

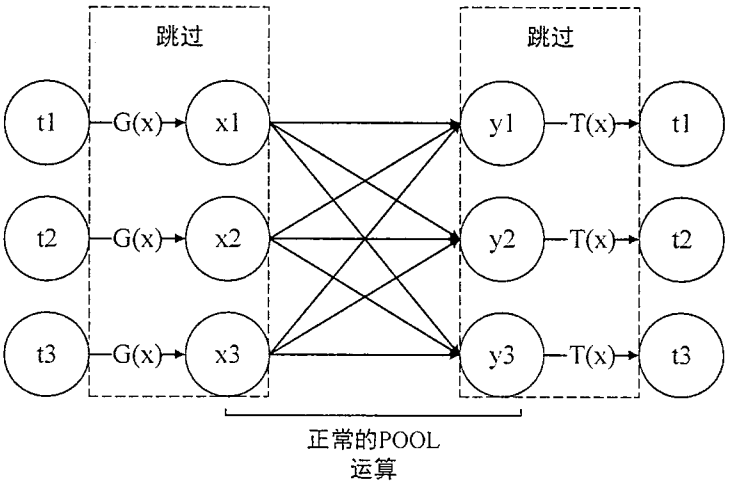


图 5.2 TDSNN池化层的部署

**Figure 5.2 The deployment of the pool layer of the TDSNN algorithm**

### 5.1.2 加速器设计分析

基于上述算法特性的分析可知，TDSNN算法已经可以直接部署到 SNN 和 CNN 硬件上。直接部署可以利用低位宽的神经元时间存储方式降低降低访存和存储开销，但是已有的硬件运算单元从未针对有限值量化进行特殊设计，因此无法充分利用该算法特性来降低运算开销。因此本章的研究将基于此运算特性，重点关注加速器的运算单元设计。本研究通过设计契合算法特性的加速器运算单元，配合更低位宽的访存和存储方式，以实现更低开销的神经网络加速器。

考虑到本研究的核心关注点在于运算单元设计，此处选取了Cambricon[114]架构为加速器的参考基准设计，如图5.3所示，主要有以下原因。首先 Cambricon 架构具备了实现 CNN 网络所需的完备的指令集，能够高效执行多种复杂的任务，而TDSNN算法只涉及神经元运算的改变，本研究无需针对指令集进行重新设计。其次 TDSNN 算法目前未做任何特殊的压缩或量化手段，例如稀疏化和二值化等，因而不会选取在适用于这些算法的加速器作为参考基准。另外，本研究考察了多种设计方案例如DianNao，DaDianNao，Cambricon甚至是服务于稀疏运算的Cambricon-X等，其核心的运算单元实现几乎是一致的。因此作为计算稠密神经网络的高效的设计之一，搭载完备指令集的Cambricon架构足以完成运算单元的设计效果的评估和算法的部署。

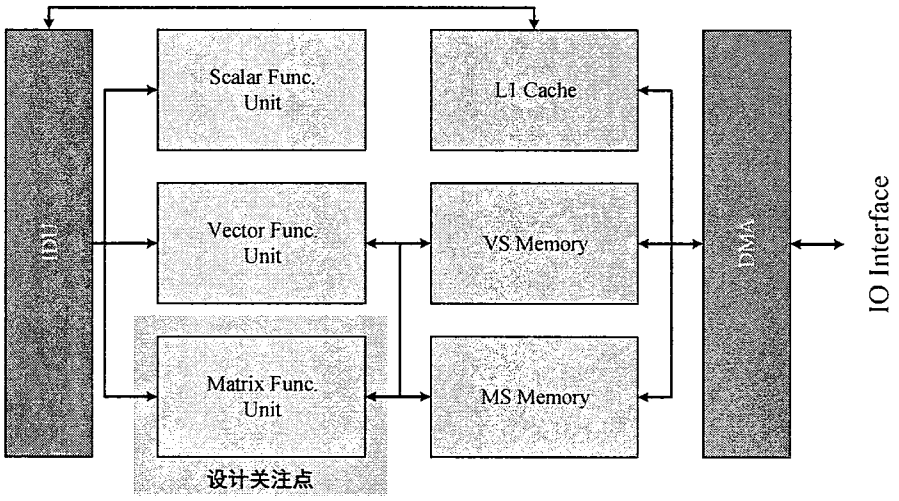


图 5.3 参考基准Cambricon结构图

**Figure 5.3 Overall Structure of Cambricon benchmark**

此外，在上述常见的稠密神经网络加速器中，神经元和权值都已经做了16比特定点数的量化，并被证明没有明显的精度损失，而且16比特的运算器开销相对与于32比特定点或其它浮点数表示存在明显的硬件开销的优势。因此本研究也对权值做了16比特定点数表示的量化实验，并且实验结果表明精度损失仍然控制在了1%这种微不足道的范围内。更低比特定点数的量化不在本文的研究范围内，本研究也不依赖权值定点数的表示位宽，16比特定点表示足以用来评估运算单元的设计效果。

## 5.2　运算单元设计

本节介绍了多种利用 TDSNN 算法特性设计出的加速器矩阵运算单元，以及相应的核心运算层的部署方案。

### 5.2.1　运算单元参考基准

首先确立两个方案为运算单元设计的参考基准。一个是Cambricon架构下原始的运算单元。该设计保留了原先16位定点输入与输出，主要用于计算网络结构调整前的原始网络的计算。原始网络中的神经元和权值数据都是16位定点数数据，并且每一层之间不需要进行脉冲时间和神经元数据的相互转化。该方案下的结构图如图5.4所示，此处将其命名为cambricon-pe-v1。输入由三部分组成，

即控制指令Control Inst、来自向量存储的VM Data和来自矩阵运算单元的MM Data三部分组成，输出数据为计算结果。控制指令选择输入的MM Data和VM Data数据完成对位的向量乘法操作，然后将乘法结果作为加法树的输入完成累加。最后选择部分和寄存器中的中间运算结果与加法树结果做加法，并将结果写回部分和寄存器。如果存在神经元复用，则切换权值使用下一拍权值完成上述运算，再写回另外一个部分和寄存器。如果存在权值复用，则类似地暂存神经权值数据，切换神经元数据完成运算后写回。当达到控制指令指定的循环次数之后，选择部分和寄存器中的数据输出，输出时经过转数模块完成16比特定点数的输出。因为在运算过程中需要支持对输入数据的复用，所以部分和寄存器有RT个，分别存储不同的输出点的部分和结果，具体的算法复用计算方案后续会有详细描述。

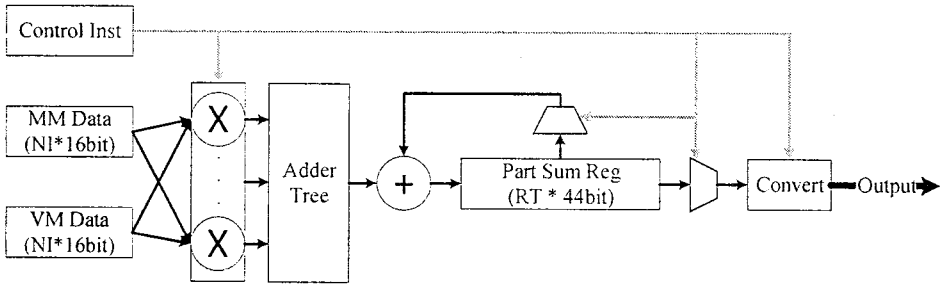


**图 5.4 cambricon-pe-v1整体结构图**

**Figure 5.4 Overall structure of cambricon-pe-v1**

另外一个参考基准是为了部署TDSNN算法所略微修改的版本，如图5.5所示。该方案在输入端和输出端做了不同的处理以最小的变化来支持TDSNN算法。首先该方案增加了表项Leaky Constant Table用于存储16种可能的神经元脉冲时间对应的神经元数据，并增加了查表单元LUT来完成每次输入数据的查表获取神经元数据的操作。权值输入和之前保持一致。在获得神经元和权值数据之后按照cambricon-pe-v1相同的方式进行运算，直到部分和结果全部计算完毕。在准备输出时，不需要通过转数模块来将部分和结果转化为16位定点数输出，而是通过已经获得的Leaky Constant Table的数值与当前结果进行比较，根据比较结果确定神经元数据落在的脉冲时间区间，即可完成脉冲时间的输出。

对于上述两个方案都存在共同的神经元和权值复用计算方案，本研究中该方案被命名为Reuse-v1。此处以卷积层和全连接层的运算为例，来详细说明其

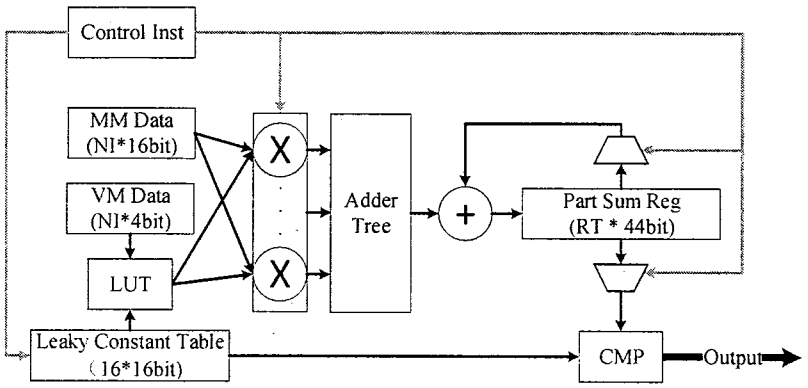


**图 5.5 cambricon-pe-v2整体结构图。**

**Figure 5.5 Overall structure of cambricon-pe-v2.**

详细的运算过程。这里特别对复用方案做详细描述，原因是后续的每个设计方案都涉及到了对复用的考虑，另外不同的设计方案中的复用方案设计也是设计的关键点。

首先是全连接层，该层的运算的特点在于不同的输出神经元对应的权值也是不同的，但是不同的输出神经元共享输入神经元数据，因此在做运算数据复用时只能考虑输入神经元数据的复用。如图5.6所示，此时每个 PE 计算 RT 个输出神经元，PE 内的 RT 个复用寄存器存储的也是不同输出神经元的部分和结果。片上存储通过 H-tree 结构将 VS Memory中的神经元数据广播到每个PE，每个 PE 读取各自部分的存储于 MS Memory 的权值。 PE 内部的运算参考cambricon-pe-v2方案中的描述。对于输入神经元数据量大于 NI 的场景，考虑对存储于片上 SRAM 的数据做分段载入。当片上存储不足以计算完整的输出神经元时，还需要利用 DMA 模块从片外存储载入数据到片上。输出神经元数据和权值数据同理。

然后考虑卷积层的部署。卷积层的特点是同一个特征图像（feature map）内的输出点共用一份权值，但是不同输出特征图像的点的权值不共用。因此一般对卷积层运算的切分方式是使得不同的 PE 计算不同特征图像层的结果，同一个 PE 内的复用寄存器存储同一个特征图像的不同输出点。如图5.7所示，VM Data 为沿不同的输入特征图像层取出的一列神经元数据，MM Data 为对应位置的一列权值数据，在接下来的 RT 拍的运算中复用这一列权值，将神经元在 feature

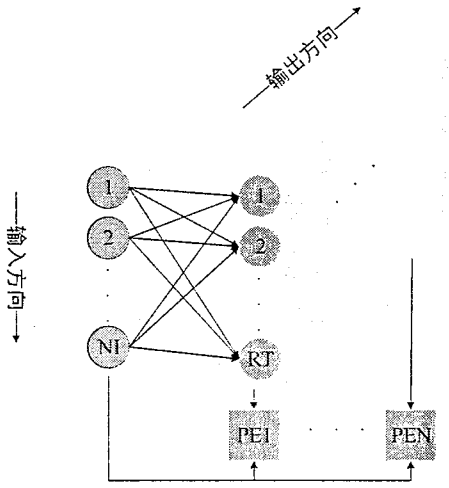


图 5.6 cambricon-pe下的全连接层复用计算示意图

**Figure 5.6 Reused calculation of fully-connected layer under cambricon-pe**

map 内滑动，神经元数据切换的方向为窗口滑动的下一个位置而不是窗口内的下一个位置。切换RT次的过程中权值被复用了RT次。当权值复用RT次之后开始切换权值，总共切换次数为$KX \times KY \times CI/NI$次，其中CI为输入feature map的深度。当权值切换次数全部完成时，RT 个输出点的计算完成。

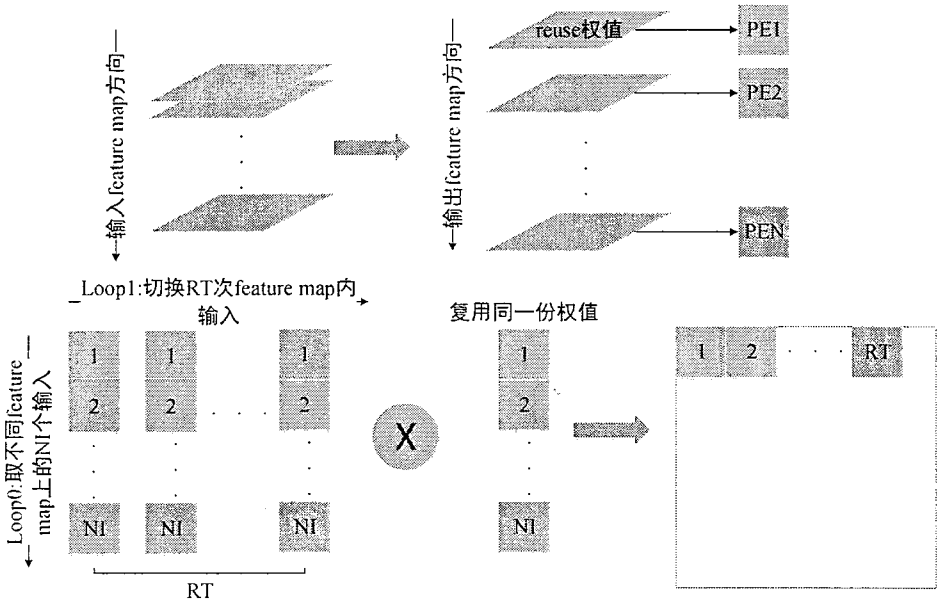


图 5.7 cambricon-pe下的卷积层复用计算示意图

**Figure 5.7 Reused calculation of convolution layer under cambricon-pe**

当然上述的切分和复用方式并非唯一，卷积层计算时 PE 内部也可以完成一定的神经元复用，多 batch 的全连接层计算时也可以完成权值复用，因此如何切分和复用取决于实际的应用场景。但是该设计和后续我们提出的新设计方案都需要考虑复用带来的影响。

### 5.2.2　离散脉冲方案

从本方案开始本研究着手利用TDSNN算法降低运算开销的特性来设计运算单元。分析发现TDSNN算法之所以在理论上可以大幅地削减乘法的开销，主要得益于乘法运算只发生在脉冲时间切换的时刻。利用该思想，本研究首先选择按照正常脉冲神经网络的执行思路。该思路下存在全局时钟的概念，输入脉冲时间决定了神经元参与运算的时间，前一个时刻的神经元全部计算完毕才能计算后一个时刻的神经元的运算。但是对于大规模的网络来说，由于脉冲时间存储于 SRAM 中，只有实际运算单元获得脉冲时间时才可获知当前的全局时钟处于的时刻是否与神经元脉冲时间匹配，从而获知是否需要执行的信息。按照这一思路，本研究设计了第一种运算单元并把该方案命名为离散脉冲方案（discrete-spike-pe），其整体结构如图5.8所示。

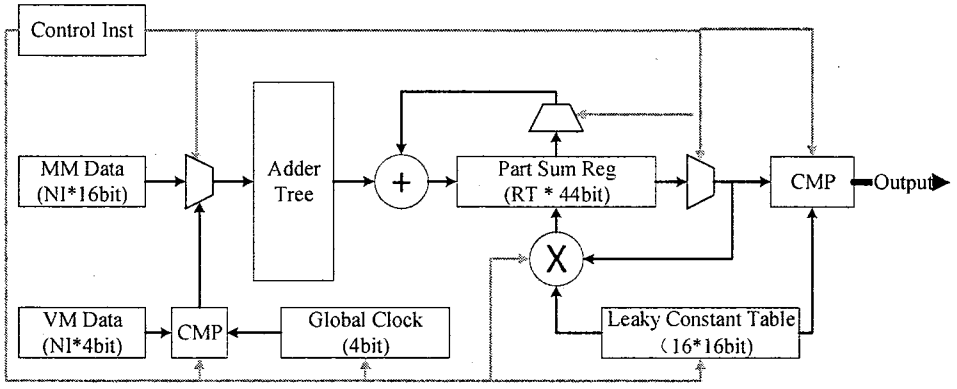


**图 5.8 discrete-spike-pe整体结构图**

**Figure 5.8 Overall structure of discrete-spike-pe**

整个运算单元的数据流运算过程按照如下描述进行。首先根据输入脉冲时间数据VM Data 和当前全局时钟（Global Clock）的时间来产生当前参与电位累加运算的掩码（Mask），掩码数值为0的脉冲不会对电位累加产生作用。假设当前时刻为t，则只对NI个输入中时间为t的脉冲对应位置的权值做累加，加法结果与暂存的电位寄存器（Part Sum Reg）做累加。出于复用的角度考虑，Part Sum

Reg 也存在 RT 项。在每个时间拍结束的时候，新的时间拍到来时，暂存在Part Sum Reg中的电位会乘以预设的泄漏常数。由于整个运算过程每一拍只可能占用一个Part Sum Reg的更新，因此在结束时的泄漏乘法只需要使用1个乘法器即可完成流水的运算。最终整个Global Clock的时间从0到15推进完，运算也会结束同时将结果写出。在输出时，数据与预先配置的leaky Constant Table做比较操作确定输出的神经元时刻。在该方案下，卷积和全连接层的访存和运算的复用计算方式仍然采用Reuse-v1的方案。可以发现这种方案下神经元和权值需要重复载入16次，并且整个运算过程需要重复16遍。该方案在运算方式上与已有的脉冲神经网络硬件非常相似，但是由于整体架构的不同，会导致脉冲时间的访存方式和触发计算的方式存在明显区别。显然由于运算部件的大量减少，该方案在设计上的面积与功耗会有一定的优势，但是重复16次的计算又会使该方案在速度与能耗等方面存在劣势。

### 5.2.3 连续脉冲方案

和离散脉冲方案相对的，可以考虑将每一个时间步中的电位数值单独存储，在最后输出时统一做泄露常数的处理。这么做既利用了TDSNN中乘法只发生在时间点更新的思想，又使得脉冲时间的访存和运算不需要重复16次，本研究将该方案命名为连续脉冲方案，如图5.9所示。

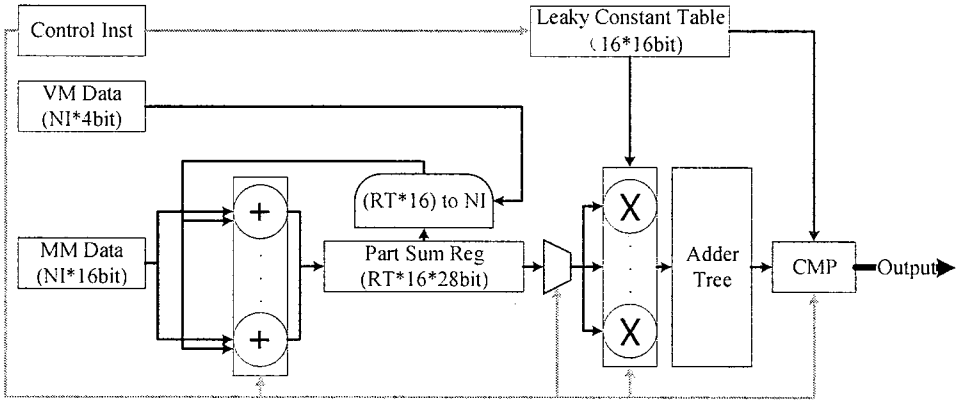


**图 5.9 continous-spike-pe整体结构图**

**Figure 5.9 Overall structure of continous-spike-pe**

整个运算单元的数据流及运算过程描述如下。首先根据当前的神经元时间

数据，选择部分和寄存器存储的对应时刻的当前数值，即从RT*16个部分和结果中选择NI个数据。选择得到到NI个输入数据，和当前的NI个权值数据完成向量加法，并将累加结果写回到对应的部分和寄存器中。由于部分和寄存器需要存储16个时刻的部分和数据，其规模已经扩大到了离散脉冲方案的16倍大小。在完成控制信号规定的全部复用与计算的循环后，即可得到RT个输出点对应的16个时刻的全部部分和数据。控制指令此时逐拍选择每个输出点的16个时刻的数据，完成与Leaky Constant table中数据的向量乘法运算，并将乘法结果通过加法树累加得到最终的部分和结果。最终的部分和结果会按照类似离散脉冲方案的方式，通过查表比对完成脉冲时刻的输出。整体上来看，该方案与参考基准方案的最大区别是乘法和加法树单元被后置到了整个运算单元的后半部分。

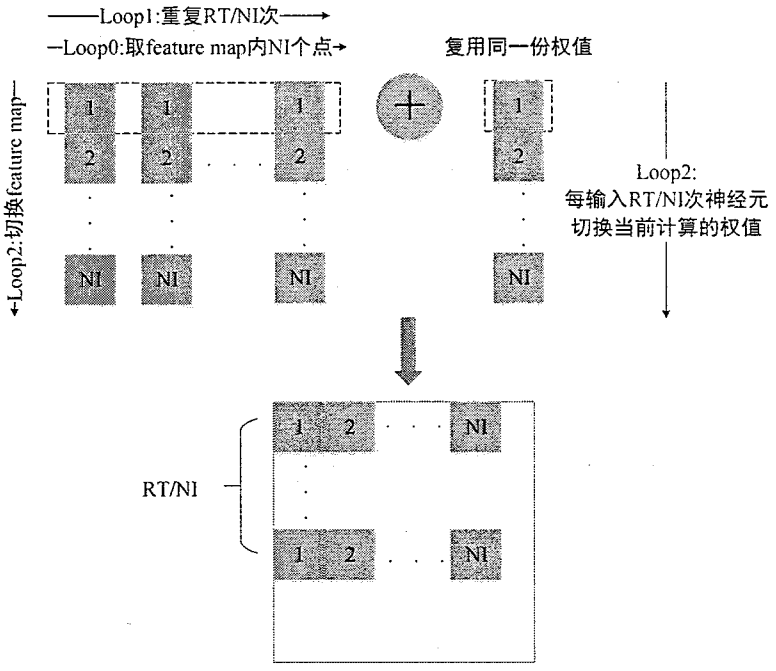


**图 5.10 continous-spike-pe中的卷积层权值复用**

**Figure 5.10 Synapse reuse of convolution layers in continous-spike-pe**

该方案的设计关键点在于如何完成复用方案的计算。如果是按照Reuse-v1的计算方案，输入的权值数据MM Data应当属于同一个输出点的输入，理应做加法树的累加，而不是图上的向量加法单元的运算。针对每次输入的NI个权值数据，需要根据脉冲时间的不同分别计算，并输出到保存不同时间结果的寄存器中。而这一步的计算对于16个时间拍每个输入就有16种可能性，在不增加复杂筛选逻辑的情况下则需要16个16输入的加法树来完成运算。为了降低硬件资源

的开销，此处使用了16个加法器代替16个16输入的加法树，每个加法器用来计算NI个输入，该种设计的前提是NI个输入数据计算的是NI个输出点。考虑到复用方案中的复用次数为RT，当满足$RT\%NI = 0$时不会出现硬件计算资源浪费，否则输入中存在部分数据对计算结果无贡献的情况发生。基于此，本研究设计了新的复用计算方案Reuse-v2，此处以卷积层和全连接层为例分别说明如何做神经元复用和权值复用计算。

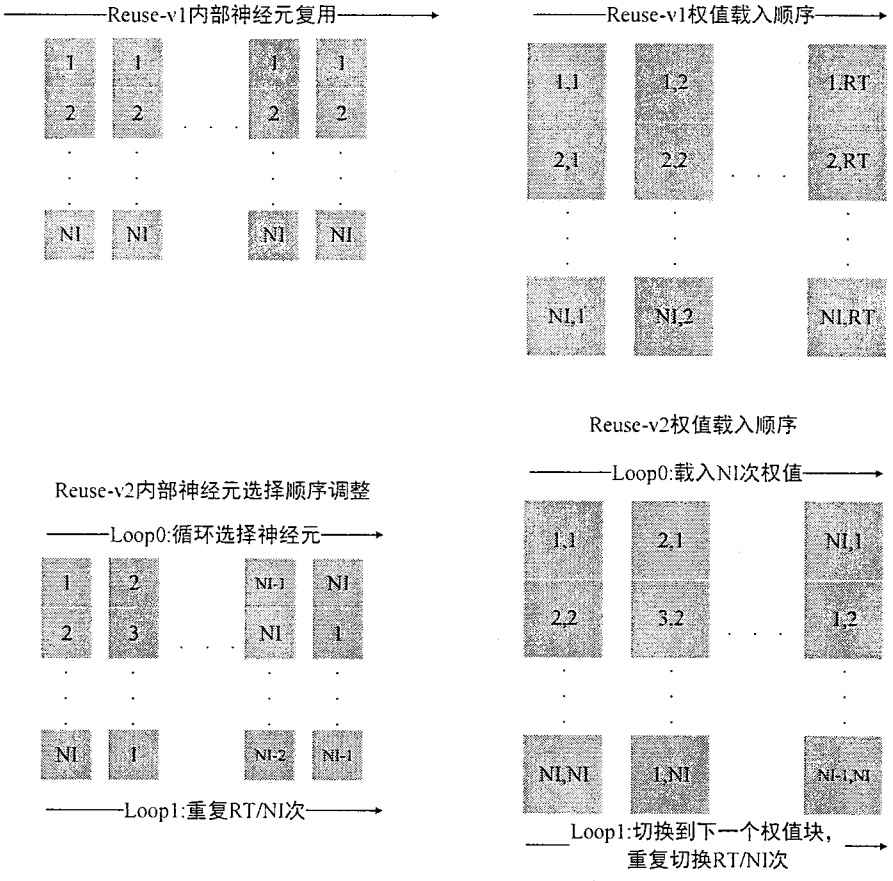


**图 5.11 continous-spike-pe中全连接层的神经元复用**

**Figure 5.11 Neuron reuse of fully-connected layers in continous-spike-pe**

对于卷积层的复用计算，如图5.10所示，此处以该图说明如何在连续脉冲方案下做权值复用。和图5.7中的计算方案进行对比可知，为了完成每个输入点计算不同输出点的同时也保持 RT 次神经元复用，此处不再沿着feature map的方向（有时也称作Channel方向）读取NI个神经元数据，而是更改为沿着feature map内取NI个点。复用的NI个权值依然只需要1拍即可完成载入，但是每一拍只会使用到1个权值，NI个神经元中的每个数据根据当前的脉冲时间选择对应的输出寄存器，将输出寄存器数据与该权值做累加，并输出到存储对应脉冲时间的中间

寄存器中。然后在接下来的NI拍中，无需载入新的权值，只需要切换当前NI个权值的选择，选择其中下一个权值完成权值复用。每次切换NI个权值内的选择时，切换神经元的feature map，直到切换NI次feature map此时NI个权值的复用完成，需要重新读取权值数据进行后续的循环。每$\lceil RT/NI \rceil$拍切换一次当前选择的权值输入，重复进行上述计算。如果滑动窗口内的神经元没有读取完，继续循环$KX \times KY$次的上述过程。如果 feature map方向的输入神经元循环没有结束，则继续按照上述步骤完成feature map方向上的输入神经元和权值的切换，切换次数$\lceil CI/NI \rceil$。

全连接层的复用计算方案如图5.11所示，图中展示了如何在运算单元内部做神经元复用，此神经元复用是在广播神经元到每个 PE 的基础上做神经元复用，这点和Reuse-v1保持一致。对比cambricon-pe的神经元复用方案与此处的设计方案即可发现二者的区别。在前者的神经元复用中，对于正常取出的NI个神经元数据，将会复用RT拍保持使用顺序的一致。由于需要每拍切换权值，此处给每个权值定义坐标$(x, y)$，其含义是第$y$个输出神经元的第$x$个权值。前者的权值数据载入顺序是按照输出神经元的出现顺序，逐拍取出对应的权值数据完成运算。但是在Reuse-v2方案中，复用力度被进一步细化，此处的一次复用循环针对的是一个$NI \times NI$的数据块。该方案同样在满足$RT\%NI = 0$的关系下才能达到运算资源无浪费，整个复用块的计算将循环执行$\lceil RT/NI \rceil$次。对于复用的神经元数据，此处标记为1到NI共NI个数据。在复用块中的NI拍计算中，每拍对神经元的选择按照轮转的方式进行取数。对于权值数据，为了满足每个神经元计算不同的输出点，权值的载入顺序需要调整为图中描述的顺序。块中运算的第一拍载入位置为$(1, 1)$、$(2, 2)$到$(NI, NI)$共NI个权值，这样保证了对位运算之后计算出的是不同的输出点。以此类推，每拍随着神经元的轮转切换下一次的权值输入，直到NI拍结束即可遍历完1个$NI \times NI$的块大小的权值，结束1个复用块的计算。但是总复用次数为RT，意味着完整的一次复用运算总共会出现$\lceil RT/NI \rceil$个复用块。所有这些复用块计算结束后，即可切换神经元进行同样一批输出点的下一批输入神经元和权值的运算。该方案对于权值在存储中的摆放存在要求，但是考虑到神经网络的权值为离线训练得出，并且本研究没有支持在线训练的需求，因而权值完全可以离线训练之后按照既定格式摆放，并载入到片上进行计算。

综上，连续脉冲方案方充分利用了TDSNN算法削减乘法运算开销的特性。虽然无法彻底消除乘法器资源，但是乘法运算和加法树的运算只发生在输出结果时，计算频率得到了大大降低，有望实现运算开销的降低。

### 5.2.4 查表方案

不管是离散脉冲方案还是连续脉冲方案，都是试图直接利用TDSNN算法逐个时间拍计算以降低乘法开销的特点。根据前文的分析可知，该特性的本质是对神经元数据进行了有限值量化，即神经元的取值限制在了16种范围内。本节将利用这一特性，设计查表方案来实现低开销的运算单元。

#### 5.2.4.1 查表方案1

考虑到权值复用时无需切换权值，只需要切换神经元数据的特点。如果能够以有限种神经元为查表索引，建立表项存储每种神经元与当前权值的乘法结果，则能够实现用查表操作代替每次神经元权值的乘法运算。基于该设计思路，本研究提出了第一种查表方案，并命名为lut-pe-v1，具体的结构图如图5.12所示。此处分别对权值复用和神经元复用两个部署方案，说明整个硬件结构的数据流和控制流程。

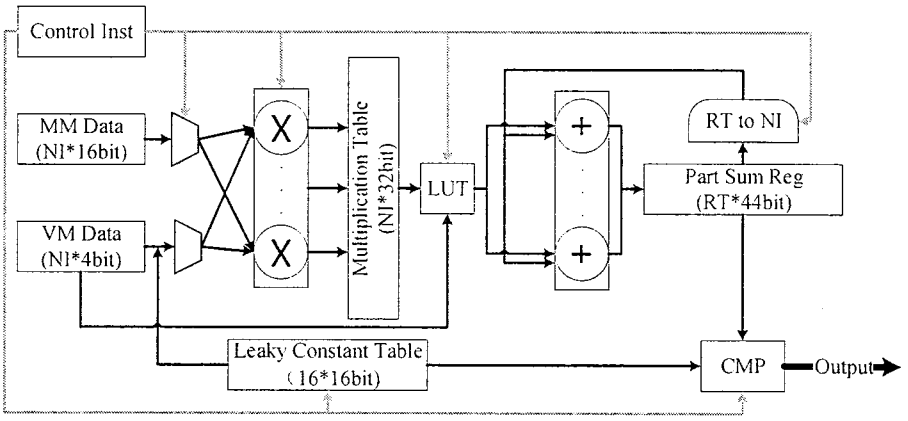


**图 5.12 lut-pe-v1整体结构图**

**Figure 5.12 Overall structure of lut-pe-v1**

在权值复用的场景下，采取的复用读数方案为连续脉冲方案中提出的Reuse-v2方案。对于输入的NI个权值数据，每个权值进行单点复用。运算单元内利

用16个乘法器，实现单个权值和Leaky Constant Table中的16个常数作为输入的向量乘法，从而计算出16个表项结果。输入的每拍NI个神经元数据将直接通过LUT模块，完成对16个权值表项的查表操作。查表获得16个乘法结果将输入NI个加法器，加法器的另外一端输入是从RT个部分和寄存器中选出的当前NI个目标输出点的部分和结果，加法结果将被逐个写回到各自的部分和寄存器。输出部分和结果的最终输出过程与前述的连续脉冲方案等方案一致，同样通过查表比较获得输出的脉冲时间。

对于神经元复用的流程，不考虑按照先计算表项后查表的方式进行计算。因为利用神经元做索引的查表方案，运算器存在逐拍供应权值的需求。此时若也对每个权值计算表项则会阻塞权值的载入，无法完成计算的流水。因此该方案保留了向量乘法部件来支持神经元复用时的向量乘法计算。神经元时间数据输入时将直接使用Leaky Constant Table获得当前的真实神经元数据，每次查表只用于完成脉冲时间到神经元数据的转换。转换后的神经元数据与当前的权值数据完成对应位置的乘法，NI个乘法结果与部分和寄存器中选择的NI个加法结果完成加法操作并写回，后续流程保持不变。

在复用计算方式上此处选择采用Reuse-v2的方案的原因如下。如果采用Reuse-v1方案，则需要一次性存储NI个权值的16种乘法结果，共需要NI*16*32bit庞大的存储。而采用Reuse-v2中的方案在做权值复用时，表项只需要维护1个权值的16个结果即可，前者所需存储大小是本方案的NI倍。另外，如果采用Reuse-v1方案，在表项完成之前无法进行神经元数据的载入查表计算，导致整个运算通路的神经元数据的载入受到阻塞，运算流水线也随之阻塞。

#### 5.2.4.2　查表方案2

查表方案1中，表项的计算放在了运算单元内部进行，因而需要额外的乘法器资源来计算表项。随着片上的运算单元的增多，这些额外的运算单元都会实时地计算表项，因此我们考虑将计算表项的硬件资源集中到向量运算单元，而不是独立分布在矩阵运算单元的方案。针对该设计原则，本研究提出了第二种查表方案并命名为lut-pe-v2，其运算单元结构如图5.13所示。

本方案下来自向量存储部件的输入主要用于传输表项，矩阵存储部件的输入MM Data用来传输数据，可以是神经元数据也可以是权值数据，具体取决于复用计算的对象。由于表项的计算被转移到向量运算单元执行，矩阵运算单元内

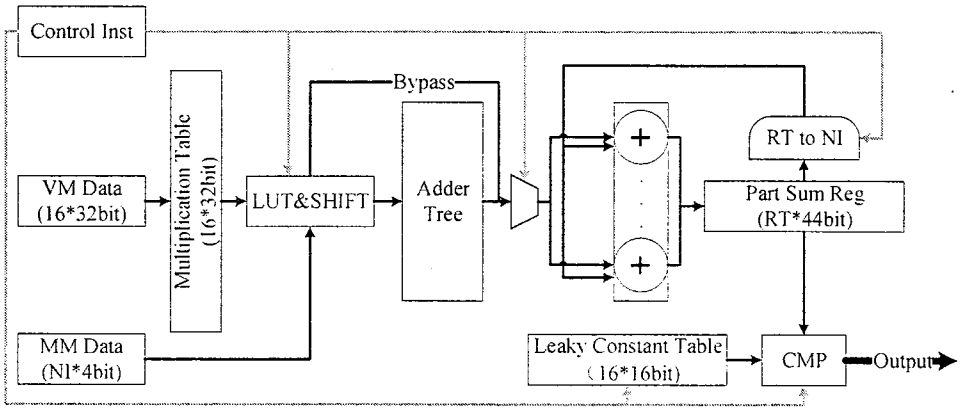


**图 5.13 lut-pe-v2整体结构图**

**Figure 5.13 Overall structure of lut-pe-v2**

部没有多余的乘法器资源。考虑到查表方案1里需要乘法器资源以支撑神经元复用特性的计算，例如全连接层的计算，本研究也开发了新的复用计算方案Reuse-v3来支撑此类运算。在该复用计算方案下的权值复用计算和查表方案1类似，此时每个权值的表项直接存储在运算单元内部。每次进入新的NI个神经元数据仍然计算NI个不同的输出点，根据当前的神经元脉冲时间数值查表获得当前权值对应的乘法结果。查表获得的NI个结果与部分和寄存器中选取的NI个部分和做累加并写回。当处于复用权值的循环中时，切换下一批神经元继续查询相同的表项，同一个权值的表项复用次数为$\lceil RT/NI \rceil$次，循环结束时切换下一个权值对应的16个表项，即切换输入的VM Data。以此类推完成RT行神经元输入和1行权值输入的复用计算，后续循环计算继续按照同样的模式进行计算直到RT个输出点的输入遍历完毕。

但是对于神经元复用的场景例如全连接层的计算中，直接计算每一批权值的表项既会导致计算的流水线出现阻塞又会导致乘法的运算数目扩大到16倍。因此为了解决全连接层难以计算的问题，本研究也为该种计算模式设计了查表计算的方案Reuse-v3。考虑到对权值复用场景下的16个表项做复用，这次的表项用于存储神经元固定的情况下，其与权值的每个比特的运算的累加结果。如图5.14所示为神经元复用计算的示意图。对于4个16比特的权值数据与4个固定的神经元数据的乘累加运算，此处对权值数据按照逐个比特进行拆分，每个比特的数据先和神经元数据做乘法再累加，将累加后的16个比特的结果通过逐

项移位操作得到16个移位后结果，这16个结果全部累加得到的加法结果即是原始4个神经元和权值输入的乘累加结果。这其中，每个比特的神经元数据的乘累加结果刚好只有16种可能性，可以预先通过向量运算单元完成表项的计算。可见，由于每拍输入4个权值和4个神经元，全连接层的计算效率降低为正常情况下的$4/NI$。特别的，当NI为16时，矩阵数据通路刚好每拍载入的权值数目为4，计算效率降低为原先的1/4。对于NI小于16的情形，运算单元无法支撑4个权值的查表计算，因此不推荐再该种参数下使用该方案。

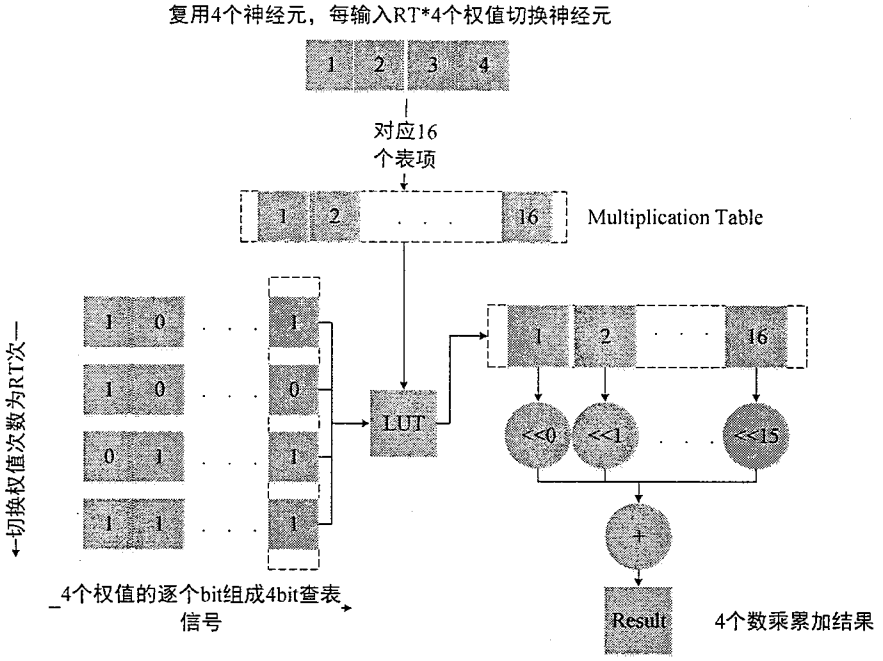


**图 5.14 Reuse-v3的神经元复用**

**Figure 5.14 Neuron reuse in Reuse-v3**

显然直接对 lut-pe-v2 做面积功耗评估是不公平的，因为其计算表项的功能被转移到PE外部的硬件执行。为了做实验的对比需求，本方案也实现了表项计算的硬件lut-pe-v2-cal-table如图5.15所示。该硬件计算出的神经元或权值表项被所有 PE 共享，因此全局只需要部署一个表项计算单元。该硬件的工作原理如下，由控制信息决定计算神经元表项还是权值表项。权值表项主要用于卷积等权值复用计算场景，而神经元表项主要用于全连接层的神经元复用计算场景。为了满足理想状态下的数据传输效率要求，输入神经元和权值数据在两种场景下的带宽保持和前述PE一致。控制指令指定做权值表项计算时，将权值作为乘法器一端输入，另外一端输入为泄漏常数表存储的常数，完成16次乘法

运算后得到1个权值的表项，以此类推逐拍输出每个权值的全部表项。对于神经元表项的计算，整个过程不需要对4个神经元之间的所有运算做累加运算操作。实际上在4比特的数据的所有16种组合情形中，对于0000的组合可以直接输出0，对于只出现1个1的4种场景直接输出对应位置的神经元查表数据，对于出现2个1的6种场景需要调用6个加法器完成6个表项的计算，对于3个1的4种场景需要调用4个3输入的加法树完成运算，最后对于全1的场景调用4输入的加法树完成运算。这样每一拍输出4个神经元之间的16种运算结果。计算后的表项通过数据通路广播到每个PE中。

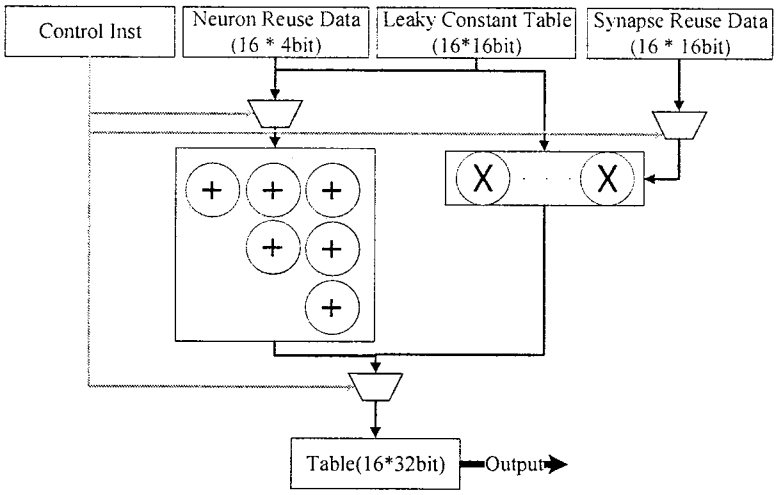


**图 5.15 lut-pe-v2-cal-table整体结构图**

**Figure 5.15 Overall structure of lut-pe-v2-cal-table**

#### 5.2.4.3 查表方案3

查表方案2对全连接层这种利用神经元复用特性来计算的层，其计算效率降幅太大。有一个原因是用于传输表项广播数据通路位宽固定，另外一个原因是来自矩阵存储的数据通路数据利用率只有4/NI。因此本研究考虑设计新的权值载入方案Reuse-v4来最大限度地使用数据，避免浪费。

如图5.16所示，此处使用坐标(x,y,z)来表示第x个输入神经元与第y个输出神经元相连接的权值的第z个比特。可以看到，由于将每4个连续输入神经元与一个输出神经元的4个连接的权值组成一组，每拍输入共有NI组。但是每拍并不是将权值完整地载入，而是逐个对比特位按照时间方向上拆分位16拍。这样恰好

每拍的输入位宽为$NI \times 4bit$。对于神经元表项的载入，一组4个输入神经元计算得到16个表项从广播数据通路VM Data载入。这样完成RT个输出点的全部复用计算，相对于参考基准，效率降低到了其1/4。考虑到对于lut-pe-v2而言，其在NI大于等于16的情况下效率最多为参考基准的四分之一，本方案算是在运算效率上做了初步的优化。并且在计算时仍然需要RT被NI整除才能够达到运算资源的最大力度的使用，实现无计算资源的浪费。权值复用的计算方案仍然按照Reuse-v3种的方案进行，这里不再赘述。

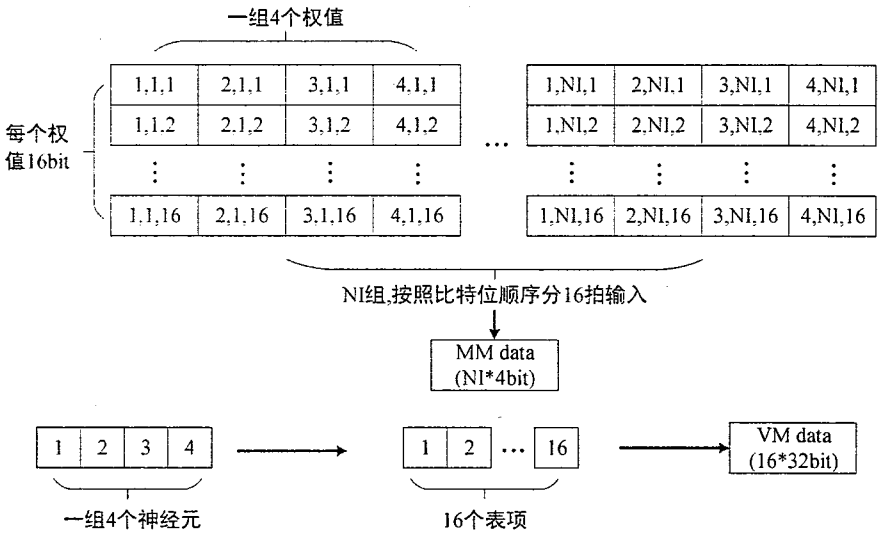


**图 5.16 Reuse-v4神经元复用方案**

**Figure 5.16 Neuron Reuse in Reuse-v4**

随着复用计算方式的改变，此时硬件计算单元的设计方案也随之发生改变。此处维持表项计算单元不变的情况下，重新设计硬件计算单元如图5.17所示，该方案被命名为lut-pe-v3。输入的表项存储在Multiplication Table中，共16项。在神经元复用的计算场景下，权值按照逐个比特的顺序载入，因此每个比特的中间结果都需要存储在Part Mult Reg中。每次权值的输入分NI组利用LUT模块完成查表操作得到NI个结果，NI个结果分别与原始存储在Part Mult Reg中的结果完成移位加法操作。这样重复16拍得到完整的乘累加结果后，按照前述方案中对部分和结果的相同处理完成后续的操作。在权值复用的计算场景下，LUT模块根据当前的神经元数据查找乘法结果，通过Bypass通路跳过神经元复用的计算部分，进而与部分和直接做后续的累加操作。整个模块的输出结果的过程与前述方案

类似，这里不做描述。

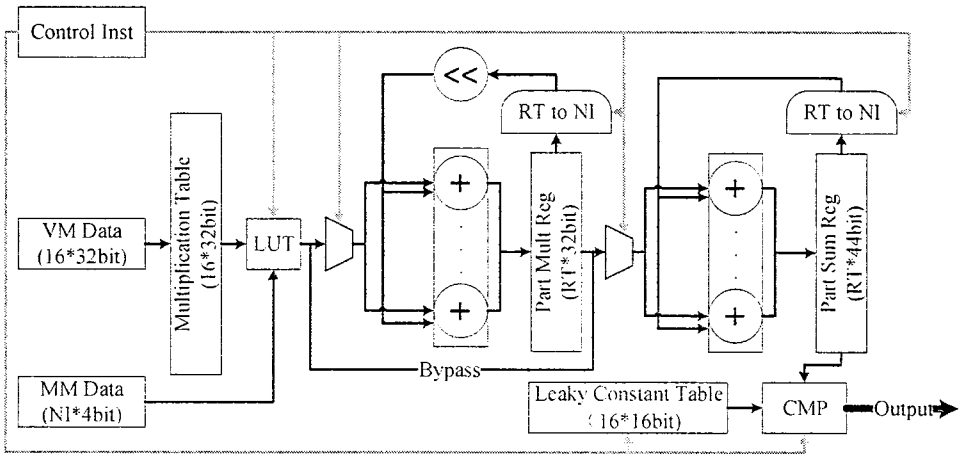


**图 5.17 lut-pe-v3整体结构图**

**Figure 5.17 Overall structure of lut-pe-v3**

至此，本研究完成了多种针对TDSNN算法在Cambricon架构下的算法部署和运算单元设计方案的设计，这些设计方案既考虑了利用TDSNN算法的特性，又充分考虑了硬件资源的节省。

## 5.3 参数特殊化场景

上述方案主要关注TDSNN算法在通用的泄漏常数配置下的部署方案。此处考虑其中存在的一种参数配置场景，该场景能够极大地简化上述设计方案并带来硬件开销上的收益。考虑$\exp(1/L) = 2$即$L = 1/\ln 2$时，此时的泄露常数表失去存在意义，因为不管是神经元权值的乘法操作还是根据神经元索引权值表项的操作，在移位运算面前都存在明显的硬件开销劣势。在算法精度上，上一章的实验表明在该参数下精度损失仍然微乎其微，因而不存在精度问题。以下将详细描述在这样一种底数为2的特殊参数优化下，产生的各个变种设计方案。

首先考虑对参考基准方案的影响，如图5.18所示。相对于cambricon-pe-v1和 cambricon-pe-v2，此时神经元和权值之间的乘法操作，亦或是根据泄露常数表查表获得神经元数据完成与权值的乘法操作，都可以转变为权值数据的移位操作，移位的数目等于当前的脉冲时间。移位得到的NI个结果经过加法树完成累加操作并与部分和寄存器做加法运算写回。在输出时可以根据当前的部分和结果，直接进行逻辑判断得到当前的数据对应的输出脉冲时间，这一步我

们用Convert模块来表示。两个参考基准方案在参数特殊化场景下都转化为了方案cambricon-pe-b2，可以看到整个设计得到了简化，相应的面积与功耗也应该存在优势。

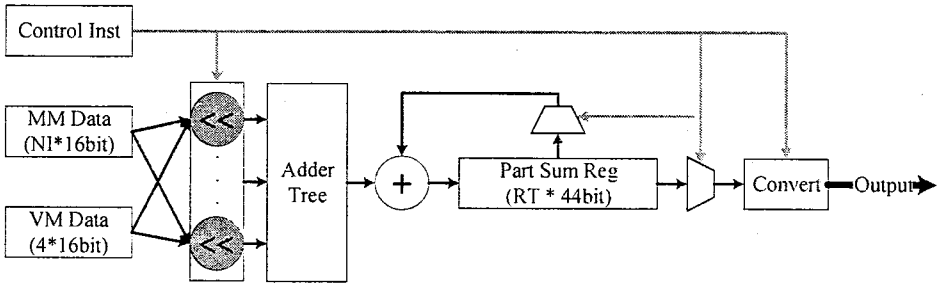


**图 5.18 cambricon-pe-b2整体结构图**

**Figure 5.18 Overall structure of cambricon-pe-v2**

对于离散脉冲和连续脉冲方案，由于这两种方案对于乘法操作的处理类似，都是集中在输出时对不同时刻的结果做泄露效应的乘法处理。如图5.19和5.20所示，相对于原有方案的一个变化是把乘法器换成了移位操作。其中离散脉冲方案中，移位操作的数目固定为1，开销相对更低。另外，由于在计算泄露效应和输出结果脉冲转化时不需要泄露常数表Leaky Constant Table的存在，硬件也无需再维护该表项。其余部件和运算流程保持不变。

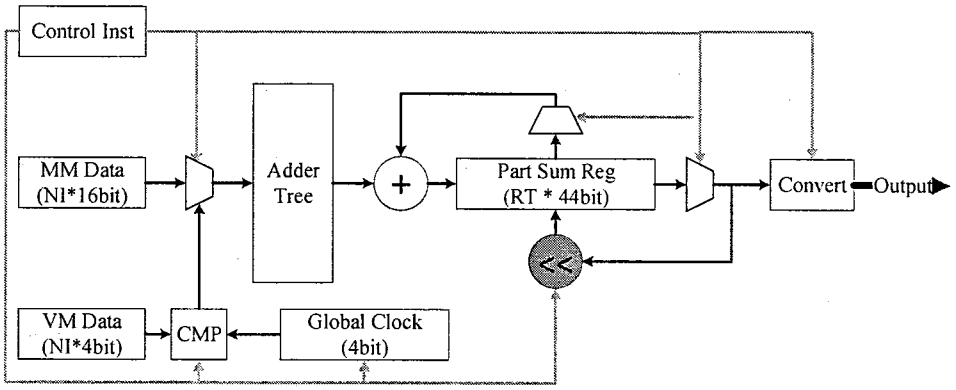


**图 5.19 discrete-spike-pe-b2整体结构图**

**Figure 5.19 Overall structure of discrete-spike-pe-b2**

最后考虑查表方案的变化。虽然在前一节里本研究提出了三种查表方案，但是在参数特殊化场景下，后两种查表方案并无太大优化空间。原因有两个，

一是底数为2的优化只会导致表项计算单元开销的缩减，并不会直接影响矩阵运算单元；二是此时计算神经元或权值的表项，并利用查表操作代替乘法的优势不再明显，大部分的开销削减已经被乘法器到移位的转化替代。因此本研究不针对查表方案2和查表方案3做底数为2的优化。而对于查表方案1，底数为2的优化不仅将原先的乘法器替换为了移位运算器，对于原先的查表操作也将全部去除，留下的移位运算过程与cambricon-pe-b2一致。它们的区别在于后续的逐个加法操作仍然得到了保留，而不是使用加法树，并且在复用计算方案上采用Reuse-v2的方案。

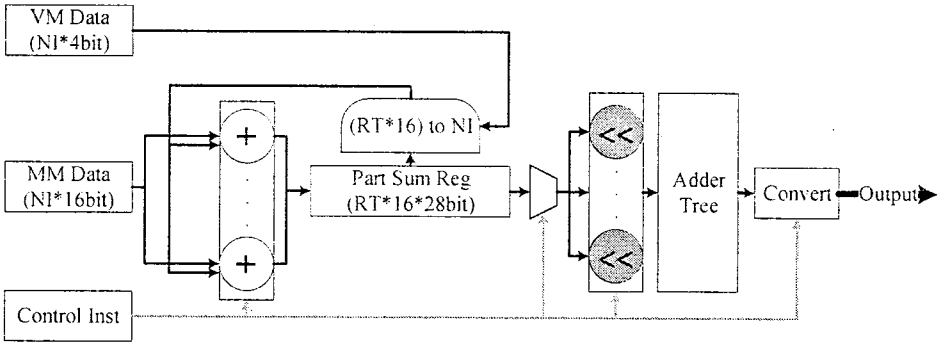


**图 5.20 continous-spike-pe-b2整体结构图**

**Figure 5.20 Overall structure of continous-spike-pe-v2**

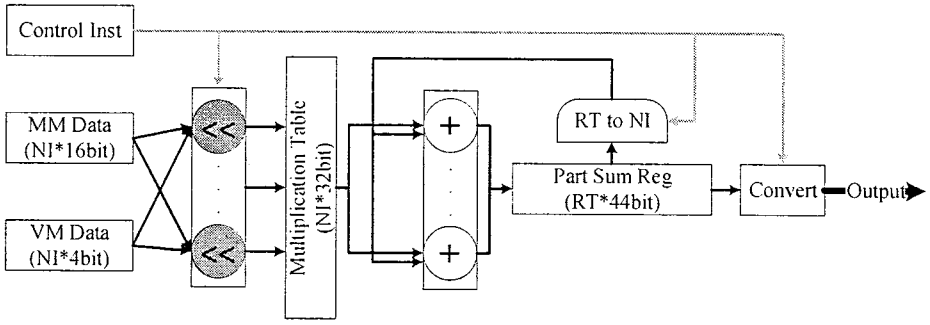


**图 5.21 lut-pe-v1-b2整体结构图**

**Figure 5.21 Overall structure of cambricon-pe-v1-b2**

需要明确该种底数为2的优化是TDSNN算法的一种极端参数场景，在目前算法测试的多个网络中对精度的影响微弱，但是如果对参数配置的可拓展性存在要求的场景，就不适合该种优化方案了。

综上，本研究根据TDSNN算法设计得到了多种可行的硬件部署方案。针对这些硬件的具体实际效果，本研究将在接下来的实验与评估一节中进行进一步分析。

## 5.4　实验与评估

本研究的实验按照如下思路进行规划。首先对两个参考基准方案进行面积和功耗的评估，以获得参考基准的关键硬件参数。然后对各个运算单元设计方案进行面积和功耗上的评估，包括特殊参数场景下的优化效果。紧接着本研究评估了各个设计方案下的神经网络性能效果，最后结合在典型网络上的能耗结果给出最优的设计方案结论。

在本节的实验中，本人采用硬件设计语言Verilog完成设计，采用TSMC 28 nm工艺库，采用Synopsys Design Compiler进行综合，后采用Synopsys IC Compiler进行后端的布局布线设计。针对性能和功能的验证与评估，我们搭建了C++语言实现的时钟精确的模拟器，能够完成基于神经网络指令集的功能与性能评估。

### 5.4.1　参考基准评估

对于包含参考基准在内的设计方案，本研究的实验中将参数NI定为16，RT定为16，本研究认为这是能够公平评估所有方案的一组参数，原因如下。首先对于 cambricon-pe-v1、cambricon-pe-v2、discrete-spike-pe而言，参数NI和RT可以自由取值，但是对于其余的方案由于复用计算方案的差异，由前述的分析可知必须满足RT被NI整除且都大于等于16时运算效率达到最大，否则会出现资源浪费。其次对于查表方案lut-pe-v2和lut-pe-v3而言，NI越大在神经元复用时的计算资源利用率越低，因此为了评估这两个方案在最优工作效率的下的硬件开销，在实验时尽可能地缩小NI的取值。另外参考DianNao、Cambricon和Cambricon-X等工作可知，主流的工作里对运算资源的评估时使用的运算单元输入NI多数情况下16已经足够，且在大多数据集中该参数下的计算资源利用率相对其它参数较高。

基于此，本研究首先对cambricon-pe-v1和cambricon-pe-v2的硬件参数做初步的评估，实验结果呈现在表5.1中。通过对比两个方案的结果可以发现，如果通过直接对 cambricon-pe-v1微调整的方式来得到的设计方案，其面积功耗相对

于原始的方案是明显增加的。这是符合预期的表现，因为后者相对于前者增加了查表和表项的开销，在硬件资源开销上比较明显地多于前者。但是这并不意味着包含运算单元在内的整个芯片的能耗开销上后者高于前者，因为只有cambricon-pe-v1仍然使用了16比特的神经元资源，其余方案都使用了4比特的脉冲时间表示的神经元，在访存和存储开销上后面的方案要远低于前者，因此前者更适合作为使用TDSNN算法前的参考基准，用于对比算法本身带来的设计效果。后者适合作为使用TDSNN算法的设计方案的参考基准，用于评估设计方案的优劣。

**表 5.1 参考基准设计方案硬件参数**

**Table 5.1 Hardware characteristics of benchmark designs**

| 设计方案 | cambricon-pe-v1 | cambricon-pe-v2 | lut-pe-v2-cal-table |
|---|---|---|---|
| 面积($\mu m^2$) | 2438.2625 | 2995.2634 | 1795.418 |
| 功耗(mW) | 1.6572 | 1.9357 | 1.6783 |

考虑到查表方案lut-pe-v1和lut-pe-v2的表项计算单元并不存在于矩阵运算单元内，而直接评估运算单元的开销又会使得比较丧失公平性，本研究实现了表项的计算单元lut-pe-v2-cal-table。该单元的开销也被列在了表中，这项参数将会被用于修正两种查表方案的面积和功耗开销。修正系数为1/16是考虑到原始的Cambricon设计中矩阵运算单元的运算单元PE的个数为16，这些PE共用一个表项计算单元，因此需要乘以系数之后修正硬件开销。

### 5.4.2 设计方案面积功耗评估

此处采用同样的参数配置评估为TDSNN算法设计的五种运算单元方案，并将结果呈现在表5.2中。

首先是离散脉冲方案，该方案下的运算单元相对于 cambricon-pe-v1 功耗降低了53.44%，面积降低了71.89%，功耗和面积降低幅度都非常大。但是由前述的分析可知，该种方案下运算单元的运算效率降低为正常运算单元的16分之1。因此综合总能耗来说本方案存在明显劣势，大幅牺牲加速比换得的低功耗也是无意义的。但是考虑到离散脉冲的方案最符合按时间发放脉冲的模型，因此如果能够解决脉冲运算效率低下的问题，该方案不失为一种潜在地低硬件开销的方案，在未来的研究工作中值得进一步研究。

表 5.2 TDSNN专用硬件方案的硬件参数表

**Table 5.2 Hardware characteristics of designs for TDSNN**

| 设计方案 | discrete-spike-pe | continous-spike-pe | lut-pe-v1 | lut-pe-v2 | lut-pe-v3 |
|---|---|---|---|---|---|
| 功耗(mW) | 0.9012 | 1.2880 | 1.4475 | 0.8337 | 0.6763 |
| 相对于cambircon-pe-v1 | -45.62% | -22.28% | -12.65% | -49.69% | -59.19% |
| 相对于cambircon-pe-v2 | -53.44% | -33.46% | -25.22% | -56.93% | -65.06% |
| 面积($\mu m^2$) | 842.0457 | 7234.0661 | 2905.8646 | 2559.0185 | 1886.3181 |
| 相对于cambircon-pe-v1 | -65.47% | +196.69% | +19.18% | +4.95% | +22.64% |
| 相对于cambircon-pe-v2 | -71.89% | +141.52% | -2.98% | -14.56% | -37.02% |

对于连续脉冲的设计方案，该方案解决了离散脉冲重复发放脉冲效率低下的问题。由实验结果可知，相对于cambricon-pe-v1，该方案的功耗降低了22.38%，但是面积却增加了196.69%。分析发现原因在于额外的选择逻辑和乘法器的位宽增加导致整体硬件的面积显著增加。在保证基本运算效率不变的前提下该方案在整体功耗的降低上还是明显的，所以对于面积要求低的场景，在上述面积损失可接受的前提下不失为一种适宜的TDSNN算法部署方案。

查表方案lut-pe-v1相对于cambricon-pe-v2在功耗和面积上分别降低了25.22%和2.98%，可见该方案相对于参考基准的TDSNN算法映射方案存在明显的功耗优势，但是在面积开销上接近。虽然在整体功耗降低幅度上低于continous-spike-pe方案，但是其相对来说更具有面积上的优势。

最后是思想类似的两种查表方案lut-pe-v2和lut-pe-v3，相对于cambricon-pe-v1这两种方案上的功耗降低显著，分别达到了49.69%和59.19%，在功耗和面积的评估中我们已经利用参考基准中的表项计算单元的面积和功耗对这两个单元的数据进行了修正。这是所有方案中功耗降低幅度最大的两种方案。在面积上lut-pe-v2的面积相对只增加了4.95%，可以认为面积非常接近。但是lut-pe-v3的面积增加幅度明显，达到了22.64%。虽然这两个方案的功耗很低，但是考虑到两个方案在神经元复用的计算上的计算效率非常低，分别为参考基准的$4/NI$和$1/4$，因此对于神经元复用运算场景较多的场景势必会影响加速比。对于卷积层还可以尽可能地拆分出权值复用的计算方案，但是仍然对卷积层的规模存在高要求，因为每个PE和PE内部的复用计算都在进行权值的复用，必然会导致在小规模卷积的计算上浪费计算资源。对于全连接层这种神经元复用计算

大量出现的场景，运算效率将会明显降低。

### 5.4.3 特殊参数场景评估

**表 5.3 底数为2的优化后的硬件参数表**

**Table 5.3 Hardware characteristics of designs after 2-base optimization**

| 设计方案 | power(mW) | area($\mu m^2$) | 功耗相对于优化前 | 面积相对于优化前 |
|---|---|---|---|---|
| cambricon-pe-b2 | 1.3419 | 1289.7641 | -19.03% | -47.10% |
| discrete-spike-pe-b2 | 0.8065 | 804.7260 | -10.51% | -4.43% |
| continous-spike-pe-b2 | 0.99 | 4838.9580 | -23.14% | -33.11% |
| lut-pe-v1-b2 | 1.2733 | 1167.2050 | -12.03% | -59.83% |

由前文分析可知，在限定了 TDSNN 算法中的泄漏常数的取值后，可以实现了其中的大部分乘法运算使用移位运算来代替，在硬件设计上移位运算相比于乘法器的面积和功耗节省是明显的。表格5.3中展示了使用底数为2的幂次优化后的硬件设计的参数。可以发现，所有的方案的硬件面积和功耗都得到了优化。面积和功耗整体优化比例最低的是discrete-spike-pe方案，原因是其原先设计的乘法器数目只有一个，导致乘法器使用移位运算代替以及移除查表操作的优化效果相对较低。另外对于连续脉冲方案而言，虽然优化后的方案上功耗仍然比lut-pe-v2和lut-pe-v3高，但是其功耗优化幅度最大，相对于原始方案降低了23.14%，面积也降低了33.11%，但是其面积仍然是所有方案中最高的。虽然对于查表方案lut-pe-v1做了底数为2的优化使得其面积功耗都获得了降低，但是这些参数上的表现仍然高于原有的lut-pe-v2和lut-pe-v3。因此在不考虑离散脉冲方案这种明显存在能耗劣势的方案外，底数为2的优化方案中最优的方案是连续脉冲方案。

### 5.4.4 性能评估

上述的实验对单个运算单元的面积功耗进行了初步评估，但是如果在大幅损失性能的情况下的低功耗设计也是不划算的，为了确定每个设计方案的总运算开销，对各个方案的运算时间开销的评估也是必要的。本研究对三个典型的网络LeNet、AlexNet和VGG-16上各个设计方案的处理时间进行实验对比，并将每个方案与参考基准cambricon-pe-v1的加速比数据结果呈现在图5.22中。

首先对于cambricon-pe-v2、continous-spike-pe、lut-pe-v1以及三者对应的特殊参数的优化方案，这些方案相对于原有的cambricon-pe-v1在性能上是一致的，既没有加速效果也没有性能损失。这是符合预期的，因为在设计时就保持了这些方案的运算单元算力是一致性。

对于离散脉冲方案和其底数为2的优化方案，这两个方案都无法避免其重复脉冲带来的运算代价，理论分析上运算效率降低为原先的十六分之一，但是实际测试在三个网络上的平均时间开销为参考基准的15.8倍，基本符合预期。

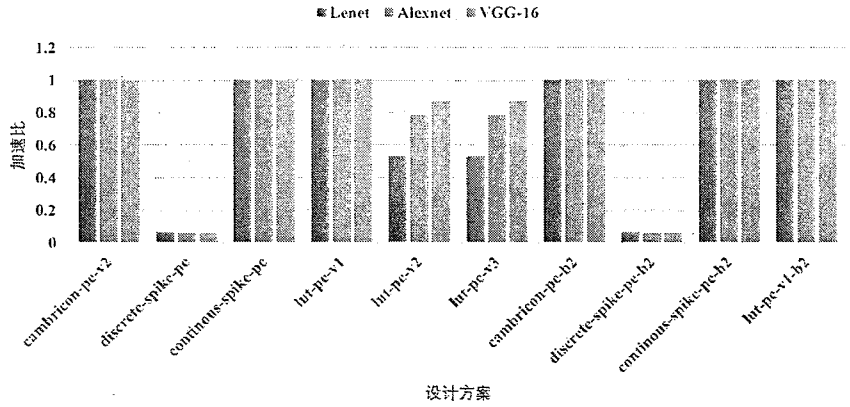


图 5.22 各个设计方案的整体加速比

**Figure 5.22 The overall acceleration ratio of each design**

对于查表方案lut-pe-v2和lut-pe-v3，运算效率也存在微弱的损失，这是由于其数据访问模式不同于其余的方案。在权值复用时广播数据通路需要广播权值表项，即意味着向量存储部件用于存储权值。并且每层的运算结果神经元数据不能直接存储于向量存储单元以供下次使用，因而会导致神经元数据到矩阵存储单元的额外搬运，在一定程度上导致运算效率的降低和开销的上升。但是观察发现随着网络规模的增大，性能的损失逐渐递减。在最小的网络LeNet中性能损失了46.88%，随着网络规模从LeNet到AlexNet再到VGG-16的逐渐增大，性能损失逐渐降低，分别是21.54%和13.08%，此处分析原因有二。一是由于随着网络规模的增大，尤其是在深层网络中，卷积层这种权值复用效益大的层的运算占比增加，相比而言全连接层的运算效率损失也开始逐渐降低。二是在小规模的网络例如LeNet中，一个feature map往往不够分给不同PE和不同PE内部完整的复用次数，造成运算资源浪费。所以对于规模越大、卷积层占比越多的深层网络而言，这两种方案的性能损失越低。

### 5.4.5 能耗评估

最后本研究通过对上述所有设计方案在三个典型网络上的能耗进行实验，来评估来说明最终的硬件能耗开销的差异，为了便于对比数据的差异，我们以参考基准cambricon-pe-v1的能耗与其余各个方案的比值为衡量标准并将结果呈现在图5.23中，也就是说柱状图数值越大的方案能耗越低。

直观上看，离散脉冲方案和其底数为2的优化方案在能耗上的表现差距与其余方案很大，结合上一小节对性能的评估，本研究认为该方案在性能和能耗上都并不具备优势，因此该方案目前并不适合部署TDSNN算法。

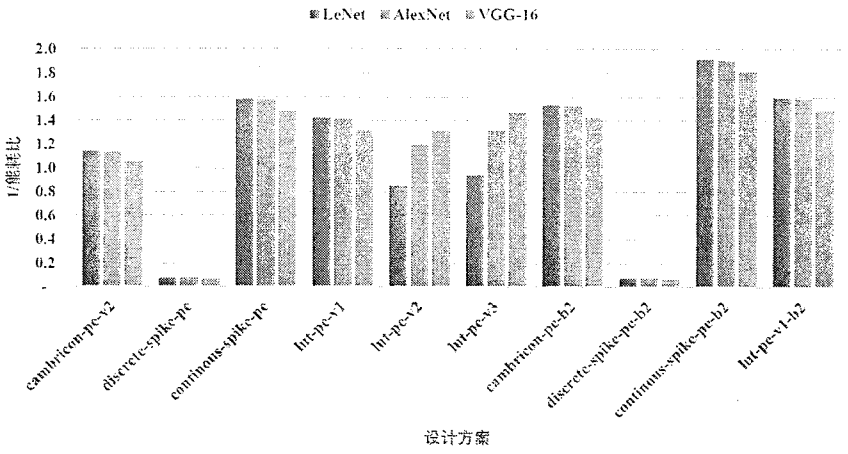


**图 5.23 各个设计方案的能耗比**

**Figure 5.23 Energy consumption ratio of each design**

对于另外一个最简易的实现TDSNN算法的方案cambricon-pe-v2，虽然其平均功耗上高于参考基准cambricon-pe-v1，但是其在三个网络上的能耗开销较低，分别降低了12.66%、12.04%和5.03%，而其底数为2的优化方案则带来了更高的能耗节省，分别达到了34.71%、34.32和29.97%。为何其功耗开销和性能相对于cambricon-pe-v1没有优势的情况下，仍然能够获得更低的能耗开销呢？其原因在于TDSNN算法的访存能耗优势上，算法使得神经元的存储和访存功耗降低为参考基准的四分之一，从而使得该方案在没有做任何运算优化的前提下也实现了如此程度的能耗节省。

能耗优化幅度最大的是底数为2优化后的连续脉冲方案，分别达到了47.79%、47.53%和44.80%。其优化前的方案也是所有优化前的方案中能耗降低比例最高

的方案。结合其功耗数据分析可知，虽然其功耗表现没有查表方案lut-pe-v2和lut-pe-v3那么明显，但是其在性能上相对于这两个方案并没有损失，且由于数据搬运模式不同，该方案相对而言没有明显的数据搬运导致的额外开销，因此整体上能耗达到了最优的效果。

而对于查表方案lut-pe-v1，其整体性能和连续脉冲方案接近，功耗略高于连续脉冲方案，访存和存储行为与连续脉冲方案行为一致。因此其整体效果略低于连续脉冲方案，但即使如此其相对于参考基准仍然实现了至少24.3%的能耗降低。其底数为2的优化方案也有类似的效果，位居所有能耗中排名第二的效果，实现了至少32.88%的能耗节省。

另外两种方案lut-pe-v2和lut-pe-v3，这两种方案的性能相对于其余方案有明显的损失，但是能耗数据显示这两种方案仍然能够实现一定程度的能耗降低。在小规模网络LeNet上，两者的能耗不降反升，相对于参考基准分别升高了17.42% 和 6.09%。结合性能数据表明这两种方案完全不适合于小规模的网络。对于后两种大规模网络上的表现，两个方案分别至少实现了16.59%和24.13%的能耗节省，且同样出现了网络规模增加节省比例增加的现象。说明后两种方案在大规模网络上存在开销节省的潜力，鉴于目前网络的规模问题无法对其做更大规模的网络的评估，未来出现网络规模特别巨大的网络时这两种方案不失为一种考虑方案。

## 5.5　小结

本研究通过对 TDSNN 算法硬件运算单元设计方案的充分研究，在理论上实现了六种通用的硬件实现方案和四种特殊参数场景下的优化方案，并针对每种硬件方案进行了复用方案的设计，以高效地部署神经网络计算。最终本研究在实验上确定了各个场景下每个方案的优劣性，为 TDSNN 算法快速部署到已有的加速器结构上做了充分的准备工作，与此同时也通过硬件运算单元的硬件实验表明，在底数为2的优化部署前后分别能够实现至少32.26%和44.80%的能耗节省。这进一步佐证了 TDSNN 算法确实具有降低运算能耗开销的效果，本研究成功提出了一款低运算开销的加速器设计方案。

# 第6章 总结与展望

本章首先对全文进行了总结，介绍了本文的主要内容和贡献，然后对本文研究引申出的一些问题进行了讨论，最后探讨了今后的工作展望。

## 6.1 本文总结

本文从大规模脉冲神经网络转化算法入手研究了高精度、低开销的大规模脉冲神经网络转化算法，在算法的基础上提取出低开销设计的运算特性，并将其运用到神经网络加速器的设计中，具体主要取得了以下创新性成果。

### 6.1.1 提出基于频率编码的大规模脉冲神经网络转化算法

针对大规模脉冲神经网络精度低的问题，本文首先从基于频率编码的转化算法角度进行了研究，提出了适用于大规模脉冲神经网络的转化算法DSNN。为了使得脉冲神经网络在硬件部署时具备易于拆分的特性，我们提出了DSNN-fold算法，该算法在精度、参数鲁棒性上的效果都优于DSNN算法。实验证明在主流的大规模卷积神经网络上，同等结构的脉冲神经网络在精度上可以实现与人工神经网络相同的精度水平。本研究将大规模脉冲神经网络的精度提高到了与CNN相同的水平。

### 6.1.2 基于时间编码的运算开销优化

本文在DSNN系列算法基础上做进一步的分析，发现基于频率编码的脉冲神经网络存在运算开销远高于同等结构的卷积神经网络的缺点。针对该问题本文研究基于时间编码的转化算法，提出了新的逆向编码的时间编码方式和适用于该编码方式的前向传播机制，并理论证明了基于此编码机制下的转化算法能够消除转化过程带来的精度损失。本文在新的编码机制下得出了基于时间编码的转化算法TDSNN，实现了卷积神经网络到时间编码脉冲神经网络的转化。实验结果表明转化过程不仅无明显精度损失，还取得了至少降低40%运算开销和75%访存开销的效果。本研究是学术界第一次成功实现基于时间编码的大规模脉冲神经网络，对后来的时间编码与转化算法研究具有启发性意义。

### 6.1.3 设计低开销的神经网络加速器

在TDSNN算法能够削减运算开销的理论基础上，本文着手将这一运算优势运用到神经网络处理器的设计中。通过对算法进行深入分析，本文首先提取得到了TDSNN算法下的神经网络核心运算层的运算模式。然后本文选取了Cambricon架构为参考基准，将设计重心放在了运算单元和算法部署上。本文从离散脉冲方案、连续脉冲方案、查表方案三个角度出发探索研究了五种不同的运算单元设计方案与其配套的算法部署方案。为了最大程度发挥TDSNN算法的运算开销削减的优势，本文还探究了参数在极端情况下带来的运算单元设计方案的变化，并相应提出了四种运算单元设计方案。通过对这九种方案与参考基准设计方案的实验进行分析，本文得到了不同方案的实际硬件面积、功耗、能耗、加速比的效果对比结果。其中相对于参考基准，在维持加速比不变的前提下，在极端参数优化前后本文的设计方案分别能够实现至少32.26%和44.80%的能耗节省。

本文的研究从脉冲神经网络转化算法的角度，探索出了一条提高脉冲神经网络精度、降低网络运算开销、设计低开销神经网络加速器的路线。

## 6.2 相关工作

本小节对本研究涉及到的相关工作进行了补充讨论。

### 6.2.1 SNN研究里程碑

为进一步说明本文研究的意义以及后续本领域相关研究的发展情况，本小节建立了SNN学术研究的里程碑，如图6.1所示。

可见，自2013年转化算法第一次应用于SNN开始，一直到2017年，SNN的前期学术研究一直局限于小规模的网络。本文的研究自2017年开始，提出了基于频率编码的大规模SNN转化算法DSNN系列算法，首次将SNN的转化算法成功应用于大规模脉冲神经网络，缩小与CNN精度差距到2%以内。

2018年开始有工作尝试利用时间编码构建转化算法，但应用范围仅限于MNIST这种简单网络。本文的研究第一次成功建立了基于时间编码的大规模SNN 转化算法TDSNN，实现了与CNN的精度差距1%以内的效果，并同时获得了运算开销上的巨大收益。

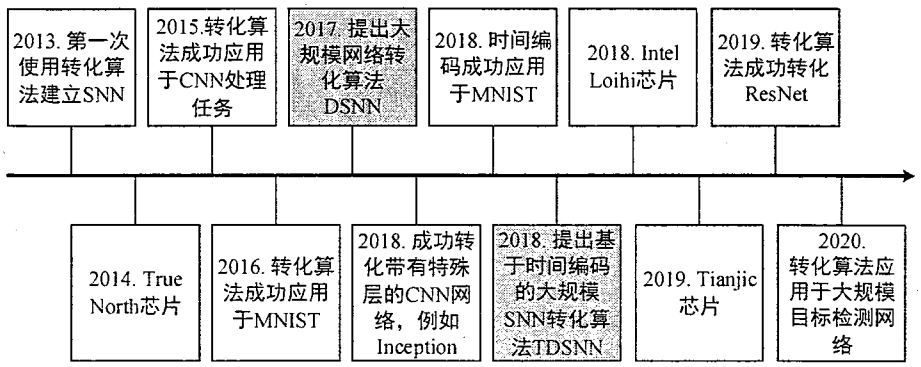


**图 6.1 SNN学术研究里程碑**

**Figure 6.1 Research Milestone of SNN**

此后，SNN转化算法的研究又陆续开展了对复杂结构构成的网络的研究，例如Inception系列网络和ResNet网络等。近期的研究也开始出现将SNN的转化算法开始应用于最新的目标检测网络等非识别任务上的研究。可见本文的工作为后续SNN转化算法的研究做出了阶段性的贡献。

### 6.2.2　SNN硬件

本文的研究中利用TDSNN算法设计了新的加速器运算单元，并成功应用于Cambricon 架构，那么本文的加速器与目前的主流SNN加速器之间的效果对比情况如何呢？本小节进行这方面的补充说明。

由于主流的SNN硬件并不一定能够支持运行本文选取的大规模网络，因此评估加速器效果的参数选择无法按照网络的基准来进行。此处选取了两个加速器关键参数：吞吐量（Throughput）和单位操作的能耗，来说明硬件的实际效果。

图6.2展示了Cambricon架构、本文工作以及三个主流的SNN硬件TrueNorth、Loihi和Tianjic的硬件吞吐量对比情况。由于性能保持与Cambricon架构保持一致，本文的工作在吞吐量上也保持一致。相比于另外三个主流的SNN硬件，以本文工作为代表的ANN硬件在吞吐量上远高于这些硬件。这说明了在性能上，SNN硬件仍然不适合处理大规模的网络任务，而ANN的高吞吐量更适合处理数据流庞大的复杂网络。

再来分析能耗情况，此处选择单位操作的能耗来作为评估标准。这里主要考虑到主流的SNN硬件并不适合处理大规模的网络任务，直接进行网络总能耗

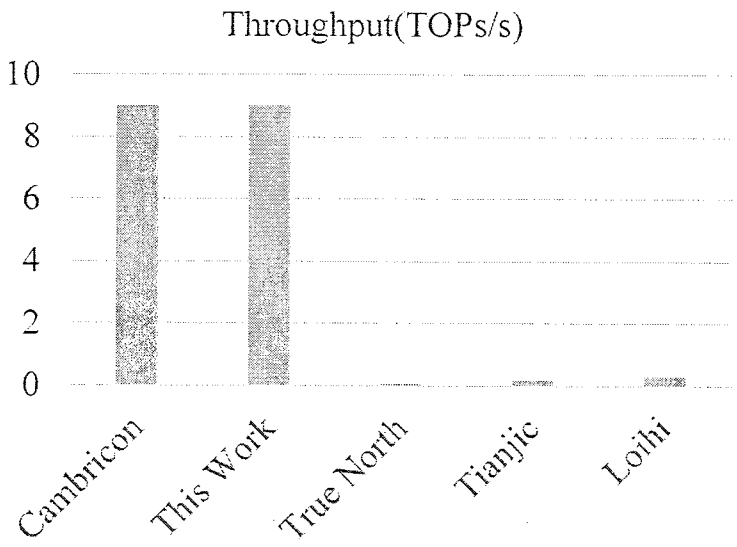


图 6.2 硬件吞吐量对比

**Figure 6.2 Hardware throughput comparison**

的评估有失公平性。另外部分硬件如Tianjic和Loihi目前并未证实能够用于图像分类任务，其大部分应用场景局限于目标检测等任务中，因此无法进行常规数据集的评估。因此此处选择单个操作的能耗能够一定程度上代表相同操作量级下的能耗开销情况。如图6.3所示，可见在能耗上，SNN硬件仍然相比于本文的工作存在明显的差距。

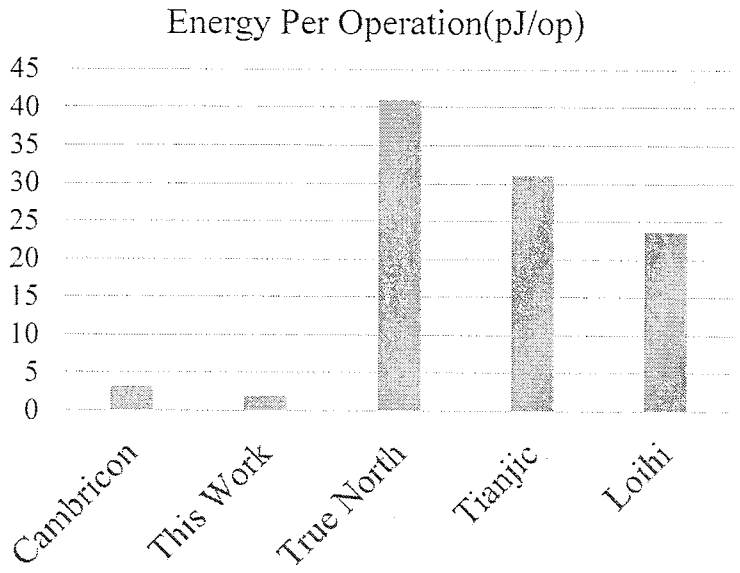


图 6.3 硬件能耗对比

**Figure 6.3 Hardware energy comparison**

这部分的结论也基本符合学术界之前关于SNN硬件与ANN硬件的讨论，考虑到SNN目前的学术研究的进程，以及硬件研究的方向，短期内SNN硬件应用于处理传统的大规模图像处理任务的可能性相对较低。另外，性能和能耗上的结果也说明了，SNN领域研究距离投入实际使用还有较大的差距。

## 6.3　讨论

本节对全文的研究中涉及到的学术性问题进行了整理与讨论。

### 6.3.1　转化算法

在本文的研究中，实现高精度的大规模脉冲神经网络必须借助于转化算法，那么转化算法是否是唯一的实现方式呢？其实并不是，学术界一直试图为大规模脉冲神经网络开发自己的训练方法解决训练难题，但是目前仍未有明显成效，转化算法只是解决SNN训练问题的短暂解决办法。本人认为研究SNN训练算法至关重要，否则SNN的精度将持续受限于已有的ANN网络，难以针对特殊结构进行重新训练。但是本人认为，转化算法的研究不止是解决SNN训练难这一个意义，它还有助于开发更高效的神经网络算法和硬件。因为转化算法的研究解决了两种网络的转化问题，未来如果产生更高效的SNN算法，将有可能借助转化算法实现更高效的ANN网络。

### 6.3.2　时间编码与频率编码

在频率编码已经解决SNN精度问题的基础上，本文又开发了时间编码的SNN 以实现更低开销的SNN，那么未来的研究是否应该一直坚持时间编码的方式？本人认为SNN的研究不能局限于时间编码。考虑到真实的生物神经网络中，频率编码、时间编码以及二者相结合的编码都广泛存在，针对单一的编码方式的研究不利于发现和利用更深处的生物学规律。另外，目前主流的SNN硬件和训练算法都依赖基于频率编码的网络，针对频率编码的SNN进行更深入的研究将有利于推广SNN到更广泛的应用中。

### 6.3.3　SNN与ANN

经过如此多学着的努力，SNN才终于实现与CNN相同的精度水平，是否意味着SNN的研究价值不大？本人始终认为脉冲神经网络的研究与人工神经网络的研究密不可分，两者的研究能够相互促进、相互启发。例如SNN中的IF神经

元模型启发了ANN的研究人员设计了ReLU激活函数，更高效地完成训练任务。本研究中时间编码的SNN也启发得到了神经网络有限值量化方法，并基于此对神经网络加速器做出了进一步优化。因此，未来两个领域的研究都非常重要，具备强生物特性的SNN有希望为ANN领域的研究注入新的动力，也可能启发设计更高效的神经网络加速器。

### 6.3.4 神经网络硬件

针对人工神经网络硬件发展成熟，而脉冲神经网络硬件应用短缺的现实情况，未来的神经网络硬件是否有必要支持SNN应用？是否有必要开发专门的SNN硬件？本人认为SNN硬件的研究有巨大的价值。短期内，针对一些低功耗的应用场景，结合特殊的事件采集传感器设备，SNN硬件能够发挥出事件触发的优势，实现低功耗地处理智能任务。长期发展来看，SNN硬件的研究结合了很多新兴的学科技术例如新材料技术、新电气设备等，有利于促进多学科的发展与技术融合，最终服务于对人工智能的进一步探索。

## 6.4 未来工作

脉冲神经网络的研究成果不仅会增进科学界对生物神经网络的理解，也会推动神经网络处理器设计领域的发展进步。但是到目前为止，脉冲神经网络算法真正应用到实际的工业级项目中的情况仍然非常罕见，仍然需要长期深入的研究。未来的工作将会沿着如下方向开展：

1.**研究表达能力更强的脉冲神经网络**。本文的研究已经表明同等结构的脉冲神经网络与同等结构的人工神经网络表达能力接近，但这是在对网络模型和参数等做出了简化和限制的基础上得出的结论。未来通过对网络结构、神经元模型、参数策略上更进一步的研究，在表达能力上同等结构的脉冲神经网络具备超过人工神经网络的潜力。此项研究将有利于开发在结构上更精简的神经网络模型，并有利于实现低开销的神经网络处理器。

2.**研究脉冲神经网络的优势领域**。目前脉冲神经网络能够在图像识别等领域取得和人工神经网络相当的效果，但是仍然无法做到替代人工神经网络的程度。未来的研究可以考虑探索目前亟待解决问题的几个领域例如逻辑推理、生成语言等，探索其中适合用脉冲神经网络处理的领域。脉冲神经网络独特的前向传播机制和神经元模型可能有利于对一些棘手的问题进行建模。

3.**研究神经网络处理器新架构**。已有的神经网络架构例如DianNao系列的发展已经走向成熟，为实现更强性能、更低开销的神经网络处理器，可以在上述两个方面的脉冲神经网络模型基础上，研究更低运算开销的网络实现方式，从而开发新的处理器架构，进一步推动神经网络处理器的发展。

未来的研究道路充满了机遇与挑战，本文的工作在解决现有问题的同时，也未来的探索道路做出了基础性的铺垫。