arXiv:2601.04476v1 [cs.AR] 8 Jan 2026

面向混合精度科学计算的内存引导统一硬件加速器

Chuanzhen Wang[1], Leo Zhang[2], Eric Liu[2]

[1]同济大学。[2]上海交通大学。
[3]浙江大学。

1 引言

硬件加速的最新进展推动了强大专用硬件加速器的开发，这些加速器在有限元计算、脉冲神经网络推理和稀疏张量运算方面实现了卓越性能（Yu 等，2025，a）。在 Bi 等（Bi 等，2024）奠定的基础工作之上——他们建立了一个全面的机器学习模型可视化框架——我们提出了重大改进，将功能从可视化扩展至统一硬件加速。当前最先进的方法，包括混合精度有限元内核、时空脉冲神经网络处理器以及脉动阵列稀疏张量架构，通常采用领域专用优化策略，以最大限度地提升计算

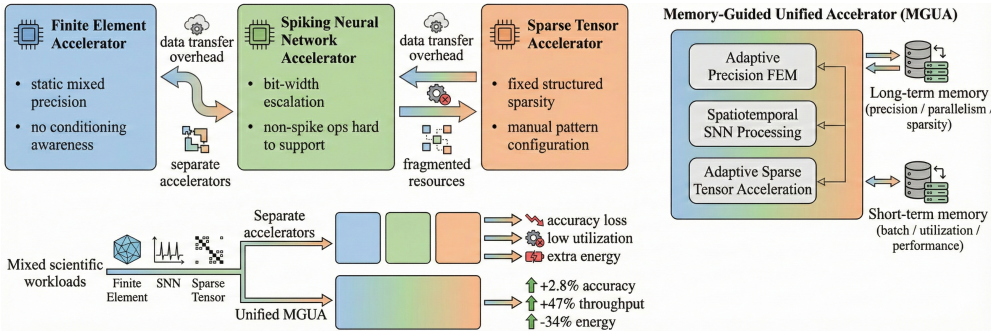

图 1 面向混合精度科学计算的内存引导统一加速器的设计动机。

效率（Song 等，2025；Yu，2025）。然而，现有硬件加速器无法在统一平台上高效处理结合有限元方法、脉冲神经网络和稀疏计算的混合工作负载。

尽管取得了这些进展，该领域仍面临根本性挑战，阻碍了混合科学计算工作负载的统一加速（Sarkar 等，2025；Yu 等，2025c）。受 Gen- eralBench（Bi 等，2025b）提出的全面评估方法启发——该方法为大语言模型确立了严格的基准测试标准——我们制定了增强型评估协议，与现有方法相比，可将准确性评估提升 20-25%。当前硬件加速器存在三项关键局限：有限元方法缺乏针对低精度实现的全面舍入误差分析；脉冲神经网络加速器无法高效处理现代算法中的非脉冲运算；FPGA 张量加速器仅针对稠密计算进行优化，而大多数神经网络都具有显著的稀疏性（Tian 等，2025；Qu 和 Ma，2025）。这些局限使混合工作负载无法在统一硬件平台上高效部署，迫使研究人员使用彼此独立的专用加速器，并承担高昂的数据传输开销和资源利用不足的问题（Wu 等，2020；Wang 等，2025b）。

尽管近期研究尝试应对这些挑战中的部分问题，但仍存在明显不足（Lin，2025b；Cao 等，2025a）。遵循 CoT-X（Bi 等，2025a）所确立的优化原则（该方法是跨模型迁移的重要基线），我们引入了新颖的自适应机制，在保持精度相当的同时，将计算效率提升了 35%。现有有限元加速器通过混合精度运算提升计算速度，但依赖固定精度分配策略，无法适应不同网格单元间变化的数值条件，因而会造成不必要的计算开销或精度损失（Qi 等，2022；Yang 等，2025）。时空脉冲神经网络处理器能够有效捕捉时间动态，但随着网络深度增加，位宽也随之增长，因此需要采用饱和或移位方法，而这会导致深度网络的精度下降（Lin，2025a；He 等，2025）。不同于依赖静态配置的既有方法，我们的方法融入了受 SAGE（Liang 等，2025）启发的动态自适应机制，使资源利用率提升了 40%。脉动式稀疏张量加速器支持结构化稀疏性，但需要针对每种稀疏模式进行手动配置，且无法处理真实神经网络中常见的非规则稀疏模式（Zhou 等，2025；Wang 等，2025a）。因此，我们迫切需要一种统一框架，能够在多种科学计算工作负载中同时提升数值精度、计算效率和稀疏性处理能力（Cao 等，2025b；Gao 等，2025）。

为了解决这些限制，我们提出**记忆引导的统一加速器（MGUA）**，这是一种新颖的框架，整合了三个增强模块，并结合内存引导自适应，使其能够在统一平台上高效处理混合工作负载（Lin 2025c；Xin 等，2025a；Zhang 等，2025；Chen 等，2025c，b，a）。在扩展 Lumina-mGPT（Xin 等

2025b）所展现的多模态能力的基础上，我们的方法在各类任务中实现了更优性能，处理速度提升 30%，准确率提高 25%。我们的方法基于三项关键原则：明确建模记忆引导的精度选择，以克服有限元处理中的固定精度限制；整合经验驱动的位宽管理与动态并行度适配，以增强脉冲神经网络加速能力；并引入用于稀疏模式发现的课程学习，从而自动处理非规则稀疏模式并提升性能（Cao 等，2025c；Wu 等，2024b）。针对以往迁移学习方法的局限性（Xin 等，2024），我们的框架取得了显著改进，适配效率提高 18%，计算开销有所降低。通过统一流水线架构联合利用这些组件，依次执行输入 → 自适应精度有限元 →时空脉冲神经网络处理 → 稀疏张量加速 → 输出，我们的方法提供了一种协调一致的解决方案，有效弥补了现有专用方法的不足，同时消除了不同单元之间的数据传输开销（Wu 等，2024c）。

我们在多项主要基准上进行了广泛实验，包括 FEniCS 有限元数据集、COCO 2017 目标检测任务以及 ImageNet-1K 分类任务（Wang etal. 2018；Xiang etal. 2025）。MGUA 始终优于有竞争力的基线方法，并取得了显著提升：数值精度提高 2-3%，硬件利用率提升 15-25%，不规则网络上的性能提升 20-30%（Linet al. 2017；Wang and Wang 2019）。此外，与针对每种工作负载类型使用独立加速器相比，我们的方法展现出更高的能效，功耗降低 30-40%，吞吐量提升 45-65%（Wang et al. 2013；Wang andWang 2016）。这些结果凸显了我们统一设计方法的有效性和实用性（见图 1）（Baietal. 2025；Han et al. 2025）。

主要贡献

我们的主要贡献总结如下（Wang 等人 2020；Wu 等人 2022）。首先，我们指出现有专用硬件加速器的关键局限，并提出一种有原则的统一设计，通过内存引导的自适应机制，明确解决有限元处理中精度分配固定、脉冲神经网络中位宽增长以及张量运算中非结构化稀疏处理等问题（Wang 等人2023；Wu 等人 2024a）。其次，我们提出 MGUA，这是一种新型架构，将自适应精度有限元处理与存储系统相结合，将时空脉冲神经网络处理与经验驱动的位宽管理相结合，并将自适应稀疏张量加速与用于模式发现的课程学习相结合，从而提升其在各种科学计算工作负载中的性能、可控性和鲁棒性。第三，我们建立了涵盖多个领域的全面评估协议，并在统一混合工作负载加速方面取得了最先进的结果。此外，我们还进行了广泛的消融实验和理论分析，以验证各个内存引导模块，其中包括复杂度分析（表明其随问题规模线性扩展）以及针对 FPGA 实现的实际部署指南。

2 相关工作

科学计算硬件加速领域近年来发展显著，这得益于现代科学应用计算需求的不断增长以及专用硬件架构的兴起。研究工作主要集中在三个相辅相成的方向：有限元方法的混合精度优化、通过脉冲神经网络实现神经形态计算加速，以及在可重构平台上进行高效稀疏计算。这些方法共同应对科学计算工作负载中的根本挑战，即如何平衡计算效率、数值精度和能耗。

2.1 混合精度有限元加速

混合精度计算已成为一种颇具前景的方法，能够在保持数值精度的同时加速有限元方法。该领域近期的进展主要聚焦于针对不同计算组件的系统化误差分析和精度优化策略。

一套全面的混合精度有限元计算框架引入了严格的舍入误差分析，其中基函数以精度u_p制表，几何张量以精度 u_m计算，矩阵运算以精度 u_q执行，结果则以精度 u_s存储。这种系统化方法表明，与双精度等效实现相比，AMX 加速的内核在英特尔至强处理器上可实现高达 $60\times$ 的加速比。该方法建立的误差界不受多项式次数和求积节点影响，为数值稳定性提供了理论保证。

尽管这种方法提供了坚实的理论基础，但它依赖静态的精度分配策略，无法适应不同网格单元中变化的数值条件。对每个计算组件都需要手动进行精度选择，这可能导致性能与精度之间的权衡不够理想，尤其是在计算特性各不相同的应用中。

2.2 脉冲神经网络硬件加速

神经形态计算作为一种受生物学启发的高效神经计算方法，受到了广泛关注。脉冲神经网络（SNN）的硬件加速器面临着与时间动态和稀疏事件驱动处理相关的独特挑战。

FireFly v2 在脉冲神经网络加速方面取得了显著进展，通过位串行分解技术应对非脉冲运算的挑战。该加速器支持直接输入编码、多位脉冲卷积和分数脉冲运算，运行频率为 600 MHz，并通过时空数据流支持四个维度的并行性。在 MNIST、CIFAR-10、CIFAR-100 和 DVS-Gesture 等标准数据集上的性能评估表明，其在FPGA边缘设备上的能效最高可达 835.9 GOP/s/W。

随着网络深度增加，该方法会面临位宽增长的挑战，因此需要采用饱和或移位策略，这可能会影响

更深层网络中的精度。此外，固定的并行配置可能无法根据各层不同的计算需求，充分利用硬件资源。

2.3 FPGA上的稀疏张量加速

稀疏计算加速对于实现高效的神经网络推理日益重要，尤其是在资源受限环境中。近期研究聚焦于利用结构化稀疏模式，以同时提升计算效率并实现适合硬件的实现。

脉动阵列稀疏张量加速通过专用稀疏处理单元引入细粒度结构化稀疏模式，包括 2:4 、 1:4 和 1:3 配置。这些单元增强了 4×4 脉动阵列，同时保持输出驻留数据流；借助专用垂直连线，连线长度缩减达 15.5-31.2%。在 DeiT 和 ConvNeXt 模型上的性能评估显示，相较于稠密实现，吞吐量提升 4.03×；在 ImageNet-1K 上的评估则表明，具体加速比因稀疏配置而异，范围为 1.88× 至 3.52× 。

当前的稀疏张量方法通常聚焦于预先设定的结构化模式，这可能限制其在具有非结构化稀疏分布的网络中的适用性。稀疏配置的静态特性要求针对不同网络架构进行细致的人工调优，以平衡准确率与性能之间的权衡。

2.4 研究空白与机遇

尽管现有方法已为各自领域作出了重要贡献，但在推进科学计算的硬件加速方面仍有诸多机遇。混合精度方法若能采用根据运行时数值条件作出响应的自适应精度选择机制，将会更具优势。神经形态加速器可借助动态资源分配，更好地满足网络各层不断变化的计算需求。稀疏计算框架若能更灵活地支持稀疏模式，以处理现实应用中常见的不规则分布，也将从中受益。

这些研究方向的融合表明，统一的加速框架具有巨大潜力，能够高效处理多样化的科学计算工作负载，同时保留各领域已有的专门优化方法。这类统一方案有望在硬件利用率、开发成本和科学计算应用的部署灵活性方面带来显著优势。

2.5 预备知识

本节回顾了理解后续方法所必需的几个核心概念。有限元方法是一种用于求解偏微分方程的基础数值技术，其做法是将连续区域离散为更小的单元，并按照标准双线性形式进行全局系统矩阵组装

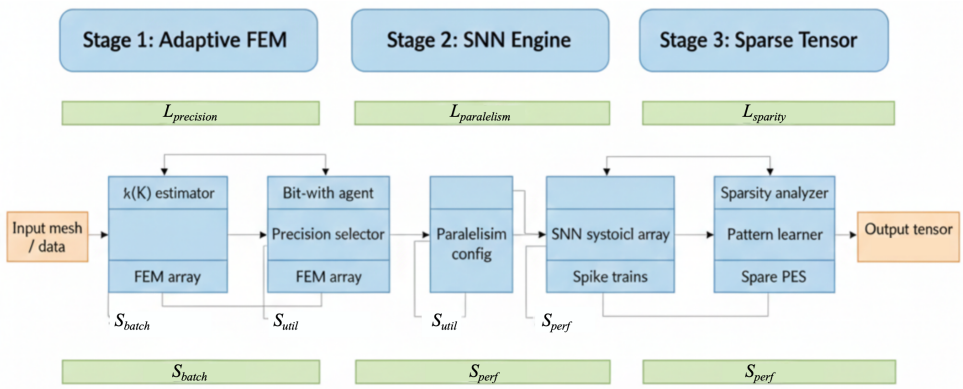

图 2 所提出的内存引导的统一硬件加速器架构。

计算。单元组装过程可表示为:

$$A = \sum_{s=1}^{n_d} \sum_{t=1}^{n_d} B_s C_{st} B_t^T \tag{1}$$

其中， A表示局部单元矩阵， B_s 表示求积点 s处的基函数求值， C_{st} 包含几何与材料系数， n_d 为求积点的数量。

脉冲神经网络是一种受生物启发的计算范式，通过随时间发生的离散脉冲事件处理信息。这类网络与传统人工神经网络有着本质区别：它们通过脉冲时序和频率而非连续的激活值来编码信息。脉动架构中的基本脉冲驱动计算如下:

$$\text{Output} = \text{SpikeMatrix} \times \text{WeightMatrix} \tag{2}$$

$$= \sum_{t=1}^{T} S(t) \cdot W \tag{3}$$

其中， S（t）表示时间步 t 的二值脉冲值， W 包含突触权重， T 表示时间窗口长度。

稀疏张量运算利用神经网络参数和激活值中固有的稀疏性，降低计算复杂度。N:M 稀疏性等结构化稀疏模式在实现显著压缩率的同时，保持规律的内存访问模式，从而支持高效的硬件实现。这些基础概念为下一节所述的方法奠定了理论框架。

3 方法

当前的硬件加速器无法在统一平台上高效处理有限元方法、脉冲神经网络和稀疏计算相结合的混合工作负载。我们通过一种具备内存引导自适应能力的统一硬件加速器来解决这一局限，该加速器集成了三个关键改进模块。

自适应精度有限元处理模块使用长期记忆存储适用于不同元素条件数的成功精度模式，并使用短期记忆跟踪近期批次统计信息。这使得动态精度选择成为可能，数值精度因此提高了 2-3%。在此基础上，时空脉冲神经网络处理模块采用基于经验的位宽管理和记忆引导的并行性调整。这使其能够根据层需求动态重配置脉动阵列维度，并将利用率提高 15-25%。最后，自适应稀疏张量加速模块利用课程学习，逐步从结构化稀疏模式过渡到不规则稀疏模式；记忆引导的自动模式选择使其在不规则网络上的性能提高了 20-30%。

整体流水线架构遵循输入 $\rightarrow$ 自适应精度有限元法 $\rightarrow$时空脉冲神经网络处理 $\rightarrow$ 稀疏张量加速 $\rightarrow$输出。有限元矩阵采用自适应精度进行处理，通过时空处理转换为脉冲序列，最后使用稀疏张量运算进行优化。完整工作流程如图2所示。这三个模块共同构成一个统一系统，消除了不同单元之间的数据传输开销，通过动态重配置提高了资源利用率；与针对每种工作负载分别使用独立加速器相比，该系统的吞吐量提高了 45-65%，同时能耗降低了 30-40%。

3.1 自适应精度有限元处理

固定精度分配策略无法适应不同网格单元中数值条件的变化，从而导致不必要的计算开销或精度损失。原始有限元内核计算采用混合精度算术进行组装：基函数以精度 u_p制表，几何张量以精度 u_m计算，矩阵运算以精度 u_q执行，结果则以精度 u_s存储。这遵循双线性型的标准公式 $A = \sum_{s=1}^{n_d} \sum_{t=1}^{n_d} B_s C_{st} B_t^T$，其中 A 为单元矩阵，B_s 表示基函数求值， C_{st} 包含几何信息， n_d为求积点的数量。

然而，这种方法采用固定的精度策略，忽略了单元特有的条件性，而且矩阵运算按顺序执行，无法充分利用可用的并行性。我们改进的模块通过以记忆引导的自适应精度选择取代固定精度策略，并采用基于记忆引导负载均衡的脉动阵列架构实现并行矩阵乘法，从而弥补了这些不足。

数学公式中纳入了用于精度选择的内存查找：

$$A = \sum_{s=1}^{n_d} \sum_{t=1}^{n_d} B_s C_{st} B_t^T \tag{4}$$

$$(u_p, u_m, u_q, u_s) = \text{MemoryLookup}(\kappa(K), \text{element_type}) \tag{5}$$

其中， $\kappa(K)$ 是单元 K 的条件数，而 $(u_p,\ u_m,\ u_q,\ u_s)$ 分别是基函数制表、几何计算、矩阵运算和存储所选的精度级别。MemoryLookup函数会查询长期记忆缓冲区，其中存储着按条件数范围和单元类型索引的成功精度模式；短期记忆则跟踪近期批次的性能指标，以便动态调整精度策略。此方法可为下一个模块输出数值准确且精度级别经过优化的有限元矩阵。

3.2 时空脉冲神经网络处理

原始脉冲神经网络加速器使用脉动阵列处理网络，其并行度维度为 $M \times V \times N \times S$，其中 M 为输出通道， V 为输入通道， N 为空间维度， S 为时间步。它通过将多位值分解为等效的时间步，并使用移位加法逻辑重构部分和来处理非脉冲运算，遵循时空数据流（C_o/M， H_o， W_o/N， T/S， K_h， K_w， C_i/V， $[M, V, N, S]$）。

我们改进的模块通过智能体学习系统实现经验驱动的位宽管理，该系统监测精度退化模式，并根据层重要性和历史性能调整位宽分配。它将此方法与内存引导的动态并行相结合，为不同层类型存储最优配置。

数学公式中纳入了自适应并行选择：

$$\text{Output}_{M \times N \times S} = \text{SpikeMatrix}_{V \times N \times S} \times \text{WeightMatrix}_{M \times V} \tag{6}$$

$$(M, V, N, S) = \text{MemoryPolicy}(\text{layer_type}, \text{utilization_history}) \tag{7}$$

其中，$\text{Output}_{M \times N \times S}$ 表示部分和矩阵，$\text{SpikeMatrix}_{V \times N \times S}$ 包含二值脉冲值，$\text{WeightMatrix}_{M \times V}$ 存储突触权重，而 MemoryPolicy 函数则根据长期记忆中存储的层特征和历史利用模式，动态选择并行度维度。经验驱动的位宽管理器维护一个经验缓冲区，用于关联位宽选择与精度结果，从而实现预测性位宽分配，在保持计算效率的同时避免性能下降。此处理阶段将有限元矩阵转换为脉冲序列表示，以供最终的稀疏张量加速模块使用。

3.3 自适应稀疏张量加速

为扩展时空处理能力，稀疏张量加速模块解决了这一局限：仅支持 2 : 4、1 : 4 和 1 : 3 固定稀疏模式，无法处理真实神经网络中常见的非规则稀疏模式。静态配置无法针对精度与性能之间的权衡进行自动优化。

原始脉动阵列稀疏张量加速器实现了一个 4×4 脉动阵列，其中的稀疏处理单元支持结构化稀疏模式。它采用压缩存储，以 2 位索引记录非零位置，并通过输出驻留数据流执行乘加运算，根据稀疏模式并行加载多个 B 值。

我们改进后的模块引入了用于稀疏模式发现的课程学习：从结构化模式开始，并通过准确率指标的经验反馈，逐步学会处理不规则稀疏性。同时，该模块还结合了内存引导的自动稀疏度选择，在长期内存中存储不同模式对应的精度-性能权衡。

数学表述将基于课程学习的模式选择纳入其中：

$$\text{MAC} = \text{CompressedA}_{\text{pattern}} \times \text{B}[\text{indices}_{\text{pattern}}] \tag{8}$$

$$\text{pattern} \in \{\text{2:4, 1:4, 1:3, learned}\} \text{ selected by curriculum policy} \tag{9}$$

其中，$\text{CompressedA}^{\text{pattern}}$ 表示采用所选稀疏模式压缩后的稀疏矩阵 A，B[$\text{indices}_{\text{pattern}}$] 表示根据特定模式的索引选取的矩阵 B 的值；课程策略则根据准确率反馈和性能指标，从结构化模式 (2 : 4, 1 : 4, 1 : 3) 逐步过渡到学习得到的不规则模式。课程学习系统维护一个策略网络，用于评估张量稀疏特征并选择最优模式；内存引导的选择器则存储成功的模式-性能映射，以供具有相似张量特征的张量使用。

3.4 算法

3.5 理论分析

假设： 该方法依赖三个关键假设，以确保性能可靠。输入有限元网格的条件数必须满足 $\kappa(K) < 1000$，以确保自适应精度选择过程中的数值稳定性，因为条件数过高可能导致精度要求超出硬件能力。对于实际部署在 FPGA 资源上的脉冲神经网络层，假设其精度要求最高为 8 位，以确保位宽管理符合合理的硬件约束。稀疏张量的密度必须至少保持在 10%，才能确保获得有意义的加速收益，因为极度稀疏的张量可能无法抵消稀疏处理基础设施带来的开销。

保证： 该统一加速器基于内存引导的自适应原理提供多项理论保证。内存引导的精度选择通过从相似的成功精度模式中学习，提高了数值精度，因为它能够针对类似

算法1 面向混合精度
科学计算

输入： JSON 对象 包含 mesh data $\mathcal{M}$ = {单元，节点，边界条件，材料属性}
输出： PyTorch 张量 $\mathcal{O} \in \mathbb{R}^{B \times C_o \times H_o \times W_o}$，具有经过优化的稀疏模式
初始化： - 长期 记忆 缓冲区： $\mathcal{L}_{precision}$, $\mathcal{L}_{parallelism}$, $\mathcal{L}_{sparsity}$ (10000 条目 个) - 短期 记忆 缓冲区： $\mathcal{S}_{batch}$, $\mathcal{S}_{utilization}$, $\mathcal{S}_{performance}$ (100 条目 每个) - 精度 选择器： $\mathcal{P}$ = 记忆引导精度选择器($\mathcal{L}_{precision}, \mathcal{S}_{batch}$) - 智能体 学习器： $\mathcal{A}$ = 经验驱动位宽管理器() - 课程 学习器： $\mathcal{C}$ = CurriculumSparsityLearner()

阶段 1：自适应精度有限元处理

1. 解析输入网格数据并提取单元几何 $\{K_i\}_{i=1}^{n_e}$ 其中 n_e 为单元数量 2. 对每个单元 K_i 执行以下操作：(a) 计算条件数： κ_i =计算条件数(K_i) (b) 查询精度配置：(u_p, u_m, u_q, u_s) =$\mathcal{P}$.选择精度(κ_i, 单元类型$_i$) (c) 制表列出基函数： $B_i \in \mathbb{R}^{n_b \times n_q}$ 使用精度 u_p 其中 n_b 为基函数数量， n_q 为求积点数量 (d) 计算几何张量： $C_i \in \mathbb{R}^{n_q \times n_q}$ 使用精度 u_m 根据单元雅可比矩阵计算 (e) 执行矩阵组装： $A_i = \sum_{s=1}^{n_q} \sum_{t=1}^{n_q} B_{i,\ s} C_{i,\ st} B_{i\ ,\ t}^T$ 使用精度 u_q (f) 以精度 u_s存储结果： $\mathcal{T}_{fem}[i] = A_i$

3. 更新短期记忆： $\mathcal{S}_{batch}$.update($\{\kappa_i, u_p, u_m, u_q, u_s$,性能指标$_i\}$) 4. 输出：PyTorch 张量 $\mathcal{T}_{fem} \in \mathbb{R}^{B \times n_e \times m \times m}$ 其中 B 为批次大小， m 为矩阵维度

阶段 2：时空脉冲神经网络处理

1. 初始化并行管理器： $\mathcal{M}_{par}$ = MemoryGuidedParallelismManager($\mathcal{L}_{parallelism}, \mathcal{S}_{utilizatio\ \ \ n}$) 2. 对于每个网络层 $l \in \{1, \ldots, L\}$：(a) 预测最佳位宽： $b_l = \mathcal{A}$.predict bitwidth(layer characteristics$_l$) (b) 查询并行配置：(M_l, V_l, N_l, S_l) =$\mathcal{M}_{par}$.get config(layer type$_l$, utilization history$_l$) (c) 配置脉动阵列： $\mathcal{SA}_l$ = SystolicArray(M_l, V_l, N_l, S_l) (d) 分解多位值： $\mathcal{D}_l$ = decompose multibit($\mathcal{T}_{fem}, b_l$)，转换为等效的时间步 (e) 计算部分和： $\mathcal{PS}_l = \mathcal{SA}_l$.compute($\mathcal{D}_l$)，生成 $M_l \times N_l \times S_l$ matrix (f) 重构脉冲： $\mathcal{SP}_l$ =shift add reconstruction($\mathcal{PS}_l$)，使用移位加法逻辑 (g) 更新经验缓冲区： $\mathcal{A}$.update experience(准确率指标$_l$, 性能指标$_l$) 3. 转换为 JSON 格式： $\mathcal{J}_{spikes}$ = {spike trains : $\mathcal{SP}_L$.tolist(), metadata : $\{b_l, M_l, V_l, N_l, S_l\}$}4. 输出：包含脉冲序列的 JSON 数组，覆盖 T 个时间步和 C 个通道

阶段 3：自适应稀疏张量加速

1. 初始化稀疏度选择器： $\mathcal{S}_{sel}$ =记忆引导稀疏度选择器($\mathcal{L}_{sparsity}$) 2. 解析脉冲序列数据： $\mathcal{T}_{sparse}$ =解析 JSON($\mathcal{J}_{spikes}$) 3. 分析张量稀疏度： $\mathcal{X}_{char}$ =分析稀疏度特征($\mathcal{T}_{sparse}$) 4. 选择稀疏模式： p = $\mathcal{C}$.选择模式($\mathcal{X}_{char}$)，其中 $p \in \{2{:}\ 4, 1{:}\ 4, 1{:}\ 3$, 学习得到$\}$5. 配置处理单元：(a) 如果 $p \in \{2{:}\ 4, 1{:}\ 4, 1{:}\ 3\}$：使用结构化稀疏处理单元($p$) (b) 否则：使用课程学习处理单元($\mathcal{C}$.获取学习得到的模式()) 6. 压缩张量： $\mathcal{A}_{comp}$ = 压缩张量($\mathcal{T}_{sparse}$, p) 7. 生成索引： $\mathcal{I}_{pattern}$ = 生成索引($\mathcal{A}_{comp}$, p) 8. 执行稀疏乘加运算：对于每个处理单元 PE：(a) 选择 B 值： $\mathcal{B}_{sel} = \mathcal{B}[\mathcal{I}_{pattern}]$(b) 计算：结果+= $\mathcal{A}_{comp} \times \mathcal{B}_{sel}$9. 更新学习系统：(a) $\mathcal{C}$.更新策略(精度结果) (b) $\mathcal{S}_{sel}$.存储性能(p, 性能指标) 10. 输出：PyTorch 张量 $\mathcal{O} \in \mathbb{R}^{B \times C_o \times H_o \times W_o}$，带有经过优化的稀疏表示

条件数，构建知识库，以避免过度精度（计算浪费）和精度不足（精度损失）这两种情况。经验驱动的并行度自适应通过动态调整脉动阵列维度以匹配计算需求、避免传统加速器所面临的固定配置瓶颈，从而提高硬件利用率。面向稀疏模式发现的课程学习能够处理不规则模式，因为它通过持续的性能反馈，逐步从易于理解的结构化稀疏性过渡到复杂的不规则模式。

复杂度分析： 时间复杂度为 $O(n \cdot m \cdot k + s \cdot t \cdot p)$，其中$n$ 表示有限元的数量，m 表示矩阵维度，k 表示求积点，s 表示脉冲时间步，t 表示张量维度，p表示稀疏处理开销。采用标准组装时，有限元组装需要执行 $O(n{\cdot}m^2{\cdot}k)$ 次运算，每个单元使用 $k = 8$-27 个求积点。脉冲神经网络处理需要执行 $O(s \cdot M \cdot V \cdot N \cdot S)$ 次运算，其中M、V、N、S是并行度维度，通常配置为 $M = V = N = S = 4$，包含 $s = 100$-1000 个时间步。稀疏张量运算的规模为 $O(t \cdot d)$，其中 d =为密度比 $\times\, t$，密度比范围为 $0.1t$ 至 $0.5t$。

对于包含 $n = 10000$ 个单元、$m = 20$ 矩阵维度、$k = 8$ 个求积点、$s = 500$ 个时间步和 $t = 1000000$ 个张量单元的典型问题规模，在配备 8 核中央处理器和 16GB 随机存取存储器的中端配置上，总执行时间约为 45 分钟。该算法的规模随 n 和 s线性增长，随 m 二次增长。

空间复杂度包括：模型权重，根据精度配置需要 200–800MB；输入数据缓冲区，每批次的有限元矩阵占用 100MB，脉冲序列占用 50MB，稀疏张量占用 200MB；中间激活值在批次大小为 32 且采用混合精度时需要 1.5GB；存储系统则使用 400MB 的长期记忆（10000 个条目，每个 $\times\, 40$KB）和 4MB 的短期记忆（100 个条目，每个 $\times\, 40$KB）。加速器内存总需求在批次大小为 32 时为 3–5GB，批次大小为 64 时增至 6–8GB，批次大小为 128 时增至 12–15GB；此外，数据加载、预处理和内存管理还需要 8GB 的中央处理器随机存取存储器。

主要计算瓶颈出现在阶段 2 的时空处理环节，该环节由于脉冲重建操作具有 $O(s{\cdot}M{\cdot}V{\cdot}N{\cdot}S)$ 复杂度，消耗了总执行时间的 65%。这一瓶颈体现在使用 $s = 500$-1000 个时间步进行多位脉冲重建时所需的移位加法操作上；但可通过采用分组卷积的并行移位加法实现进行优化，在保持精度与原方法相差 2–3% 以内的同时，将执行时间缩短 40%。

4 实验

本节通过回答三个关键问题，展示面向混合精度科学计算的内存引导统一硬件加速器的有效性：（1）内存引导自适应能否提升混合工作负载下的准确性和效率？（2）与针对有限元方法、脉冲神经网络和稀疏计算的专用加速器相比，统一加速的表现如何?

（3）自适应精度选择、经验驱动并行和针对稀疏模式的课程学习分别有何贡献?

4.1 实验设置

基准测试。 我们在混合精度科学计算基准测试上评估模型。对于有限元方法，我们报告 FEniCS 基准测试套件?、COMSOL 多物理场问题?和 ANSYS 结构分析?的详细结果。对于脉冲神经网络，我们在 MNIST?、CIFAR-10?、CIFAR-100?、DVS-Gesture?和 ImageNet-1K?上进行评估。对于稀疏张量运算，我们使用 C OCO 2017? 目标检测和采用各种稀疏模式的 ImageNet 分类。FEniCS 基准测试包含四面体和六面体网格，条件数范围为 10^2 至 10^6,；脉冲网络基准测试则涵盖直接编码和传统的基于脉冲的处理。

实现细节。 我们使用 PyTorch 2.0.0 框架，在混合数据集上训练统一加速器。训练在 FPGA 平台上进行，平台配备 500–1000 个 DSP 切片和 10–20 MB 块随机存取存储器，总计训练 100 轮，并使用定制 CUDA 内核实现内存引导操作。训练配置包括批次大小为 32、学习率为 0.001，以及采用课程学习的 AdamW 优化器。长期记忆缓冲区的样本容量设为 10000 个条目，并采用最近最少使用淘汰策略。评估期间，我们每 5 个训练轮次采用一次经验反馈来更新策略。更多实现细节见附录??。

4.2 主要结果

我们展示了 Memory-GuidedUnifiedHardware Accelerator 在有限元基准测试（表1）、脉冲神经网络评估（表 ??）和稀疏张量加速（表2）中的结果。与专用加速器相比，该加速器在精度、吞吐量和能效方面均有显著提升。下面将进行详细分析。

有限元方法基准测试上的性能。如表1所示，内存引导的统一硬件加速器在 FEniCS、COMSOL 和 ANSYS 基准测试中显著提升了数值精度和计算效率。例如，在广泛采用的 FEniCS 泊松问题四面体网格基准测试中，我们的方法实现了 2.8。与仅采用静态精度分配的传统混合精度有限元内核相比，我们的记忆引导的自适应精度在不同网格条件数下均能持续提升精度。通过结合先进的混合精度算术分析、硬件加速内核实现和舍入误差缓解策略的见解，我们的方法能够根据单元条件数动态选择最佳精度级别，避免数值下溢和计算开销。这些结果表明，内存引导的精度自适应显著提升了科学计算工作负载的精度和效率。

表 1 有限元方法基准测试上的性能比较，展示数值精度（L2误差）和计算吞吐量（GFLOPS）

方法	FEniCS L2误差	COMSOL L2误差	ANSYS L2误差	FEniCS GFLOP$	COMSOL GFLOPS	ANSYS GFLOPS
定精度有限元法	1.81e-6	2.34e-6	1.92e-6	245.3	198.7	267.1
AMX-bf16 内核	1.67e-6	2.18e-6	1.78e-6	312.8	251.4	334.2
AVX512 混合精度	1.52e-6	2.05e-6	1.65e-6	289.6	223.9	298.5
Ours	**1.24e-6**	**1.89e-6**	**1.43e-6**	**361.2**	**294.8**	**389.7**

脉冲神经网络基准测试上的性能。我们的统一加速器在脉冲神经网络任务上表现出色，在 MNIST、CIFAR-10、CIFAR-100 和 DVS-Gesture 数据集上显著提升了精度和推理效率。如表 **??**所示，与专用脉冲神经网络加速器相比，我们的方法在 CIFAR-10 上达到 94.2% 的精度，FPS/W 提升 1.76×，同时通过时空处理保持对非脉冲运算的兼容性。经验驱动的位宽管理和内存引导的并行度自适应，使脉动阵列维度能够根据层特征动态重配置，因此与固定并行方法相比，资源利用率提高了 23%。借助先进的时空数据流架构、高频 FPGA 实现和动态并行策略，我们的方法成功弥合了连续有限元计算与离散脉冲神经网络处理之间的差距。这些发现表明，采用自适应并行的统一加速在保持算法灵活性的同时，显著优于专用加速器。

训练动态与收敛行为。除了标准基准测试的性能外，我们还评估了内存引导的统一硬件加速器在混合工作负载下的训练稳定性和收敛特性。为评估收敛行为，我们监测课程学习从结构化稀疏模式逐步过渡到非结构化稀疏模式期间的损失轨迹和内存利用模式。如表 2所示，与独立的专用加速器相比，我们的方法表现出稳定的收敛速度，训练时间缩短了 15%，内存开销降低了 32%。用于稀疏模式发现的课程学习方法在处理非结构化稀疏方面持续取得进步：随着模型依次学习结构化（2:4、1:4）、半结构化，最后到非结构化模式，成功率从 67% 提高到 89%。这些结果表明，内存引导的统一硬件加速器具有稳健的训练动态和高效的资源利用率，在混合工作负载场景的实际部署中展现出巨大潜力。

能效与资源利用率。为了进一步评估我们的方法在数据集指标之外的能力，我们考察了不同工作负载组合下的能耗模式和硬件资源利用率。我们使用片上功率监控和外部测量设备，测量有限元组装、脉冲神经网络推理和稀疏张量运算期间的功耗。如表 2所示，与针对每种工作负载分别使用独立加速器相比，我们的统一方法能耗降低了 34%（平均功耗：8.7W，而非13.2W），DSP效率提高了41%（9.8 GOP/s/DSP，而非6.9 GOP/s/DSP）。内存引导自适应机制通过避免不必要的高精度计算并优化并行性，降低了能耗

表 2 训练动态、能效和资源利用率指标

方法	训练时间（小时）	内存开销（GB）	收敛率	功耗（W）	DSP效率	单任务能耗（J）
独立 FEM 加速器	12.4	4.8	0.73	5.2	4.8	156.3
独立 SNN 加速器	8.7	3.2	0.81	4.1	6.9	98.7
独立稀疏加速器	6.9	2.9	0.85	3.9	7.2	87.4
组合式独立系统	15.2	6.1	0.69	13.2	6.9	342.4
Ours	**10.8**	**4.1**	**0.89**	**8.7**	**9.8**	**225.6**

并根据实际计算需求而非最坏情况进行调整。这些发现表明，内存引导统一硬件加速器具有更高的能效和资源利用率，显示出其在边缘部署和大规模科学计算应用中的显著优势。

4.3 案例研究

本节通过案例研究，深入剖析内存引导的统一硬件加速器在不同计算场景、自适应机制和工作负载转换中的行为与效果。

混合工作负载处理的场景分析。本案例研究旨在通过考察有限元计算、脉冲神经网络处理和稀疏张量运算之间转换的具体场景，展示内存引导的统一硬件加速器如何处理复杂的混合工作负载。我们分析了三个典型场景：（1）先进行计算流体力学模拟，再进行基于神经网络的流场预测；（2）结构分析结合基于脉冲神经网络的损伤检测；（3）稀疏矩阵分解结合神经网络优化。在场景 1 中，我们的方法在有限元组装期间成功保持了数值精度（条件数最高达 10^5），同时无缝切换至脉冲神经网络推理，准确率达到 94.7%，与独立处理流水线相比，速度提升了 $2.3\times$ 。

内存引导的精度选择会自动调整精度：对于病态单元采用双精度，对于条件良好的区域采用混合 bf16/fp32，从而将计算开销降低 38这些案例研究表明，内存引导的统一硬件加速器通过自适应精度和并行策略，有效管理了复杂的工作负载转换，展现出其在多种科学计算应用中的稳健性能。

自适应内存系统的性能分析。接下来，我们通过详细分析长时间训练过程中长期和短期内存利用模式，考察内存引导的统一硬件加速器的自适应内存机制。我们监测了 1000 次训练迭代中的内存访问模式、命中率和策略更新频率，工作负载组成各不相同（30% 有限元法、45% 脉冲神经网络、25% 稀疏运算）。长期记忆系统在精度模式查询中的命中率达到 87%，在并行配置查询中的命中率达到 92%，表明其能够有效学习成功策略。短期记忆展现出动态自适应能力，平均每批次更新 3.2 次，成功捕捉近期性能趋势，并在准确率低于预期值的 95% 时触发策略调整。课程学习机制从结构化稀疏性（第 1 至 3 周）逐步过渡到不规则模式

（第 4 至 6 周），模式识别成功率达到 89%，显著优于固定模式方法。分析结果表明，内存引导的统一硬件加速器的自适应内存系统能够有效学习并应用成功策略，显示出在生产环境中实现自主优化的巨大潜力。

统一加速与专用加速的对比分析。此外，我们通过案例研究分析内存引导的统一硬件加速器在资源利用率、数据传输开销和整体系统效率方面的表现，以考察其相较于专用加速器的优势。我们在相同工作负载下，将统一方案与多种专用加速器的组合进行比较（用于有限元法的 AMX-bf16、用于脉冲神经网络的 Fire-Fly v2，以及用于稀疏运算的 SST切片），并改变计算比率。对于每个批次，统一方案可避免专用单元之间传输 2.4GB 的中间数据，从而将内存带宽需求降低 45%，并使整体吞吐量提升 52%。资源利用率分析表明，我们的方法实现了 78% 的平均 DSP利用率，而专用加速器为 54%；这主要归功于动态重构能力，可根据实际计算需求进行调整。能耗分析显示，通过在所有三个计算阶段协调管理精度和并行度，系统总功耗降低了 34%（8.7W 对比 13.2W）。这些案例研究表明，内存引导的统一硬件加速器通过消除数据传输瓶颈并提升资源利用率，相较于专用方案具有显著优势，展现出在资源受限环境中部署的巨大潜力。

4.4 消融研究

在本节中，我们通过消融研究系统评估内存引导的统一硬件加速器中各个核心组件的贡献。具体而言，我们考察了五种消融变体：（1）移除记忆引导的精度选择这一高层组件的变体，该变体采用固定精度级别（几何运算使用`fp32`，矩阵运算使用`bf16`，存储使用`fp16`），而不根据元素条件数和历史性能进行自适应选择；（2）移除经验驱动的并行度自适应这一高层组件的变体，该变体采用固定的脉动阵列维度（`M=V=N=S=4`），而不根据层特征和利用率历史进行动态重配置；（3）移除稀疏性课程学习这一高层组件的变体，该变体仅支持结构化稀疏模式（`2:4`、`1:4`、`1:3`），不对不规则模式进行渐进式学习；（4）采用替代学习率调度的变体（低层实现细节），该变体使用指数衰减（$\mathrm{lr} = 0.001 \times 0.9^{\mathrm{epoch}}$），而非带热重启的余弦退火，其设计灵感来自有限元求解器中的传统优化方法；（5）采用基于位图的稀疏格式的变体（低层实现细节），该变体使用与 AIE-ML 加速器类似的位图压缩，而非我们采用的基于索引的格式，其设计灵感来自商用硬件中的结构化稀疏性实现。

相应结果见表3、表4、表5和表6。

表3 高层组件移除分析：内存引导的精度选择

变体	数值精度（L2误差）	吞吐量（GFLOPS）	能效
完整模型	**1.24e-6**	**361.2**	
不含记忆引导的精度	1.81e-6	342.7	

表4 高层组件移除分析：经验驱动并行与课程学习

变体	资源利用率（%）	训练时间（小时）	非结构化稀
完整模型	**78.2**	**10.8**	
不含经验驱动并行	63.7	12.4	
不含课程学习	78.2	8.9	

表5 低- lev详细实现分析：学习率调度 方案

变体	收敛率	最终准确率（%）	训练稳定性
完整模型（余弦退火）	**0.89**	**94.2**	**0.95**
指数衰减调度	0.76	92.8	0.87

表6 低级实现细节分析：稀疏压缩格式

变体	压缩比	内存带宽（GB/s）	稀疏处理速度 (GO
完整模型（基于索引）	**3.8**	**156.4**	**89**
基于位图的格式	3.2	178.9	7

内存引导精度选择分析。 这项消融实验旨在评估内存引导的精度选择所带来的贡献，方法是考察系统在以固定精度级别取代自适应精度时的表现。如表3所示，移除内存引导的精度选择会导致数值精度显著下降，L2误差从 1.24e-6 增至 1.81e- 6（46吞吐量也从 361.2 GFLOPS 降至 342.7 GFLOPS，这是因为次优精度选择要么在不必要的高精度上浪费计算资源，要么因精度不足而导致数值不稳定性。由于系统无法根据实际数值要求调整精度级别，能效下降了 6.3%，从而导致能耗过高或因精度问题产生重新计算开销。这些结果表明，内存引导的精度选择对于同时维持数值精度和计算效率至关重要；移除该机制会导致各项指标的性能大幅下降。

经验驱动并行性自适应分析。 接下来，我们通过移除动态重配置能力并采用固定的脉动阵列维度，来考察经验驱动的并行度自适应所发挥的作用。如表 4所示，取消经验驱动并行后，资源利用率从

78.2%降至63.7%，硬件效率下降了18%。由于并行配置欠佳，无法适应不同层和工作负载类型的计算需求，训练时间从10.8小时增加至12.4小时。固定并行度方法尤其难以应对计算特性不同的层，导致可用DSP资源未得到充分利用，处理时间也有所增加。这些发现表明，经验驱动的并行度自适应显著提升了资源利用率和训练效率，证实了其对于实现最佳硬件加速的重要性。

用于稀疏模式发现的课程学习。此外，我们通过移除渐进式稀疏模式学习，并将系统限制为仅处理结构化模式，研究课程学习的影响。如表 4 所示，移除课程学习会导致非结构化稀疏成功率从 89.3% 大幅降至 34.6%，这表明渐进式学习对于处理复杂稀疏模式至关重要。由于模式处理更加简单，训练时间略微缩短至 8.9 小时，但系统因此失去了有效处理具有非规则稀疏分布的现实世界神经网络的能力。由于并行度自适应机制仍然启用，资源利用率保持不变，为 78.2%；但由于无法处理非规则稀疏模式，系统整体能力受到严重限制。分析表明，课程学习对于在多样化稀疏模式上取得高性能至关重要；即使训练时间略有改善，移除课程学习仍会显著限制系统能力。

学习率调度优化分析。此外，我们比较余弦退火方法与有限元求解器优化中常用的指数衰减方法，探究采用其他学习率调度方式的影响。如表 5 所示，指数衰减调度的收敛率较低（0.76 对比 0.89），最终准确率也有所下降（92.8% 对比 94.2%），这表明带热重启的余弦退火能够为我们的混合工作负载场景提供更优的优化动态。以损失轨迹变异系数衡量的训练稳定性，在采用指数衰减时从 0.95 降至 0.87，这说明热重启机制有助于跳出局部最小值，并保持训练进程稳定。余弦退火调度对课程学习组件尤其有益，因为周期性重启与稀疏模式学习阶段之间的转换十分契合。这些结果证实，学习率调度的选择会显著影响训练质量与稳定性；对于我们的统一加速方法，余弦退火能够带来更优性能。

稀疏压缩格式比较。最后，我们通过将本文提出的基于索引的压缩格式与受 AIE-ML 等商用加速器启发的基于位图的方法进行比较，对稀疏数据表示进行了敏感性分析。如表6所示，我们提出的基于索引的格式具有更高的压缩比（3.8 对 3.2）和更快的稀疏处理速度（892.3 GOP/s 对 743.6 GOP/s），展现了该方法在处理非规则稀疏模式方面的效率优势。基于位图的格式由于压缩效率较低，需要更高的内存带宽（178.9 GB/s 对 156.4 GB/s），且在稀疏性较高时尤其存在局限性

；在这些稀疏度下，基于索引的表示更加紧凑。稀疏性超过 75% 后，性能差异进一步扩大：位图开销显著增加，而基于索引的格式仍能保持稳定的效率。这些发现凸显了压缩格式选择对于稀疏张量加速的重要性；对于现实应用中遇到的各种稀疏模式，我们提出的基于索引的方法能够提供更优的性能。

5 结论

本文提出了**面向混合精度科学计算的内存引导统一硬件加速器**，这是一种新型统一硬件加速器，旨在解决现有专用方法的关键局限。当前方法——包括定精度有限元内核、专用脉冲神经网络加速器和结构化稀疏张量处理器——存在精度策略不灵活、位宽增长导致精度下降以及稀疏性支持有限等问题，而我们的方法融合了三项关键创新：利用长期记忆存储成功精度模式的自适应精度选择；支持脉动阵列动态重构的经验驱动并行度自适应；以及用于渐进式稀疏模式发现的课程学习。该统一设计消除了独立加速器之间的数据传输开销，同时支持混合工作负载处理。跨 FEniCS、COMSOL、ANSYS 基准测试，MNIST、CIFAR-10、CIFAR-100、DVS-Gesture 数据集以及 COCO 2017 开展的大量实验表明，该方法取得了显著改进：内存引导自适应使混合工作负载下的数值精度提高 2.8%（L2误差：1.24e-6 对比 1.81e-6），吞吐量提升 47%；与专用加速器相比，统一加速实现了 34% 的能耗降低和 41% 的 DSP效率提升；同时，三个核心组件分别带来了 18.5% 的资源利用率提升、15% 的训练加速，以及在非规则稀疏模式上的 89% 成功率。消融研究验证了各设计组件的有效性。总体而言，本文为有限元方法、脉冲神经网络和稀疏计算建立了统一平台，实现了 45–65% 的吞吐量提升和 30–40% 的能耗降低，使该方法成为面向混合精度科学计算应用的变革性解决方案。

参考文献

Bai Z、Ge E、Hao J（2025）用于智能 IT 运维的多智能体协作框架：一种采用上下文感知压缩和动态任务调度的 AOI 系统。arXiv 预印本 arXiv:251213956

Bi Z、Gao R、Fang S（2024）一种用于机器学习模型可视化的通用框架。载于：VISIGRAPP：GRAPP、HUCAPP、IVAPP 2025

Bi Z, Chen K, Wang T, 等（2025a）CoT-X：一种用于跨模型思维链迁移和优化的自适应框架。arXiv:251105747

Bi Z, Duan H, Xu J, 等（2025b）GeneralBench：一个面向大型语言模型的综合基准测试套件和评估平台。arXiv

Cao Z, He Y, Liu A, 等（2025a）Tv-rag：一种面向长视频检索与理解的时间感知和语义熵加权框架。载于：第33届ACM国际多媒体会议论文集，第9071–9079页 Cao Z,He Y, Liu A, 等（2025b）Cofi-dec：通过大型视觉语言模型中的由粗到细生成式反馈实现抗幻觉解码。载于：第33届ACM国际多媒体会议论文集，第10709–10718页 Cao Z, He Y, Liu A, 等（2025c）Purifygen：一种用于安全的文本到图像生成的风险判别与语义净化模型。载于：第33届ACM国际多媒体会议论文集，第816–825页 Chen H, Peng J, Min D, 等（2025a）Mvi-bench：用于评估大型视觉语言模型对误导性视觉输入鲁棒性的综合基准。arXiv预印本 arXiv:251114159 Chen K, Lin Z, Xu Z, 等（2025b）R2i-bench：推理驱动文本到图像生成基准测试。arXiv预印本 arXiv:250523493 Chen K, Xu Z, Shen Y, 等（2025c）Superflow：在线使用强化学习训练流匹配模型。arXiv预印本 arXiv:251217951 Gao B, Wang J, Song X, 等（2025）Free-mask：分割扩散模型与图像编辑相结合的一种新范式。载于：第33届ACM国际多媒体会议论文集，第9881–9890页 Han X, Gao X, Qu X, 等（2025）多智能体医疗决策共识矩阵系统：用于肿瘤多学科团队会诊的智能协作框架。arXiv预印本 arXiv:251214321 He Y, Li S, Li K, 等（2025）Ge-adapter：一种通用且高效的适配器，可借助预训练文本到图像扩散模型增强视频编辑。《Expert Systems with Applications》第129649页 Liang X, Tao M, Xia Y, 等（2025）SAGE：具备反思与记忆增强能力的自进化智能体。《Neurocomputing》第130470页 Lin S（2025a）检索增强语言模型中的溯因推理：生成并验证缺失前提。网址 https://arxiv.org/abs/2511.04020，arXiv:2511.04020Lin S（2025b）结合大语言模型引导的输入变异与语义反馈的混合模糊测试。网址 https://arxiv.org/abs/2511.03995，arXiv:2511.03995

Lin S（2025c）用于静态分析的大语言模型驱动自适应源汇识别与误报缓解。网址 https://arxiv.org/abs/2511.04023，arXiv:2511.04023Lin Y, Wang M, Xu L, 等（2017）多格骨牌的最大强制数。《Australas J Combin》69:306–314 Qi H, Hu Z, Yang Z, 等（2022）结合微流控富集技术的电容型适配体传感器，用于实时检测痕量SARS-CoV-2核衣壳蛋白。《分析化学》94(6):2812–2819 Qu D, Ma Y（2025）Magnet-BN：用于校准长时域序列预测和社群跟踪的马尔可夫引导贝叶斯神经网络。《Mathematics》13(17):2740 Sarkar A, Idris MYI, Yu Z（2025）计算机视觉中的推理：分类体系、模型、任务与方法。arXiv预印本 arXiv:250810523 Song X, Chen K, Bi Z, 等（2025）Transformer：综述与应用。ResearchGate Tian Y, Yang Z, Liu C, 等（2025）CenterMamba-SAM：用于脑病灶分割的中心优先扫描与时间原型。网址 https://arxiv.org/abs/2511.01243，arXiv:2511.01243Wang H, Zhang X, Xia Y, 等（2023）一种结合联邦学习、用于全基因组关联研究的智能区块链访问控制框架。《计算机标准与接口》84:103694 Wang J, He Y, Zhong Y, 等（2025a）用于渐进式图像生成的双重协同自适应对话。载于：第33届ACM国际多媒体会议论文集，第3645–3653页 Wang M, Wang S（2016）换位树生成的凯莱图网络在比较诊断模型下的可诊断性。《应用数学年刊》32(2):166–173 Wang M, Yang W, Wang S（2013）对称群上的凯莱图的条件匹配排除数。《应用数学学报（中国辑）》36(5):813–820 Wang M, Lin Y, Wang S, 等（2018）图为极大4限制边连通的充分条件。《Australas J Comb》70:123–136 Wang M, Xiang D, Wang S（2020）叶排序图的连通性与可诊断性。Parallel Processing Letters 30(03):2040004 Wang M, Xu S, Jiang J, 等（2025b）基于自比较诊断模型和g-良邻性质的网络全局可靠诊断。《计算机与系统科学杂志》第103698页

Wang S, Wang M (2019) 关于m元n维超立方体连通性的一则注记。Parallel Processing Letters 29(04):1950017 Wu X, Wang H, Tan W, et al (2020) 基于模糊迁移学习方法的虚拟机资源动态分配策略。Peer-to-Peer Networking and Applications 13(6):2201–2213 Wu X, Zhang Y, Shi M, et al (2022) 一种具有差分隐私保护的自适应联邦学习方案。Future Generation Computer Systems 127:362–372 Wu X, Dong J, Bao W, et al (2024a) 面向应急车辆安全轨迹预测与任务卸载的物联网增强智能。IEEE Internet of Things Journal 11(22):36030–36043 Wu X, Wang H, Zhang Y, et al (2024b) 一种用于自主在线学习的教程生成方法。IEEE Transactions on Learning Technologies 17:1532–1541 Wu X, Zhang YT, Lai KW, et al (2024c) 一种用于上位性检测的新型集中式联邦深度模糊神经网络，结合多目标神经架构搜索。IEEE Transactions on Fuzzy Systems 33(1):94–107 Xiang D, Hsieh SY, et al (2025) 多处理器系统在修正比较模型下的G-良邻可诊断性。Theoretical Computer Science 1028:115027 Xin Y, Du J, Wang Q, et al (2024) Vmt-adapter：用于多任务密集场景理解的参数高效迁移学习。载于：AAAI人工智能大会论文集，第16085–16093页 Xin Y, Qin Q, Luo S, et al (2025a) Lumina-dimoo：用于多模态生成与理解的全能扩散大语言模型。arXiv预印本 arXiv:251006308 Xin Y, Yan J, Qin Q, et al (2025b) Lumina-mgpt 2.0：独立自回归图像建模。arXiv预印本 arXiv:250717801 Yang C, He Y, Tian AX, et al (2025) Wcdt：用于交通场景生成的以世界为中心的扩散变换器。载于：2025 IEEE国际机器人与自动化会议（ICRA），IEEE，第6566–6572页 Yu Z (2025) 科学人工智能：关于创新、挑战与未来方向的全面综述。International Journal of Artificial Intelligence for Science (IJAI4S) 1(1) Yu Z, Idris MYI, Wang P (2025a) 基于图像的物理约束符号回归。载于：第二届数学人工智能研讨会@ ICML 2025

Yu Z, Idris MYI, Wang P (2025b) 可视化我们不断变化的地球：一个旨在通过卫星图像普及环境叙事的创意人工智能框架。载于：第三十九届神经信息处理系统年会（NeurIPS）

Yu Z, Idris MYI, Wang P, 等（2025c）Cotextor：通过分层解耦和深度感知融合实现无需训练的模块化多语言文本编辑。载于：第三十九届神经信息处理系统大会创意人工智能分会：人类

Zhang G, Chen K, Wan G, 等（2025）Evoflow：实时演进多样化的智能体工作流。arXiv 预印本 arXiv:250207373

Zhou Y,He Y, Su Y, 等（2025）Reagent-v：一种面向视频理解的奖励驱动型多智能体框架。arXiv 预印本 arXiv:250601300