IPSI TIR
JOURNAL

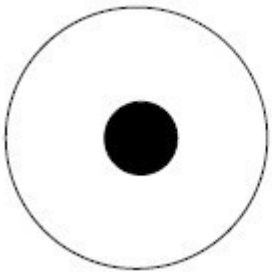
IPSI Transactions on Internet Research

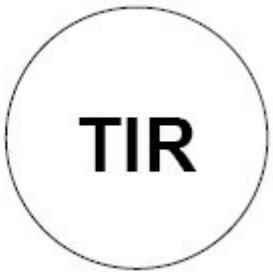
From Silicon to Society: Efficiency, Security, and Responsible Integration of Artificial Intelligence

Guest Editor: Ivan Ratković

OCTOBER 2026, VOLUME 22, ISSUE 4

A publication of IPSI Internet Research Society
Petah Tikva, Indianapolis, Tokyo, and Belgrade
ISSN 1820-4503
DOI: 10.58245/ipsi.tir





The IPSI Transactions on Internet Research

Multi-, Inter-, and Trans-disciplinary Issues in Computer Science and Engineering

A publication of IPSI Internet Research Society Petah Tikva, Indianapolis, Tokyo, and Belgrade
October 2026 Volume 22 Number 4 (ISSN 1820-4503)

From Silicon to Society: Efficiency, Security, and Responsible Integration of Artificial Intelligence

Guest Editor: Ivan Ratković

Table of Contents:

Editorial
Ivan Ratković ……..1

Workflow Systems and Intelligent Data Integration for Personalized Therapeutics in Brain Disorders
The article is undergoing changes ……..5

CopyCNN: Revere Engineering based Extraction of Convolutional Neural Networks
Lusky, Yehonatan; Mendelson, Avi; and Nemcovsky, Yaniv ……..……...6

The MKA Law and the Milutinovic–Katulevskiy Architecture: ´ A Post-Moore Roadmap for AI-Scale, Photonic, and Quantum-Adjacent Computing
Katulevskiy, Daniil; Katulevskaya, Elizaveta; Katulevskiy, Aleksey; and Milutinovic, Veljko ….....16

Students' Awareness, Perceptions, and Use of Generative AI Tools in Higher Education
Radaković, Davorka; Steingartner, William; and Seleši, Dora …..……..…….42

ForiC-a Fast and Effective Method for Predicting the Origin of Bacterial Chromosome Replication
Stojićević, Marko ….....61

The IPSI Internet Research Society

The IPSI Internet Research Society is an association of people with professional interests in the field of the Internet. All members will receive IPSI Transaction Journals upon registering at the Society.

Member copies of Transactions are for personal use only
IPSI TRANSACTIONS ON ADVANCED RESEARCH
www.ipsitransactions.org

Editorial Board

Veljko Milutinovic **Co-Editor-in-Chief**	**Jakob Salom** **Co-Editor-in-Chief**	**Nenad Korolija** **Co-Editor-in-Chief**
Department of Computer Science University of Indiana Bloomington Bloomington, Indiana, USA vm@etf.rs	Department of Computer Science Mathematical Institute of SANU Belgrade, Serbia ipsi.journals@gmail.com	IPSI Internet Research Society Dalmatinska 55 Belgrade, Serbia ipsi.journals@gmail.com
Radović Marković, Mirjana	**Gonzalez, Victor**	**Milligan, Charles**
Belgrade Bankers' Academy Belgrade,	University of Oviedo, Gijon,	StorageTek, Colorado,
Serbia	Spain	USA
Filipović, Nenad	**Daković, Nevena**	**Steingartner, William**
Faculty of Engineering, University of Kragujevac,	Faculty of Dramatic Arts Belgrade,	Faculty of Electrical Engineering and Informatics, Košice,
Serbia	Serbia	Slovakia
Marinković, Marko	**Jutla, Dawn**	**Neuhold, Erich**
Faculty of Civil Engineering, University of Belgrade	Sant Marry's University, Halifax,	UNIWIE, Vienna,
Serbia	Canada	Austria
Domenici, Andrea	**Karabeg, Dino**	**Piccardi, Massimo**
University of Pisa, Pisa,	Oslo University, Oslo,	Sydney University of Technology, Sydney,
Italy	Norway	Australia
Flynn, Michael	**Kiong, Tan Kok**	**Miljanic, Scepan**
Stanford University, Palo Alto, California,	National University of Singapore,	The School of Physical Chemistry, Belgrade,
USA	Singapore	Serbia
Rishe, Naphtali David	**Kovacevic, Branko**	**Rutledge, Chip**
Florida International University Miami,	The School of Electrical Engineering, Belgrade,	Purdue Discovery Park, Indiana,
USA	Serbia	USA
Ganascia, Jean-Luc	**Patricelli, Frederic**	**Mester, Gyula**
Paris University, Paris,	ICTEK Worldwide, Pizzoli, L'Aquila,	Óbuda University, Budapest,
France	Italy	Hungary

Editorial

From Silicon to Society: Efficient, Secure, and Responsible Computing in the Age of Artificial Intelligence

Guest Editor: Ivan Ratković

From Silicon to Society: Efficient, Secure, and Responsible Computing in the Age of Artificial Intelligence

Guest Editor: Ivan Ratković

The progress of artificial intelligence (AI) is no longer determined by algorithms and models alone. It depends equally on the efficiency of the hardware and algorithms that execute computational workloads, on the security of models once they are deployed, and on the ability of organizations to integrate AI responsibly into everyday practice. This special issue brings together contributions that address these challenges at different levels - from post-Moore computing substrates and the confidentiality of trained neural networks to AI-supported clinical workflows, the use of generative AI in higher education, the data-driven operation of electric power grids on the path to decarbonization, and lightweight algorithms for large-scale genome analysis.

Although the papers belong to distinct research fields, they share a common perspective: rather than evaluating an isolated component, they examine how technology behaves as part of a larger system. They show how total energy and data movement determine the efficiency of AI hardware, how the outputs of a deployed model can reveal the model itself, how physics-based models and machine learning can be combined to manage uncertainty in the power grid, how an algorithm that relies only on intrinsic sequence signals can make genome analysis fast and independent of external databases, and how data, analytics, institutional policies, and human judgment must be coordinated for AI to deliver real value in hospitals and universities. By combining formal, experimental, and empirical approaches, the issue aims to stimulate dialogue between the hardware, security, and application communities.

1. The paper **"Workflow Systems and Intelligent Data Integration for Personalized Therapeutics in Brain Disorders"** by Daniel A. Ardila-Herrera, Laura Dipietro, Ciro Ramos-Estebanez, Paulo E. P. Teixeira, and Tim Wagner addresses personalized therapeutics in brain disorders from an information-systems perspective. The authors observe that sensing technologies and AI have advanced faster than the clinical information systems and care workflows expected to use them, so that sensing devices, analytic tools, and decision-support systems often remain only partially connected. In response, they propose a workflow-integrated framework that organizes therapeutic care into four phases - data capture, analytic processing, decision integration, and feedback with continuous adaptation - implemented through a layered architecture in which a workflow orchestration layer coordinates data acquisition, analytics, and the clinical interface. The framework is illustrated through the Integrated Motion Analysis Suite (IMAS), with published case studies in stroke rehabilitation and Parkinson's disease; in a cohort of 28 stroke patients, for example, the system predicted Fugl-Meyer scores and motor recovery potential with R^2 values of 0.75 and 0.83, respectively. Opioid use disorder and chronic pain are discussed as ongoing extension cases. By also examining cloud and federated deployment, interoperability, governance, and barriers to clinical adoption, the paper makes a convincing case that workflow orchestration is a key information-systems function for translating multimodal data and AI into clinical use.

2. The paper **"CopyCNN: Reverse Engineering Based Extraction of Convolutional Neural Networks"** by Yehonatan Lusky, Avi Mendelson, and Yaniv Nemcovsky addresses model extraction, a growing threat to the security and intellectual property of AI systems. The authors present CopyCNN, an attack that recovers both the architecture and the precise weights of black-box, ReLU-based convolutional neural networks using only query access - to their knowledge, the first method capable of recovering exact weights from such networks. Building on cryptanalytic extraction techniques previously demonstrated for multilayer perceptrons, which exploit the critical points of piecewise-linear ReLU networks, the method introduces dedicated strategies for convolutional, residual, batch-normalization, and pooling layers, and

combines graph-based weight correlation and unification with mixed-integer linear programming for global sign recovery. The architecture itself is inferred by testing candidate layer configurations and checking whether the recovered weights form coherent clusters. Experiments demonstrate the successful extraction of two LeNet-5 variants and a ResNet block, as well as higher extraction precision than prior methods on multilayer perceptrons, often with fewer queries. The authors openly discuss the current limitations - accumulated numerical error in deeper networks, the need for access to logits, and the large number of adaptive queries - and outline countermeasures such as query monitoring, label-only outputs, input randomization, and moving-target defenses. For designers and operators of AI systems, the paper is a timely reminder that the weights of a deployed model may be far less private than is commonly assumed.

3. The paper **"The MKA Law and the Milutinović–Katulevskiy Architecture: A Post-Moore Graph-Fabric Roadmap for AI-Scale, Photonic, and Quantum-Adjacent Computing"** by Daniil Katulevskiy, Elizaveta Katulevskaya, Aleksey Katulevskiy, and Veljko Milutinović argues that the next major efficiency jump in computing will not come from device density alone, but from jointly reducing the total task energy and the effective graph depth of workloads executed at fixed quality. The authors formalize this idea through a fixed-workload graph acceleration density metric and the MKA Law, and identify a Compound Gain Regime in which reductions in energy and depth reinforce each other multiplicatively rather than being traded against one another. Building on the Honeycomb architecture lineage originated by Veljko Milutinović, the paper defines a substrate-independent graph fabric spanning six regimes: reconfigurable FPGA/CGRA overlays, analog fabrics, processing-in-memory and memristive fabrics, silicon-photonic fabrics, quantum-dot substrates, and entanglement-assisted quantum/hybrid graph edges. A recently reported analog circuit that solves symmetric positive-definite linear systems in time independent of matrix size anchors the analog regime, while large language model (LLM) training and inference, modeled as token-flow graphs, serve as the first industrial test case, including a conditional projection of billion-dollar-class effects for AI data centers. Particularly valuable from a computer-architecture perspective is the insistence on full-task energy accounting - covering data movement, control, conversion, calibration, and cooling rather than isolated energy per operation - together with explicit falsification criteria, a benchmark suite, and a staged research agenda that make the roadmap testable.

4. The paper **"Students' Awareness, Perceptions, and Use of Generative AI Tools in Higher Education"** by Davorka Radaković, William Steingartner, and Dora Seleši presents an international empirical study of how students use and perceive generative AI in their studies. A structured questionnaire was administered in two phases to students at five universities in Poland, Slovakia, Slovenia, and Serbia, yielding 227 consenting respondents, and the data were analyzed with considerable statistical care, including confidence intervals, nonparametric comparisons, effect sizes, correction for multiple comparisons, and sensitivity analyses adjusted for recruitment phase and institution. The results show near-universal adoption: 94.3% of respondents had used AI tools, most of them several times a week, primarily for problem solving and programming. Positive perceptions of AI as a learning aid coexist with widespread experience of its errors - almost all users had encountered incorrect output, yet most still reported at least some trust in AI tools, pointing to conditional reliance rather than uncritical confidence. The study also reveals a clear governance gap: only about a quarter of AI users knew that their institution had AI-use guidelines, whereas most supported formal policies and explicit instruction in ethical use, and nearly half reported using AI to generate complete homework tasks. Gender differences proved limited once multiple comparisons were accounted for. The authors conclude that universities should respond with clear guidance, AI-literacy training, verification practices, and responsible-use policies rather than with simple promotion or prohibition of AI tools.

5. The paper **"ForiC - a Fast and Effective Method for Predicting the Origin of Bacterial Chromosome Replication"** by Marko Stojićević addresses a fundamental task in computational genomics: locating the origin of replication (oriC) in bacterial chromosomes. Classical GC-skew and cumulative GC-skew (CGC) analyses are computationally simple but often imprecise, whereas more accurate approaches based on DnaA-box detection and comparative genomics are resource-intensive and depend on external biological information. ForiC bridges this gap with a deterministic, linear-time rule that combines three signals derived directly from the genome sequence - the CGC global maximum, the CGC inflection point associated with it, and the alignment of that point with the GC global minimum - without motif detection, homology searches, or external databases. Calibrated and evaluated on 36 bacterial genomes with experimentally validated origins, using five phylum-stratified training/test splits, ForiC reached a hit accuracy of 90.91% on a representative test subset with a two-window reporting interval, clearly outperforming the GC and CGC skew baselines. Compared with the reference tool OriFinder, which achieves higher positional

precision but often reports multiple candidate regions and relies on BLAST searches, ForiC always returns a single, directly interpretable prediction, remained comparatively stable when oriC-associated regions were removed, and ran in 0.75 seconds versus about 28 minutes in the reported example. Positioned by the author as a screening-oriented tool that complements rather than replaces reference-assisted methods, ForiC is particularly attractive for rapid, large-scale, database-independent genome analysis.

Even though the papers in this issue address different domains and levels of abstraction, they converge on a common conclusion: the impact of AI and other computational methods is determined not only by the capabilities of models and algorithms, but also by the efficiency with which they are executed, the protection of the knowledge they embody, their grounding in the physics of the infrastructure they help to operate, and the workflows, policies, and competencies that govern their use. We hope that readers will find these contributions useful and that they will inspire further research across disciplinary boundaries. We sincerely thank all the authors for their contributions and the reviewers for their careful and constructive evaluations.

===== About guest editor =====

Ivan Ratković, PhD, is a computer architect with twenty years of experience in designing, architecting, developing, and taping out energy-efficient multicore processors and hardware accelerators for artificial intelligence (AI) and high-performance computing (HPC). He is a Chief Architect and CTO of a company developing area- and power-efficient HPC/AI processors, where he leads architecture, design, and verification, with a special focus on the dot-product unit, the RISC-V IOMMU, and the RISC-V vector processing unit. Previously, at Esperanto Technologies, he served as Power Manager and Architect, leading all power- and energy-efficiency efforts - from applications and the operating system down to silicon - for a RISC-V system-on-chip for HPC and AI. Earlier, as a member of the company's senior VLSI technical staff, he worked on vector/tensor unit design, power estimation and optimization, low-power and full-chip verification, gate-level sign-off, and silicon validation, contributing to the successful tape-out of a 24-billion-transistor chip with more than a thousand RISC-V vector cores.

Dr. Ratković received his PhD in Computer Architecture, cum laude, from the Universitat Politècnica de Catalunya (UPC-BarcelonaTech), within a doctoral program run jointly with the Barcelona Supercomputing Center (BSC) and completed partly at UC Berkeley (International Doctorate). His thesis addressed the design of power- and energy-efficient functional units for vector processors. He has published in venues such as IEEE Transactions on VLSI Systems, ACM Transactions on Architecture and Code Optimization, ISLPED, and the Journal of Big Data, serves as a reviewer for IEEE Transactions on VLSI Systems, and has served on a PhD defense committee at UPC-BarcelonaTech. His interests include computer architecture, vector processors, floating-point arithmetic, low-power and energy-efficient design, VLSI, RISC-V, and machine learning. Email: ivan.ratkovic.bg@gmail.com, ORCID ID: 0000-0002-0524-7227.

Workflow Systems and Intelligent Data Integration for Personalized Therapeutics in Brain Disorders

The article is undergoing Changes

CopyCNN: Reverse Engineering Based Extraction of Convolutional Neural Networks

Lusky, Yehonatan; Mendelson, Avi; and Nemcovsky, Yaniv

Abstract: ***Neural network extraction attacks pose a significant threat to the security and intellectual property of AI systems, enabling adversaries to replicate proprietary models, compromise their confidentiality, or exploit their functionality. In this paper, we present CopyCNN, a novel attack that extracts the exact weights and architecture of black-box, deep, ReLU-based convolutional neural networks (CNNs) using only query access. To the best of our knowledge, CopyCNN is the first method capable of recovering precise weights from black-box CNNs. It effectively reverse engineers key components—including fully connected, convolutional, residual, pooling, and batch normalization layers—enabling the extraction of architectures such as AlexNet, LeNet-5, and variants of VGG and ResNet. We establish the theoretical basis for CopyCNN and demonstrate its effectiveness through the successful extraction of several architectures. Our results highlight the practicality of CopyCNN for convolutional neural network extraction and its implications on AI security and computer vision applications.***

Index Terms: ***Convolutional neural networks, model extraction attack, reverse engineering***

1. Introduction

Convolutional neural networks (CNNs) have become a cornerstone of modern machine learning, driving breakthroughs in computer vision and related tasks such as image classification, object detection, and facial recognition. As these models grow increasingly complex and effective, the intellectual property (IP) embedded in their architectures and parameters becomes highly valuable—particularly for organizations that develop proprietary CNNs to maintain a competitive edge.

The widespread deployment of neural networks in real-world applications [1] has also introduced new security challenges. Among these, neural network extraction has emerged as a significant threat: it enables adversaries to reverse-engineer trained models to uncover their architecture and parameters. A successful extraction allows attackers to replicate a model's functionality without access to the original training data or computational resources. This poses serious risks to model owners, including intellectual property theft [2], the facilitation of adversarial attacks [3], [4], model poisoning [5], and even training data reconstruction [6]. Many of these attacks are far more difficult, or even infeasible, when the model is accessed purely as a black box [7].

1.1. Related Work

Neural network extraction has been studied across various attack surfaces, including oracle-based methods, side channels, and fault injection. Oracle-based attacks aim to train a substitute model by classifying data according to the target model [8], [9]. These attacks replicate decision boundaries without access to internal parameters, and can be optimized through query synthesis and hyperparameter tuning [10]. In contrast, side-channel techniques passively infer model information through observable signals such as power consumption or memory access patterns [11]–[14], while fault injection methods actively perturb the system (e.g., voltage or EM glitches) to recover internal parameters [15].

A more recent and powerful approach involves reverse engineering, which uses carefully chosen inputs to directly infer architectural components and even the exact weights of the target model. This method, inspired by cryptanalytic chosen-plaintext attacks, achieves high precision with fewer queries and has shown promising results against MLPs [16]–[19]. Our work builds on this line of research allowing to reverse engineering convolutional architectures.

1.2. Our Contribution

We present **CopyCNN**, the first reverse-engineering-based model extraction attack capable of extracting both the architecture and the precise weights of ReLU-based convolutional neural networks (CNNs). While prior work has focused on multilayer perceptrons (MLPs), CopyCNN addresses the structural challenges of CNNs, including convolutional layers, residual connections, pooling, and normalization. We formalize the attack, implement it end-to-end, and demonstrate its effectiveness across multiple architectures, achieving high fidelity with low query complexity.

2. Preliminaries

In this section, we introduce key notations and review essential concepts from previous attacks that are fundamental towards understanding our proposed attack. While we strive to maintain consistency with the notations used in prior work

for comparability, we introduce adjustments where necessary to align more closely with the specifics of our method, ensuring clarity and accuracy.

2.1. Notations

Notation 1: Let X and Y denote the input and output spaces, respectively. Given an input $x \in X$, let $F : X \to Y$ represent a neural network, where $y = F(x) \in Y$ is the output logit vector. The network is modeled as a composition of k functions, each corresponding to a layer:

$$y = F(x) = F_k \circ F_{k-1} \circ \cdots \circ F_1(x) \tag{1}$$

Notation 2: Let $h_x^{(k)}$ denote the output of the k-th layer for input x. It is recursively defined by:

$$h_x^{(k)} = F_k(h_x^{(k-1)}), \quad \text{with } h_x^{(0)} = x \tag{2}$$

Notation 3: We denote common architectural components as follows:

$$\begin{aligned} f &\leftarrow \text{Fully connected layer} \\ c &\leftarrow \text{Convolutional layer} \\ \sigma &\leftarrow \text{ReLU activation function} \\ \text{BN} &\leftarrow \text{Batch Normalization} \\ \text{AvgPool} &\leftarrow \text{Average Pooling} \\ \text{MaxPool} &\leftarrow \text{Max Pooling} \\ \text{MinPool} &\leftarrow \text{Min Pooling} \end{aligned}$$

Notation 4: A fully connected layer is modeled as an affine transformation. Let $A^{(k)} \in \mathbb{R}^{m \times n}$ be the weight matrix and $B^{(k)} \in \mathbb{R}^m$ be the bias vector of the k-th layer. Then, for input $x \in \mathbb{R}^n$, the output is:

$$f_k(x) = A^{(k)}x + B^{(k)} \tag{3}$$

Notation 5: Let $\nu(\eta; x)$ denote the pre-activation value (i.e., before applying the activation function) of neuron η when the network is evaluated on input x.

Notation 6: A neuron η is said to be at a *critical point* for input x if its pre-activation value is zero, i.e., $\nu(\eta; x) = 0$.

Notation 7: The set of all inputs for which a neuron η is at a critical point is denoted $W(\eta)$:

$$W(\eta) = \{x \in X \mid \nu(\eta; x) = 0\} \tag{4}$$

Notation 8: Let e_j denote the j-th standard basis vector in $\mathbb{R}^n$ — a vector with 1 in the j-th coordinate and 0 elsewhere.

2.2. Network's Piecewise Linearity and Critical Points

The success of our extraction method relies on the piecewise linear nature of ReLU-based neural networks. While fully connected and convolutional layers are inherently linear, the ReLU activation function introduces non-linearities that partition the input space into distinct linear regions. Within each region, the network behaves as a linear function.

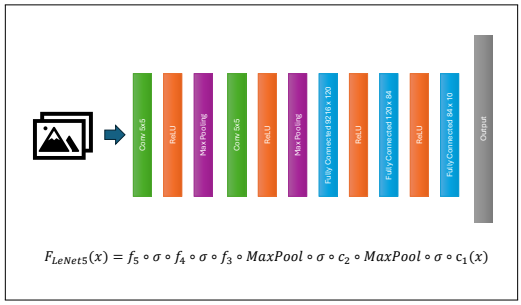


Fig. 1. Architecture of a modernized LeNet-5 neural network variant incorporating ReLU activation function

Formally, the ReLU activation at layer k can be modeled by a diagonal matrix $I^{(k)}$, where each entry is 1 for active neurons and 0 otherwise:

$$I^{(k)} = \begin{cases} 1 & \text{if } \nu(\eta; x) > 0, \\ 0 & \text{otherwise} \end{cases}$$

This yields the linear form: $F_k(x) = I^{(k)} \cdot (A^{(k)}x + B^{(k)})$.

The boundaries between linear regions correspond to discontinuities in the gradient of the network's output. The use of these gradient discontinuities, also known as critical points to reverse-engineer neural network parameters is rooted in classical cryptanalytic techniques, where structured input-output differences reveal internal system components [20].

To identify critical points, we use the 2-linearity algorithm, which locates them for arbitrary neurons without revealing their neuron or layer of origin [18].

3. Our Method

Modern CNN architectures are composed of a diverse set of layer types, including linear layers (fully connected, convolutional, and residual blocks), pooling layers (min, max, and average pooling), normalization layers (e.g., batch normalization), regularization layers (dropout) and activation functions such as ReLU (Rectified Linear Unit). Prominent architectures within this scope include AlexNet, LeNet-5, ResNet, and VGG.

In this section, we establish the theoretical framework underpinning **CopyCNN**, detailing how it enables the weight extraction from CNN architectures. We systematically address the unique challenges posed by each of the possible layer types and present a tailored strategy to overcome them.

Our method extracts the weights layer by layer, following the sequential structure of the target model. Each layer-specific strategy serves as a building block of the overall attack and is applied in the sequential order, according to the architecture of the targeted model. To achieve this,

we classify the layers into two classes and outline the approach for each:

- For **Linear layers:**
  - Develop a method to extract the layer's weights.
- For **Non-linear layers:**
  - Explain how the overall piecewise linearity characteristic of the neural network is preserved despite the non-linear nature of the layer. This property is essential to maintain the existing extraction methods for fully connected layers [18].
  - Propose strategies to overcome the challenges posed by the layer while maintaining the extraction methods of the linear layers.

3.1. Convolutional Layers

Convolutional layers can be considered as a specialized form of fully connected layers. The convolution operation is a linear transformation, allowing it to be expressed as a matrix multiplication, similar to fully connected layers [21]. This reformulation enables the direct application of extraction techniques developed for fully connected layers [18], without requiring changes to the core algorithm.

However, matrices representing convolutional operations are inherently sparse and repetitive, as the same convolutional kernels are applied across overlapping input windows. We exploit these characteristics and propose an enhancement to the existing method.

The convolution weights can be represented as a 4D tensor:

$$C \in \mathbb{R}^{C_{\text{out}} \times C_{\text{in}} \times K_h \times K_w}$$

where C_{out} and C_{in} correspond to the output and input channels, respectively K_h and K_w represent the height and width of the convolution kernel.

Given a critical point x^* to a neuron η, where $x^* \in W(\eta)$ the neuron lies in one of the output channels, we'll denote this channel as c_o. Suppose we are extracting the weights of k-th convolutional layer, having already extracted all preceding layers. The value corresponding to neuron η is the result of applying the convolution kernel to the previous layer's output, h_{k-1} at an offset (x, y) over the input. The relationship can be expressed as:

$$\sum_{c_i=1}^{C_{\text{in}}} \sum_{i=1}^{K_h} \sum_{j=1}^{K_w} C^k_{c_o,c_i,i,j} \cdot h_{k-1_{c_i,(x+i),(y+j)}} + B^{(k)}_{c_o} = 0$$

For each convolutional layer, there are C_{out} sets of convolution kernels to extract, with each critical point enabling the extraction of one convolution kernel set. Each set contains C_{in} kernels. This extraction process is analogous to the row-wise weight extraction of fully connected layers. We begin the extraction by sampling $N = C_{in} \cdot K_h \cdot K_w + 1$ directions around the critical point.

Next, we "slice" the input window that is covered by the convolution kernel and corresponds to neuron η's value. We flatten the results into vectors, resulting in the following vectors:

$$\{v_j\} = \Big\{ \Big(\mathbf{1}\Big(h^{(k-1)}_{x^*} > 0\Big) \cdot A^{(k-1)} \cdot ... \\ ... \cdot \mathbf{1}\Big(h^{(1)}_{x^*} > 0\Big) \cdot A^{(1)} \cdot \delta_j \Big)_{0:c_{in},x:x+k_h,y:y+k_w} \Big\}_{j=1}^{N+1}$$

Where δ_j represents the perturbation applied in the j-th direction and $\mathbf{1}$ represents the indicator function. To retrieve the weights, we proceed with finite differences attack [18] to solve a weight vector w such that $y_j = v_j \cdot w$ for all j, up to a global sign.

Since the neuron corresponding to a given critical point is not known a priori, we perform this calculation for every possible convolution window within the layer's input and then correlate the results using the weight correlation method described in Section 3-F. Notably, once the second gradient for each direction is computed, further queries to the original model are unnecessary. As so, the remaining calculations, including the window slicing and weight correlation, can be performed offline for each critical point (i.e., without additional queries to the target model).

For simplicity, convolutional layers can be treated as matrix multiplications, however in practice their extraction is made using this tailored method.

3.2. Residual Blocks

Residual blocks are widely used in modern neural networks, particularly in CNN architectures such as ResNet and its variants. Their ability to improve training stability and performance has made them a standard component in many state-of-the-art models for tasks such as image classification, object detection, and natural language processing [22].

Residual connections maintain the linearity of the neural network. In their simplest form, they add the output of a preceding layer to a subsequent layer. Other variations operate another piecewise linear function and add its output to a given layer output. Typically, a ReLU activation function is added after adding both results. The two computational paths in a residual block are commonly referred to as the *main path* (which undergoes a series of transformations) and the *shortcut path* or *skip connection* (which provides a direct bypass, often with fewer transformations).

For example, consider a residual block spanning from the k'th layer to the $(k+L)$'th layer of a network. We denote this residual block as $F_{k...k+L}$ and define the total number of layers in the shortcut path as Z. The residual block can be mathematically expressed as follows:

$$\begin{aligned} F_{k...k+L}(x) &= \sigma \circ \left(F^{(Main)}(x) + F^{(Shortcut)}(x) \right) \\ &= \sigma \circ \Big(F_L^{(Main)} \circ ... \circ F_2^{(Main)} \circ F_1^{(Main)}(x) \\ &+ F_Z^{(Shortcut)} \circ ... \circ F_2^{(Shortcut)} \circ F_1^{(Shortcut)}(x) \Big) \\ &= \sigma \circ \Big(f_L^{(Main)} \circ F_{L-1...1}^{(Main)}(x) \\ &\quad + f_Z^{(Shortcut)} \circ F_{Z-1...1}^{(Shortcut)}(x) \Big) \end{aligned}$$

where $F^{(Main)}$ denotes the main computational path within the residual block and $F^{(Shortcut)}$ represents the shortcut path.

Unlike standard layers, where ReLU is applied to each individual layer's output, residual blocks apply ReLU only after summing the outputs of $f_L^{(Main)}$ and $f_Z^{(Shortcut)}$. This structural difference suggests a modified approach for extracting weights in the presence of residual blocks. Additionally, since a residual block consists of two separate computational paths, an important consideration is the order in which layers should be extracted.

1) Layer Extraction Order: Except for the final layers in the main and shortcut paths, the extraction of layers in each path can be performed independently. Within each path, layers must be extracted iteratively in the same sequential order as in a neural network without residual blocks. Once all layers in both paths (except for the final layers) have been extracted, we proceed to extract the last layers of the residual block, which require special treatment.

2) Last Layers' Extraction: Since a ReLU layer is applied after summing the outputs of $f_L^{(Main)}$ and $f_Z^{(Shortcut)}$, extracting their weights requires a modified approach. Specifically, given a critical point to neuron in that layer, weights from the main path and shortcut path must be considered simultaneously when applying the finite differences attack. To achieve this, we perform the following steps:

1) **Random Direction Sampling:** Sample a number of random directions equal to the total number of weights in both operators plus one. We denote it as (N).
2) **Input Segmentation:** If one or more of the operators is a convolutional layer, select the input window according to the kernel applied.
3) **Equation Construction:** Construct equations for the finite differences attack method based on the concatenated inputs for both operators.

Given the layer types and the two distinct computational paths that an input may traverse, we construct two sets of vectors: $\{v_j^{(Main)}\}$ for the main path and $\{v_j^{(Shortcut)}\}$ for the shortcut path. We then apply the finite differences attack to the concatenated vector set $\{v_j\}$, where:

$$\{v_j\} = \{v_j^{(Main)} | v_j^{(Shortcut)}\}$$

This process enables the parallel extraction of the weights from the final layers of the residual block: $f_L^{(Main)}$ and $f_Z^{(Shortcut)}$. Residual blocks introduce additional challenges in the weight extraction process. These challenges, along with solutions, are further discussed in Section 3-G.

3.3. Batch Normalization

Batch normalization is a technique used to stabilize and accelerate the training of deep neural networks by normalizing the inputs to each layer across mini-batches. It reduces internal covariate shift, which helps the network converge faster and become more resilient to different initializations. The data is normalized, shifted and scaled using learnable parameters μ, σ_{var}, γ and β, allowing the network to maintain representational capacity. During inference these parameters remain constant for each feature.

Given an input vector x, the batch normalization transformation produces an output $\hat{x}$, where each element $\hat{x}$ is given by:

$$\begin{aligned} \hat{x_i} = BN_i(x) &= \frac{x_i - \mu_i}{\sqrt{{\sigma_{var}}_i^2 + \epsilon}} \cdot \gamma_i + \beta_i \\ &= \frac{\gamma_i}{\sqrt{{\sigma_{var}}_i^2 + \epsilon}} \cdot x_i + \left(\frac{-\mu_i \cdot \gamma_i}{\sqrt{{\sigma_{var}}_i^2 + \epsilon}} + \beta_i \right) \end{aligned}$$

This transformation can be considered as fully connected layer, where the weight matrix A_{BN} is a diagonal matrix with $\frac{\gamma_i}{\sqrt{{\sigma_{var}}_i^2+\epsilon}}$ at the i-th diagonal element, and the bias vector B_{BN} with $\frac{-\mu_i \cdot \gamma_i}{\sqrt{{\sigma_{var}}_i^2+\epsilon}} + \beta_i$ at the i-th index.

In practice, batch normalization is typically applied after a linear layer and before the ReLU activation function. This allows us to treat the batch normalization and the linear layer as a single, functionally equivalent linear layer f':

$$\begin{aligned} f' &= A_{BN}(A \cdot x + B) + B_{BN} = \\ &\quad A_{BN}A \cdot x + A_{BN}B + B_{BN} = A' \cdot x + B' \end{aligned}$$

The equivalent weight matrix A' is given by $A_{BN}A$, which is the weight matrix A scaled by the diagonal elements of A_{BN}. Similarly, the equivalent bias vector B' can be computed as $B' = A_{BN}B + B_{BN}$.

This equivalence implies that batch normalization can be seamlessly incorporated

into the preceding linear layer during the extraction process. Consequently, the presence of a batch normalization layer has no meaningful impact on the extracted network. In fact, batch normalization layers can effectively be ignored during extraction, as the weights retrieved through the finite differences attack are already scaled.

3.4. Pooling Layers

Pooling layers are fundamental components in modern convolutional neural networks (CNNs), primarily used to reduce spatial dimensions. This downsampling lowers computational cost, reduces model complexity, and improves robustness against overfitting. Two common variants are max pooling and average pooling which we will discuss.

1) Average Pooling: Average pooling computes the mean over an $s \times s$ window in the input feature map. It can be expressed as a convolution with uniform weights of $\frac{1}{s^2}$ and zero bias. As a result, it is inherently linear and preserves the network's piecewise linearity.

In practice, average pooling is typically applied after ReLU and before the next linear layer. To accommodate this during extraction, we apply the same average pooling operation to the output of the preceding ReLU layer. The resulting feature maps are then used to construct the vectors $\{v_j\}$ for weight recovery procedure.

2) Min/Max Pooling: Min and max pooling introduce localized non-linearities, as the active neuron within a pooling window may change depending on the input. While these operations preserve the overall piecewise linearity of the network, they alter linear region boundaries.

Mathematically, this operation can be represented as a matrix multiplication. The 3D input tensor h'_k is flattened and multiplied by a matrix M defined as:

$$M = \begin{cases} 1 & \text{if element is the min/max} \\ & \text{within the window,} \\ 0 & \text{otherwise} \end{cases}$$

As long as the min/max indices remain fixed, M is constant, and the operation behaves linearly.

We note that during the critical point search, transitions in the active neuron within a pooling window may be misidentified as critical points. These false positives are illustrated in Fig. 2 and are later filtered during the weight correlation and unification process.

3.5. Dropout

Dropout is a common regularization technique in convolutional neural networks, where a fraction of neurons is randomly deactivated during training to prevent overfitting [23]. During inference, dropout is disabled and has no effect on the network's structure or learned weights. As a result, CopyCNN does not need to account for dropout layers during extraction.

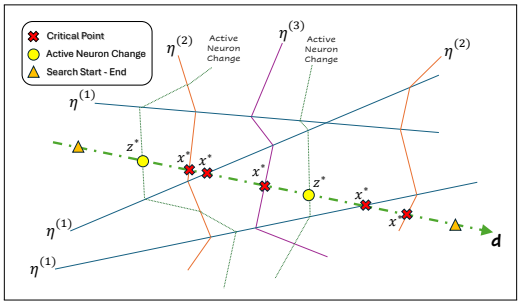


Fig. 2. This figure illustrates a critical point search process in a neural network containing a Max/Min pooling layer. Points labeled x^* are true critical points identified during the search, while Z^* marks boundaries where the active neuron changes - falsely identified as critical points. The notation $\eta^{(i)}$ denotes neurons in the i-th layer.

3.6. Weights Correlation and Unification

After extracting candidate weights from critical points, we must identify those that correspond to the current layer. This is complicated by false positives—especially in the presence of min/max pooling layers which may yield weights from unrelated neurons or layers.

To address this, we extend prior correlation and unification techniques [17], [18], adapting them for the challenges posed by CNNs. We define a correlation function f_{criteria} that identifies weight vectors as scaled versions of the same true weight (i.e., $w_1 = c \cdot w_2$) and use it to construct a graph where nodes represent extracted weights and edges indicate correlation.

Clusters (cliques) in this graph represent sets of consistent results. We unify each cluster by scaling to a common baseline and aggregating results (e.g., via element-wise median) to reconstruct full rows or kernels. Spurious weights typically fail to form large, consistent clusters and are naturally filtered out in this process. This combined correlation and unification step ensures robust recovery of the target layer's weights up to a global sign.

3.7. Global Sign Recovery

Recovering the global signs of elements in deeper layers poses challenges, especially in the presence of convolutions and residual blocks. This task involves determining the global sign of each element in a linear layer, i.e., each row in a fully connected layer or output channel in a convolutional layer. Residual blocks introduce further complexity. If both paths in the residual block start with the same type of layer and characteristics (e.g., fully connected or convolutional with the same kernel size), clustering may yield elements from both paths.

This makes it difficult to assign the correct global sign to each element and determine its origin path.

1) Brute-Force Approach: A straightforward solution is to brute-force all possible sign combinations and path assignments (in case a residual block is present). For each combination, if subsequent layer is extraction succeeds, it indicates the correct signs and assignments were found.

Formally, consider a residual block starting at the z-th layer, whose weights have been extracted up to their global signs. Let $\{x_j^*\}$ be critical points corresponding to a neuron in one of the $z+1$ layers, and let $h_{x_j^*}^{(z)}$ denote the output from the z-th layers. There exists a vector $s \in \{-1, 0, 1\}^N$, a weight vector w, and a bias b belonging to one of the elements in the $z+1$ layer such that:

$$s \cdot h_{x_j^*}^{(z)} \cdot w + b = 0, \quad \text{for all } j.$$

$s_i = 0$ indicates that the i-th element does not belong to the layer we are targeting, while $s_i \in \{-1, 1\}$ indicates the elements' global sign.

This brute-force method avoids additional queries, but is computationally expensive. With N elements, the number of combinations is 3^N, which becomes infeasible for large layers. For non-residual layers, where all elements originate in a single source path, the search space reduces to 2^N.

2) Mixed-Integer Linear Programming (MILP): To overcome brute-force complexity, we use mixed-integer linear programming (MILP), which efficiently handles both discrete (sign) and continuous (weight) variables [24]. Prior work [18] showed that critical points for a neuron in the $z+1$ layer can be retrieved using a linear number of queries. Given such a set $\{x_j^*\}$, we solve for s and w with the bias fixed to -1:

$$s \cdot h_{x_j^*}^{(z)} \cdot w = 1, \quad \text{for all } j.$$

While MILP has worst-case exponential complexity, modern solvers perform well in practice, making it a practical and efficient solution for global sign recovery and element selection.

4. Architecture Extraction

So far, we have described how to extract a network's weights when its architecture is known. In practice, most models are deployed as black boxes, making prior knowledge of the type and order of layers unrealistic.

CopyCNN addresses this challenge without requiring additional queries. As long as the network consists of layer supported by our method, the architecture can be reconstructed by adjusting the weight extraction process. The successful extraction of a layer's weights inherently validates the guessed type and structure of that layer, providing a guiding signal for reconstructing the architecture sequentially

4.1. Key Challenges and Brute-Force Strategy

During the extraction process, there are two steps that require the knowledge of the type, number and shape of the extracted layer:

1) **Weight Computation:** For convolutional layers, the kernel size must be known in order to apply the correct window to the layer's input and compute the weights. Without this information, the weight extraction process cannot proceed correctly.
2) **Clustering and Unification:** While clustering the weights graph to retrieve the extracted weights, it is essential to know the number of rows (for fully connected layers) or the number of output channels (for convolutional layers). This determines the number of clusters taken to the weights reconstruction.

Since this information is unknown, we adopt a brute-force approach. The number of parameters per element, denoted $n_{\text{parameters}}$, can be iteratively guessed. Begin with sufficiently small $n_{\text{parameters}}$ and perform the following:

1) Assume it is a convolutional layer and perform the weight extraction procedure given $n_{\text{parameters}}$.
2) Inspect the clustering results. If large, coherent clusters form, the guess is likely to be correct (Section 4-C). If not, increment $n_{\text{parameters}}$ and retry.
3) If $n_{\text{parameters}}$ reaches the input size, the layer is assumed to be fully connected.

As the second-gradient computations are independent of $n_{\text{parameters}}$, previously computed values can be reused, making the search computationally cheaper and query-efficient.

4.2. Handling Pooling Layers

Pooling layers (average, max, or min) and their window sizes are also unknown. We extend the brute-force process to test common configurations—e.g., 2×2 or 3×3—with each pooling type. The small configuration space keeps this process tractable.

4.3. Inferring Output Size via Clustering

The number of elements in every layer - rows for fully connected layers or output channels for convolutional layers, can be inferred using clustering. As unrelated weight vectors rarely form large clusters, the number of significant clusters provides a strong signal about the true output size. Thus, clustering not only aids in weight recovery but also reveals structural information about the targeted layer.

5. Evaluation

Our attack aims to extract the model's weights as accurately as possible while minimizing the number of queries to the target network. This is critical because each query may incur costs—monetary, computational, or stealth-related (e.g., avoiding detection during model extraction). These constraints guided us in the design and implementation of CopyCNN.

5.1. Evaluation Methods

To ensure comparability with prior work, we adopted standard evaluation metrics from recent research [17], [18]: **functional equivalence** and **weight equivalence**. These metrics provide a comprehensive assessment of the attack's success.

1) Functional Equivalence: Functional equivalence measures how closely an extracted model g replicates the behavior of the target model f. Formally, f and g are (ϵ, δ)-functionally equivalent over the input space S if:

$$\Pr_{x \in S}[|f(x) - g(x)| \leq \epsilon] \geq 1 - \delta.$$

2) Weight Equivalence: Weight equivalence measures the direct similarity between the extracted weights θ and the target model weights $\hat{\theta}$. This is quantified as $\max |\theta - \hat{\theta}|$. Since CopyCNN extracts an unordered and scaled version of the weights, we first align the extracted weights by reordering (e.g., rows, columns, or output channels) and scaling. After alignment, weight equivalence is computed to assess extraction accuracy.

5.2. Finite Precision

In theory, CopyCNN should enable the extraction of the exact model's weights. However, practical limitations arise due to the finite precision of computational systems. Modern machines typically use floating-point arithmetic which introduces rounding errors and limited precision when performing operations on very small or very large numbers [25]. These numerical errors accumulate across layers, leading to slightly reduced precision in the extracted weights of deeper layers as depicted in Fig. 3.

The choice of data type used for mathematical calculations also plays a significant role. Higher-precision data types yield better extraction results by minimizing numerical errors. In our implementation, we used PyTorch's `float64` data type to maximize precision. While lower-precision options are available, they lead to reduced accuracy and may necessitate additional queries to achieve sufficient precision to enable the extraction for deeper layers.

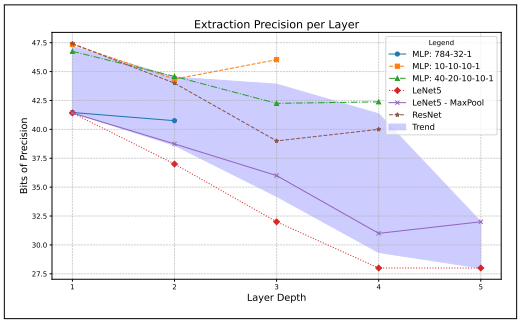


Fig. 3. This figure illustrates the precision (in bits) achieved in each extracted layer across the architectures under consideration. The graph reveals a trend where deeper layers generally exhibit reduced precision compared to earlier layers.

5.3. Results

The results of our experiments are presented in Table 1. For MLP architectures, although CopyCNN extraction methods are similar to the prior method [18], for many of the extractions we achieve results that are significantly superior in terms of the extraction precision, functional equivalence and total queries to the target model. We relate this to various optimizations that we have made through CopyCNN implementation.

The most notable achievement of CopyCNN is its ability to extract weights from architectures that were previously non extractable. In our experiments, we successfully extracted two variants of the LeNet-5 architecture: one with intermediate max-pooling layers and one without. This demonstrates CopyCNN's capability to handle networks with convolutional layers, batch normalization, and pooling layers. Furthermore, we tested CopyCNN on a ResNet block, the residual block foundational in most ResNet variants - confirming its ability to extract networks with residual blocks.

Although these results offer strong evidence for the correctness and applicability of CopyCNN in CNN weight extraction, they are demonstrated on relatively simple architectures. Scaling the attack to deeper or more complex networks presents several implementation challenges. In practice, a successful extraction often requires careful fine-tuning and adjustment of multiple sensitive thresholds, making full automation difficult for large models. Additionally, as illustrated in Fig. 2, the phenomenon of critical hyperplane bending, where critical hyperplanes from deeper layers are distorted by those of preceding layers, making it harder to find certain critical points, especially in larger models.

6. Discussion

6.1. Limitations

Our work has several limitations. First, our evaluation focuses on convolutional neural

Table 1
COMPARATIVE EVALUATION OF COPYCNN AND PRIOR METHODS UNDER A COMMON SETTING WHERE THE TARGET MODEL ARCHITECTURE IS KNOWN, LOGITS ARE OBSERVABLE, AND ARBITRARY INPUTS CAN BE QUERIED.

Attack	**Queries**	$(\epsilon, 10^{-9})$	$(\epsilon, 0)$	$\max \lvert\theta - \hat{\theta}\rvert$
MLP: 784-32-1				Params: 25,120
CopyCNN	2^{20}	$\mathbf{2^{-37.2}}$	$\mathbf{2^{-32}}$	$\mathbf{2^{-40.7}}$
CJM [18]	$2^{19.2}$	$2^{-28.8}$	$2^{-27.4}$	$2^{-30.2}$
JCB [17]	$\mathbf{2^{18.2}}$	$2^{3.2}$	$2^{4.5}$	$2^{-1.7}$
MLP: 10-10-10-1				Params: 420
CopyCNN	$\mathbf{2^{14.7}}$	$\mathbf{2^{-44.7}}$	$\mathbf{2^{-40.5}}$	$\mathbf{2^{-44.3}}$
CJM [18]	2^{16}	$2^{-42.7}$	$2^{-37.98}$	2^{-36}
RK [16]	2^{22}	$2^{-10.3}$	$2^{-3.4}$	2^{-12}
MLP: 40-20-10-10-1				Params: 1,110
CopyCNN	$\mathbf{2^{17.1}}$	$\mathbf{2^{-41.5}}$	$\mathbf{2^{-36.2}}$	$\mathbf{2^{-42.2}}$
CJM [18]	$2^{17.8}$	$2^{-31.7}$	$2^{-23.4}$	$2^{-27.1}$
LeNet5				Params: 1,117,770
CopyCNN	$\mathbf{2^{34}}$	$\mathbf{2^{-27}}$	$\mathbf{2^{-23}}$	$\mathbf{2^{-28}}$
LeNet5 w/ MaxPool				Params: 59,850
CopyCNN	$\mathbf{2^{31.7}}$	$\mathbf{2^{-29}}$	$\mathbf{2^{-26}}$	$\mathbf{2^{-31}}$
ResNet Block				Params: 3,096
CopyCNN	$\mathbf{2^{29.2}}$	$\mathbf{2^{-38}}$	$\mathbf{2^{-35}}$	$\mathbf{2^{-39}}$

networks that are relatively small compared to contemporary deep vision architectures, also, it does not address the modern and popular large-scale models. While the extraction of large convolutional networks is possible in principle, in practice we observed that finite numerical precision and accumulated reconstruction errors make it difficult to fully recover deep architectures such as VGG-style networks. This points to an important direction for future work - improving extraction robustness through better numerical methods and higher-precision computation. In this work, we therefore focused on smaller networks in order to validate the correctness and feasibility of our proposed method.

Second, our attack assumes access to pre-softmax outputs. This setting provides richer information than the hard-label regime, in which only the final predicted class is observable. Recent work has shown that multilayer perceptrons can be extracted even in the hard-label setting [19]. Extending such techniques to convolutional architectures remains an open problem and a promising direction for future research.

Finally, our attack requires a large number of adaptive queries and assumes that the adversary is not subject to strict query limits, rate limiting, or monitoring. In realistic deployment settings where such controls are typically enforced, the practicality of the attack may be reduced or may require significantly more time and resources.

6.2. Defenses

Various mitigations exist to defend neural networks against model extraction attacks such as CopyCNN. While there is no silver bullet, a layered defense can substantially increase the cost and complexity of extraction. In deployed systems, neural networks should be treated as sensitive assets and protected accordingly.

1) General Best Practices:

- **Anomalous Query Detection:** Model extraction typically requires a large number of carefully structured queries. Monitoring query volume and distribution can reveal suspicious behavior. Indicators include unusually high query rates, repeated access from a single client, and inputs sampled from a typical or adversarial distributions. Practical mitigations include rate limiting, authentication with quotas, and behavioral fingerprinting [10].
- **API Output Hardening:** Public APIs should minimize the information revealed by model outputs. Returning only predicted class labels instead of logits or confidence scores significantly reduces the information available to the attacker and hinders substitute model training and gradient estimation.

2) CopyCNN-Specific Defenses:

1) **Input Randomization:** Adding controlled noise or random perturbations to inputs can reduce the attacker's ability to perform precise probing. However, such techniques must be carefully tuned to avoid degrading model correctness and may offer limited protection on their own.
2) **Network Swapping / Architectural Randomization:** Employing a moving-target defense, where multiple functionally equivalent models (e.g., with different architectures or weights) are randomly rotated at runtime, could increase the attacker costs significantly and may even

render extraction infeasible under realistic budgets.

3) **Label-Only Outputs:** Restricting outputs to hard labels (top-1 predictions) directly limits CopyCNN's ability to estimate gradients and recover internal parameters.

7. Conclusion

We have shown that precise, layer-wise extraction of convolutional neural networks is feasible in practice, and that both model architectures and parameters can be recovered with high fidelity under certain access assumptions. Beyond the specific techniques presented, we hope the methodology introduced in this work will serve as a useful foundation for future research on the extraction of larger, more complex, and different architectures of neural networks.

References

[1] W. Samek, G. Montavon, S. Lapuschkin, C. J. Anders, and K.-R. Müller, "Explaining deep neural networks and beyond: A review of methods and applications," *Proceedings of the IEEE*, vol. 109, no. 3, pp. 247–278, 2021.

[2] J. Zhang, Z. Gu, J. Jang, H. Wu, M. P. Stoecklin, H. Huang, and I. Molloy, "Protecting intellectual property of deep neural networks with watermarking," in *Proceedings of the 2018 on Asia conference on computer and communications security*, 2018, pp. 159–172.

[3] J. Ebrahimi, A. Rao, D. Lowd, and D. Dou, "Hotflip: White-box adversarial examples for text classification," *arXiv preprint arXiv:1712.06751*, 2017.

[4] C. Xiao, B. Li, J.-Y. Zhu, W. He, M. Liu, and D. Song, "Generating adversarial examples with adversarial networks," *arXiv preprint arXiv:1801.02610*, 2018.

[5] M. Fang, X. Cao, J. Jia, and N. Gong, "Local model poisoning attacks to {Byzantine-Robust} federated learning," in *29th USENIX security symposium (USENIX Security 20)*, 2020, pp. 1605–1622.

[6] N. Haim, G. Vardi, G. Yehudai, O. Shamir, and M. Irani, "Reconstructing training data from trained neural networks," *Advances in Neural Information Processing Systems*, vol. 35, pp. 22 911–22 924, 2022.

[7] N. Papernot, P. McDaniel, I. Goodfellow, S. Jha, Z. B. Celik, and A. Swami, "Practical black-box attacks against machine learning," in *Proceedings of the 2017 ACM on Asia conference on computer and communications security*, 2017, pp. 506–519.

[8] F. Tramèr, F. Zhang, A. Juels, M. K. Reiter, and T. Ristenpart, "Stealing machine learning models via prediction {APIs}," in *25th USENIX security symposium (USENIX Security 16)*, 2016, pp. 601–618.

[9] T. Orekondy, B. Schiele, and M. Fritz, "Knockoff nets: Stealing functionality of black-box models," in *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, 2019, pp. 4954–4963.

[10] M. Juuti, S. Szyller, S. Marchal, and N. Asokan, "Prada: protecting against dnn model stealing attacks," in *2019 IEEE European Symposium on Security and Privacy (EuroS&P)*. IEEE, 2019, pp. 512–527.

[11] L. Batina, S. Bhasin, D. Jap, and S. Picek, "Csi neural network: Using side-channels to recover your artificial neural network information," *arXiv preprint arXiv:1810.09076*, 2018.

[12] M. Yan, C. W. Fletcher, and J. Torrellas, "Cache telepathy: Leveraging shared resource attacks to learn {DNN} architectures," in *29th USENIX Security Symposium (USENIX Security 20)*, 2020, pp. 2003–2020.

[13] Y. Zhu, Y. Cheng, H. Zhou, and Y. Lu, "Hermes attack: Steal {DNN} models with lossless inference accuracy," in *30th USENIX Security Symposium (USENIX Security 21)*, 2021.

[14] B. Coqueret, M. Carbone, O. Sentieys, and G. Zaid, "A hard-label cryptanalytic extraction of non-fully connected deep neural networks using side-channel attacks," *arXiv preprint arXiv:2411.10174*, 2024.

[15] J. Breier, X. Hou, D. Jap, L. Ma, S. Bhasin, and Y. Liu, "Practical fault attack on deep neural networks," in *Proceedings of the 2018 ACM SIGSAC Conference on Computer and Communications Security*, 2018, pp. 2204–2206.

[16] D. Rolnick and K. Kording, "Reverse-engineering deep relu networks," in *International conference on machine learning*. PMLR, 2020, pp. 8178–8187.

[17] M. Jagielski, N. Carlini, D. Berthelot, A. Kurakin, and N. Papernot, "High accuracy and high fidelity extraction of neural networks," in *29th USENIX security symposium (USENIX Security 20)*, 2020, pp. 1345–1362.

[18] N. Carlini, M. Jagielski, and I. Mironov, "Cryptanalytic extraction of neural network models," in *Annual international cryptology conference*. Springer, 2020, pp. 189–218.

[19] N. Carlini, J. Chávez-Saab, A. Hambitzer, F. Rodríguez-Henríquez, and A. Shamir, "Polynomial time cryptanalytic extraction of deep neural networks in the hard-label setting," in *Annual International Conference on the Theory and Applications of Cryptographic Techniques*. Springer, 2025, pp. 364–396.

[20] E. Biham and A. Shamir, *Differential Cryptanalysis of the Data Encryption Standard*. Springer, 1993.

[21] A. Vasudevan, A. Anderson, and D. Gregg, "Parallel multi channel convolution using general matrix multiplication," in *2017 IEEE 28th international conference on application-specific systems, architectures and processors (ASAP)*. IEEE, 2017, pp. 19–24.

[22] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2016, pp. 770–778.

[23] G. Ghiasi, T.-Y. Lin, and Q. V. Le, "Dropblock: A regularization method for convolutional networks," *Advances in neural information processing systems*, vol. 31, 2018.

[24] A. H. Land and A. G. Doig, *An automatic method for solving discrete programming problems*. Springer, 2010.

[25] D. Goldberg, "What every computer scientist should know about floating-point arithmetic," *ACM computing surveys (CSUR)*, vol. 23, no. 1, pp. 5–48, 1991.

Yehonatan Lusky is a Security Architect at NVIDIA and holds an M.Sc. in Computer Science from the Technion - Israel Institute of Technology. His research interests focus on system security, AI security, and the intersection between the two.

Avi Mendelson is a professor at the CS department at Technion, head of the ATLAS (AI Trust, Learning, Architecture, and Security) Lab, and an IEEE Fellow. He has experience in both industrial and academic settings, spanning security and safety of LLM-based systems, computer architecture, hardware security, and machine learning architectures.

Among his industrial jobs, he worked for 11 years as a senior researcher and principal engineer at Intel, and served as a manager of the Academic-Relation of Microsoft R&D Israel. His current academic work focuses on Hardware Security and the security of LLM systems in general and agentic systems in particular.

Professor Avi Mendelson was a member of the ACM-Europe Council (2009-2014) and the IEEE Computer Society Board of Governors (2017-2019 as a member and second VP). He served as a member and chair of the Eckert-Mauchly Award committee. He also served as an associate editor for various journals, including IEEE CAL, IEEE Transactions on Computers, and IEEE Transactions on Emerging Topics in Computing

Yaniv Nemcovsky is an Associate Research Scientist in Enterprise AI at the Infosys Topaz Columbia University Enterprise AI Center, working with Professor Junfeng Yang. His research develops foundations for trustworthy AI by establishing robust performance evaluation and adaptive routing as core mechanisms for system reliability and safety across CV, NLP, RAG, and multimodal architectures. His work addresses physically realizable adversarial attacks on vision-based autonomous systems, jailbreak mechanisms and bias concealment in large language models, and success-based semantic representations for per-query performance prediction and model selection. His research on physical adversarial attacks received the Most Promising Paper Award at the ECCV 2022 AROW Workshop as one of the first credible demonstrations of real-world adversarial threats.
Prior to joining Columbia, he was a Postdoctoral Researcher at Ben-Gurion University of the Negev with Professor Chaim Baskin, where he led a cooperative academia–industry research group on physical adversarial threats, and a Research Intern at IBM Research, where he studied industrial-scale multimodal retrieval systems. He received his PhD in Computer Science from the Technion in 2025, advised by Professors Alex Bronstein and Avi Mendelson, and during his doctoral studies served as an Adjunct Senior Lecturer at the Technion, where he designed and instructed a graduate-level course on adversarial ML. He also holds an MSc in Computer Science and a double BSc in Computer Science and Physics from the Technion. He has published in top-tier venues and co-authored successful national grant proposals, serving as assistant investigator on the resulting projects and directing collaborative teams of researchers and engineers.

The MKA Law and the Milutinović–Katulevskiy Architecture: A Post-Moore Roadmap for AI-Scale, Photonic, and Quantum-Adjacent Computing

Katulevskiy, Daniil; Katulevskaya, Elizaveta;
Katulevskiy, Aleksey; and Milutinović, Veljko

Abstract: ***Progress beyond conventional transistor scaling increasingly depends on whether gains in devices, memory, interconnects, compilers, and system software survive integration into complete workloads. The Milutinović–Katulevskiy Architecture (MKA) develops a common research framework for this problem. Building on Veljko Milutinović's Honeycomb architecture and graph-mapping lineage, MKA treats computation, stored state, communication, placement, and physical substrate as one architectural design space. The article formulates a broader system-level principle: a local hardware improvement counts as genuine architectural progress only when its advantage survives data movement, state placement, interfaces and conversion, scheduling and queueing, accuracy, reliability, and service-level constraints at the boundary of completed useful work. The proposed MKA Law operationalizes this principle through the energy–time product of completed workloads under fixed quality and service requirements.***

Communication bounds, placement models, uncertainty-aware optimization, and substrate-specific break-even conditions determine when locality, reuse, heterogeneous execution, and changes of physical medium can improve the whole system. An exact reference planner makes one part executable. Across a synthetic planning study, coordinated placement improves feasible plans relative to local and fixed-placement strategies; strong global alternatives recover most of that benefit. Full accounting for movement and setup prevents decisions that look attractive locally but fail system constraints. These results establish a starting point for calibrated hardware, concurrent graph workloads, and experimental implementations.

MKA follows the quantitative architecture tradition developed by John L. Hennessy and David A. Patterson, where workload, compiler, machine, memory, communication, and implementation technology are evaluated together. Oskar Mencer's dataflow work supplies an implementation precedent for compiler-directed spatial execution. Mor Harchol-Balter's queueing and server-power research connects execution capacity to delivered service. Dimitris Bertsimas's robust and prescriptive optimization provides the uncertainty-aware decision layer. Daniela Rus's graph grammars, decentralized control, and physical co-design motivate search over realizable, controllable structures.

MKA organizes these ideas into six regimes spanning reconfigurable digital fabrics, analog computation, processing near memory, photonic interconnects, quantum-dot systems, and entanglement-assisted architectures for quantum-native workloads. Artificial intelligence infrastructure provides an immediate high-pressure application, while scientific simulation, finance, genomics, telecommunications, and other graph-intensive domains broaden the research space.

At datacenter scale, sustained system-level efficiency improvements could translate into major reductions in energy use and operating cost; in science, the same gains could enlarge the computations achievable within finite power and time budgets. MKA therefore proposes more than an individual accelerator design: it establishes a shared architectural language and an experimentally falsifiable research direction intended to carry useful design knowledge across successive generations of computing technology.

Prospectively, the MKA Law can be applied to open-weight LLM serving as a six-vector optimization over compute and precision, resident weights, KV and request state, expert and interconnect routing, batching and queueing, and active capacity with its power and cooling burden. Llama 4 Maverick, GLM-5.3, Kimi K3, and Qwen3.8 expose different resource demands, motivating model-specific operating-point searches. At hyperscale, the economic leverage is substantial: under a 5-GW average AI-serving load, PUE 1.20, and electricity at $0.07/kWh, every 10% reduction in system-level IT energy per completed request corresponds to approximately $368 million in annual electricity savings at fixed demand and unchanged PUE. A 30% reduction would exceed $1.1 billion annually. These figures are sensitivity thresholds, not assumed MKA performance, and illustrate the scale at which experimentally measured system-level gains would translate into economic impact.

Index Terms: ***Computer architecture, dataflow, energy–delay product, graph placement, heterogeneous computing, large language models, queueing, robust optimization.***

1. Introduction

SELECTING an efficient kernel can leave an inefficient application. Each change of execution site moves an intermediate value, may require a conversion, and can activate resources whose setup must be repaid. In the reference study below,

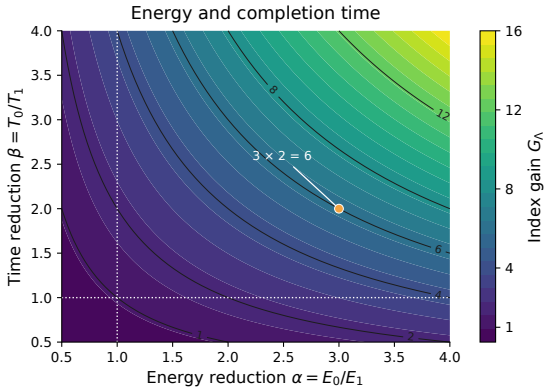


Fig. 1. Index gain $G_\Lambda = \alpha\beta$. Dotted lines mark unchanged energy and time; the upper-right region improves both. The marked example gives sixfold gain. MKA impact: simultaneous energy and latency improvements become visible in one completed-work metric.

a planner that omits transfers chooses deadline-violating plans in 189 of 540 cases. The same operations and available sites admit a feasible plan in every case. The difference is the information used to make the placement decision. Workload-based evaluation and computing-energy analysis motivate charging these costs to the complete application [1], [2].

MKA proposes a common description of operations, stored state, physical connections, and service requirements. Honeycomb research supplies the local-organization hypothesis. The research program asks where that organization improves a completed workload and how the decision changes when a computation moves to another substrate. AI serving is an immediate application, but scientific simulation, graph analytics, communication receivers, and numerical finance present related costs.

The proposed *MKA Law* compares energy and normalized completion time at fixed quality. It uses an inverse energy–delay product, with the individual quantities retained. Moore's integrated-circuit density trend supplied an earlier device-level target [3]; here the engineering target is completed useful work. The underlying multiplicative identity is algebraic. The substantive questions concern the mappings and operating conditions that improve it.

We contribute an executable planning subproblem, an exactness argument, and a complete computational comparison. The planner tracks intermediate ownership and configured sites, making transfer and setup costs part of the choice. The study includes exact single-objective alternatives and cases where joint planning ties them. Its synthetic inputs isolate the decision mechanism; component calibration and general graph mapping remain separate experimental tasks.

The decoder example shows why these decisions matter at scale. At 32K context, each active sequence needs 10 GiB of KV state. Raising batch size from 16 to 128 lowers modeled reads per token by about 39% while increasing resident block weights and KV state almost fivefold. FlashAttention and PagedAttention demonstrate the practical value of optimizing movement and allocation [4], [5]. The MKA program connects such decisions to substrate interfaces, uncertainty, and the service experienced by users.

Six regimes cover spatial digital execution, analog operations, processing near memory, photonic links, quantum dots, and entanglement-assisted systems. They can coexist, and their experiments have different maturity requirements. We recommend validating a digital mapping on existing hardware first, then testing whichever interface or communication cost remains limiting. The broader program is organized around reusable measurements and operating regions.

2. Architectural Foundations

2.1. Honeycomb Placement and the Cost of Regularity

Honeycomb research maps interconnection structures and computational graphs onto regular configurable cells [6], [7]. Hoch's Purdue study documents an early prototype under Veljko Milutinović's supervision [8]. V. Milutinović, D. Milutinović, and Furht subsequently developed the mapping patterns and their spatial and timing costs [9].

For one family in that analysis, the required cell count is

$$N_{\text{cells}} = N_{\text{pe}} \left(\frac{N_{\text{pe}}}{2} + 1 + M_c \right), \tag{1}$$

where $N_{\text{pe}} = 2^\ell$ and M_c counts mapped memory cells per processing element. Occupancy includes routing and storage, so it differs from useful processor activity.

Figure 2 shows regularity's trade-off: fewer global hops require more connectivity. Link length and bandwidth then determine physical latency.

Each Honeycomb-inspired implementation must specify its adjacency, operations, and buffers. Hexagonal geometry alone is ambiguous: tiling vertices have three neighbors, whereas cells joined across their sides have six.

2.2. Hennessy: Evaluate the Compiler and Machine Together

John L. Hennessy's work supplies the evaluation method. MIPS jointly designed compiler and processor [10]; his VLSI study related architecture to implementation constraints [11]. A graph representation likewise succeeds through the schedules its hardware can execute.

Hennessy and Jouppi studied technology-driven architectural choices [12]; Stanford FLASH extended co-design to memory and communication [13]. These costs remain part of the ma-

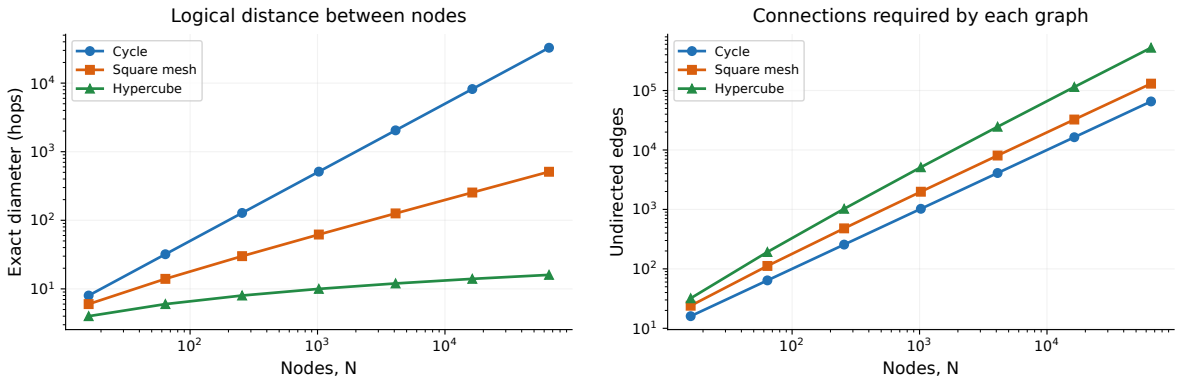


Fig. 2. Exact graph quantities for $N = q^2 = 4^k$. Cycle, square-mesh, and hypercube diameters (left) are $N/2$, $2(q-1)$, and $\log_2 N$; edge counts (right) are N, $2q(q-1)$, and $N \log_2 N/2$. MKA impact: shorter logical paths are valuable only when the added connectivity and physical cost also fit the workload budget.

chine when specialization relocates them. Hennessy and Patterson's post-Moore account places domain-specific design within this workload-based method [14].

2.3. Mencer: Make Dataflow Executable

Oskar Mencer's studies of extreme-performance dataflow and predictable reconfigurable systems connect application analysis with resource allocation and bottleneck prediction [15], [16]. Convolutional networks on dataflow engines provide an application example [17].

Buffers and synchronization govern scheduled dataflow. Static pipelines suit regular streams; irregular arrivals need a runtime. Harchol-Balter, Bertsimas, and Rus supply methods for service, uncertainty, and buildable structures.

2.4. Related Tools and the Contribution of the Framework

Timeloop explores DNN mappings and memory hierarchies [18]; MAESTRO analyzes reuse, occupancy, and dataflow cost [19]. Accelergy supplies hierarchical component energy estimates [20], while CiMLoop models interactions between representations, circuits, and compute-in-memory workloads [21]. Halide separates algorithm semantics from storage and execution schedules [22]. Fern further exposes production and consumption patterns to compose black-box subroutines with better locality [23].

The 2024 IRDS *Systems and Architectures* chapter already connects application requirements to heterogeneous computing, memory fabrics, electronic and photonic communication, and whole-stack energy accounting [24]. Its cloud, edge, and cyber-physical categories show that system-level evaluation is an established roadmap concern. MKA specializes that perspective through an explicit workload contract, ownership and interface accounting, and a reproducible planning subproblem. The six regimes organize candidate mechanisms for that evaluation; the general need for cross-layer design predates this framework.

MKA builds on these established ideas. Its proposed integration couples ownership and interface costs to admissibility, service power, and a substrate research agenda. The reference planner is a small, inspectable specialization of multiobjective path labeling [25]. It exposes exactly when a local decision loses to a joint one. The comparisons below are between fully specified policies in that model; they do not benchmark Timeloop, MAESTRO, Halide, or fabricated hardware. Calibrated cost adapters from such tools are a concrete next integration step.

3. Measuring Completed Work

3.1. Workload and Service Requirements

Inference fixes the model, request distribution, and acceptable error; a solver fixes the matrix family and accuracy. The workload contract is

$$\mathcal{C}_W = (\mathcal{D}, \mathcal{Q}, \mathcal{L}, p_{\max}, \mathcal{B}), \tag{2}$$

where $\mathcal{D}$ specifies inputs and demand, $\mathcal{Q}$ output quality, $\mathcal{L}$ timing and throughput, $p_{\max}$ allowed failure probability, and $\mathcal{B}$ the measurement boundary.

The predicate $\mathcal{Q}(A) = 1$ accepts all required accuracy floors and error ceilings. Admissible implementations satisfy

$$\mathcal{A}_W = \{A : \mathcal{Q}(A) = 1,\ P_{\text{fail}}(A) \le p_{\max},\ A \models \mathcal{L}\}. \tag{3}$$

Margins are fixed before evaluation; a looser quality target defines a different comparison.

A *fixed-work experiment* fixes the job or batch, initial model/data residency, cache and power states, and setup reuse. Measure $T_W = t_{\text{done}} - t_{\text{release}}$ from input release at the declared boundary, before required preparation, until all outputs and required completion checks are available there. Charge setup once across the specified batch.

A *service experiment* uses boundary-arrival and completed-response events per request, including

admission waiting, batching, execution, and delivery. Predeclare the mean or empirical quantile used for T_W, with its estimator. Fix the offered trace, warm-up policy, rejection rules, and treatment of unfinished requests; follow a finite admitted trace through completion or declare a stationary window and censoring rule. Use the same protocol across implementations.

Completed-work survival principle. MKA treats a local improvement as system progress only to the extent that its advantage remains after all changed data movement, state placement, interfaces and conversion, scheduling and queueing, accuracy, reliability, and service constraints are charged within the common boundary around completed useful work. A local speedup or efficiency result that disappears after those terms are included remains a valid component result, but it is not an end-to-end architectural gain. The principle is broader than the scalar MKA Law: it provides a common standard by which device, compiler, runtime, and infrastructure innovations can be compared on the same completed work rather than on incompatible local metrics [1], [2], [26].

3.2. The Proposed MKA Progress Rule

For the protocol's positive energy and time quantities, define

$$D_W = \frac{T_W}{\tau_{\mathrm{ref}}}, \qquad \Lambda_W = \frac{1}{E_W D_W} = \frac{\tau_{\mathrm{ref}}}{E_W T_W}, \quad (4)$$

using the same $\tau_{\mathrm{ref}} > 0$ across substrates, conventionally one second. With T_W in seconds and E_W in joules, $D_{\mathrm{eff}} \equiv D_W$ is dimensionless and Λ_W has units J^{-1}. Effective depth is normalized realized time, not hop count. Compute, transfers, conversion, scheduling, and queueing contribute through elapsed time; overlapping durations are not added twice. Publish event timestamps, aggregation code, clock resolution and synchronization error. A single monotonic clock or synchronized clocks must cover both endpoints. The same trace and protocol must reproduce the same D_W; repeated executions are compared within stated uncertainty.

Gonzalez and Horowitz used the energy–delay product to compare processor energy and performance jointly [27]. Equation (4) is its normalized inverse under a fixed workload contract. The term *MKA Law* denotes the proposed comparison rule; its product identity is algebraic and does not establish an empirical scaling law. For isolated work, $E_W = \bar{P}_W T_W$ weights the denominator as $\bar{P}_W T_W^2$. Energy budgets and deadlines remain constraints.

For two admissible implementations, write

$$\alpha = \frac{E_0}{E_1}, \quad \beta = \frac{T_0}{T_1}, \quad G_\Lambda = \frac{\Lambda_1}{\Lambda_0} = \alpha\beta. \quad (5)$$

The rule accepts $G_\Lambda > 1$ among admissible implementations. The *Compound Gain Regime* requires $\alpha > 1$ and $\beta > 1$, hence $G_\Lambda > \max(\alpha, \beta)$ regardless of statistical dependence. Throughput and financial value are evaluated separately.

Substitution in Eq. (4) gives $G_\Lambda = (E_0/E_1)(T_0/T_1)$ without an independence assumption. For example, $E_1 = E_0/2$ and $T_1 = 1.2T_0$ give $G_\Lambda = 5/3 > 1$, provided the slower design remains admissible. Simultaneous reductions are sufficient but unnecessary. A 4-J, 4-s design improves the product relative to a 10-J, 2-s baseline but fails a 3-s deadline. Reporting (E_W, T_W) alongside the index preserves this distinction.

For per-request quantities, covariance distinguishes the mean product from the product of means:

$$\mathbb{E}[ET] = \mathbb{E}[E]\mathbb{E}[T] + \mathrm{Cov}(E, T). \quad (6)$$

Service summaries and paired block log ratios serve different estimands; weighting and estimand are fixed before testing.

3.3. Energy Boundaries and Observation Periods

Observed input energy at boundary $\mathcal{B}$ is

$$E_{\mathrm{obs}} = \int_{t_0}^{t_1} P_{\mathcal{B}}(t)\, dt. \quad (7)$$

For a finite job, $E_W = E_{\mathrm{obs}}$; for a trace followed through completion, $E_W = E_{\mathrm{obs}}/N_{\mathrm{done}}$. Stationary service uses long-run energy per completion. Both alternatives include the same required hosts, optical sources, and board, server, or facility boundary. Programming and calibration are allocated over actual reuse, including preparation before timing.

Disjoint component accounting counts each cost once. Required static power is included; idle-subtracted measurements additionally describe incremental activity.

If fraction f_E of baseline energy improves by factor a_E and all other terms stay unchanged, Amdahl-style accounting gives [26]

$$\alpha_{\mathrm{sys}} = \left(1 - f_E + \frac{f_E}{a_E}\right)^{-1}. \quad (8)$$

For $f_E = 0.2$, $a_E = 10$, the gain is about 1.22. The time analogue requires unchanged, non-overlapping phase fractions; changed critical paths or queues require new timing analysis.

4. A Decoder and the Limits of Batching

4.1. Arithmetic and State in One Block

Prefill reuses weights across input positions. Decode produces one token per active request from stored keys and values [28]; at low batch, weight reads can dominate the small amount of arithmetic. Each step reads prior KV state, appends new keys and values, and feeds the sampled token into the next step. Finite traces unroll this recurrence.

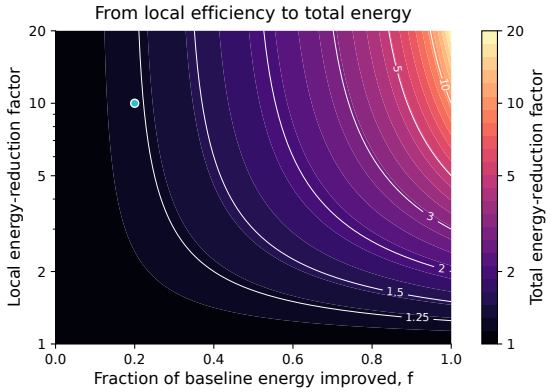


Fig. 3. Total energy gain when fraction f improves and the remainder stays unchanged. At $f = 0.2$, a local factor of 10 gives $1.22\times$ overall. MKA impact: impressive local efficiency becomes important only when it changes a meaningful share of full-task energy.

Count a multiply–add as two FLOPs. Let d be hidden width, f feed-forward width, n_h query heads, and $d_h = d/n_h$. With n_{kv} key/value heads, $d_{kv} = n_{kv}d_h$. GQA shares KV heads across query groups; MQA uses one [29], [30]. Variant comparisons require matched quality.

A GQA block with three-matrix SwiGLU [31] has, excluding biases and normalization,

$$N_{\text{block}} = 2d^2 + 2dd_{kv} + 3df. \tag{9}$$

The terms count query/output, key/value, and SwiGLU matrices. Fixed token arithmetic per layer is $F_0 = 2N_{\text{block}}$; a two-matrix feed-forward layer replaces $3df$ with $2df$.

4.2. Prefill, Decode, and Causality

Score formation and weighted-value accumulation cost approximately $4d$ FLOPs per query–key position pair, giving

$$F_{\text{prefill,layer}} \simeq sF_0 + 4dN_{\text{pair}},$$
$$N_{\text{pair}} = \begin{cases} s^2, & \text{full attention,} \\ s(s+1)/2, & \text{causal attention.} \end{cases} \tag{10}$$

These are useful causal operations; square-then-mask execution or tile padding may add work. FlashAttention reduces memory traffic by tiling exact attention [4].

A decoded token attending to s positions, including itself, requires

$$F_{\text{decode,layer}} \simeq F_0 + 4sd. \tag{11}$$

A full-model count adds softmax, normalization, positional encoding, gating, and $2dV_{\text{vocab}}$ FLOPs for a vocabulary of size V_{vocab}.

Equating the two terms in Eq. (11) gives an arithmetic crossover

$$s_\times = d + d_{kv} + \tfrac{3}{2}f. \tag{12}$$

Table 1
32K DECODER TRAFFIC AND RESIDENT BLOCK WEIGHTS PLUS KV CACHES, BEFORE WORKSPACE AND NON-BLOCK COMPONENTS.

Batch	Read GB/token	FLOP/byte	Resident GiB
1	147.64	1.51	137.5
16	19.29	11.55	287.5
128	11.81	18.87	1407.5

Equal arithmetic contributions do not determine the bandwidth bottleneck, which depends on traffic and hardware.

4.3. The 32K-Context Example

Use $L = 80$, $d = 8192$, $f = 28672$, $n_h = 64$, $d_h = 128$, and $n_{kv} = 8$, representative Llama-like dimensions [32]. The analytical workload assumes 32K context and two bytes per weight and KV element; checkpoint context support and quality still require validation.

Equation (9) yields 68.451 billion block parameters and 136.902 GB of weights, exactly 127.5 GiB. Arithmetic and traffic use this block-only scope; embeddings, output head, and normalization are additional costs.

At $s = 32{,}768$, fixed arithmetic contributes 136.902 GFLOP and attention 85.899, totaling 222.801 GFLOP per token. Attention's share is 38.6%; crossover occurs at $s_\times = 52{,}224$ (Fig. 4).

For one sequence, KV storage is

$$V_{\text{KV}} = 2Lsd_{kv}b_{kv}, \tag{13}$$

where b_{kv} is bytes per element. Per-sequence storage is 10 GiB for GQA, 80 GiB for 64-head MHA, and 1.25 GiB for MQA at fixed remaining dimensions. Appending one token adds 320 KiB across 80 layers before alignment and allocator overhead.

Assume B live sequences, one block-weight read per step, and one full KV read per sequence. With $W_b = 136.902$ GB and $K_b = 10.7374$ GB,

$$V_{\text{read}}(B) \simeq \frac{W_b}{B} + K_b, \qquad I(B) = \frac{222.801 \times 10^9}{V_{\text{read}}(B)}. \tag{14}$$

V_{read} is in bytes. Each KV load is reused across its query group; rereads, appends, activations, and output-head traffic add to this estimate.

Weight and KV reads are equal at $B = 12.75$. Further batching mainly reduces the smaller term, approaching 20.75 FLOP/byte while resident memory continues growing (Fig. 5, Table 1).

Batch 128 requires over 1.37 TiB of modeled state. Sharding supplies memory at a communication cost. Quantization, shared prefixes, cache residency, and alternative attention can reduce storage or traffic, subject to quality and execution tests.

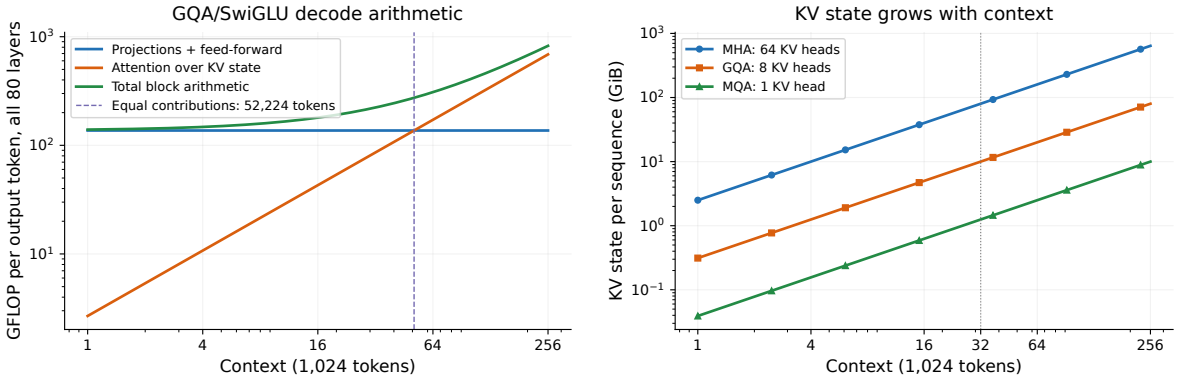


Fig. 4. Decoder with $L = 80$, $d = 8192$, $f = 28672$. Left: GQA/SwiGLU block arithmetic, excluding vocabulary head and elementwise work. Right: KV capacity under three attention layouts at fixed layer and head dimensions. MKA impact: model architecture can move the dominant optimization target from arithmetic to resident state.

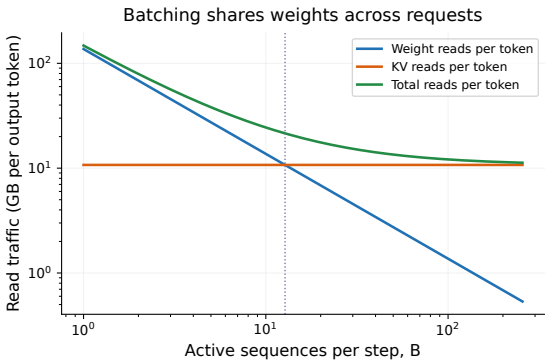


Fig. 5. Weight and KV reads at 32K context, excluding appends, MoE dispatch/return traffic, and scheduling. Batching amortizes weights; each sequence retains its KV reads. Table 1 gives capacity. MKA impact: once weight reuse saturates, KV placement becomes the next system-level target.

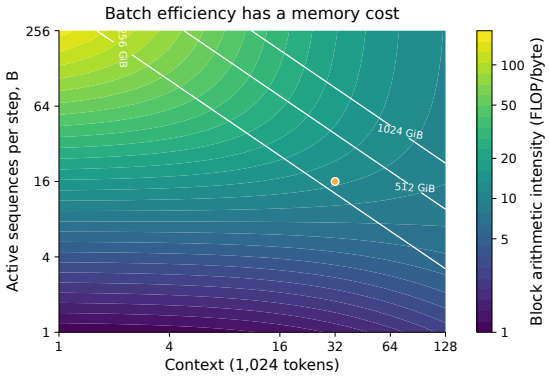


Fig. 6. GQA intensity across context and batch. White contours mark resident weights plus KV at 256, 512, and 1024 GiB. At 32K, batch 16 needs 287.5 GiB before workspace and non-block components. MKA impact: the feasible operating point is jointly set by context, batch, and memory capacity.

4.4. Can the Useful Batch Fit?

Let M_{dev} be available device memory, V_{work} reserved workspace, W resident weights, and K KV bytes per sequence. The capacity limit is

$$B_{\text{cap}} = \left\lfloor \frac{M_{\text{dev}} - W - V_{\text{work}}}{K} \right\rfloor. \tag{15}$$

A positive batch needs $M_{\text{dev}} \geq W + V_{\text{work}} + K$; fragmentation, duplication, and placement constraints further reduce capacity.

For target intensity $I_\star$ (for example, useful compute capacity divided by memory bandwidth) and F FLOPs per token, $F/(W/B + K) \geq I_\star$ requires

$$F > I_\star K, \qquad B \geq B_{\text{ridge}} = \frac{W}{F/I_\star - K}. \tag{16}$$

The target requires $\lceil B_{\text{ridge}} \rceil \leq B_{\text{cap}}$ as well as the intensity ceiling. Network and scheduling impose further limits.

At 32K, 50 FLOP/byte is impossible because the ceiling is 20.75. Reaching 10 FLOP/byte needs $B_{\text{ridge}} \simeq 11.86$, hence 12 sequences and 247.5 GiB. A 256-GiB budget fits at most 12 before reserving workspace, leaving 8.5 GiB for workspace and omitted model components.

Below crossover, batching repays weight reads. Long context favors reducing KV bytes or access cost. Evaluate sharding and new substrates where they change feasible batch and service time.

4.5. Parallel Execution and Request Scheduling

For ring all-reduce of m bytes across p participants, step latency τ_l and bandwidth B_l in bytes/s give

$$T_{\text{ring}} \simeq 2(p-1)\tau_l + \frac{2(p-1)m}{pB_l}. \tag{17}$$

This assumes balanced chunks and uncongested ring links; local reduction adds cost. Tensor-parallel execution depends on the bottleneck along the actual route [33], [34].

For MoE [35], expert-parallel execution includes token fan-out, all-to-all dispatch/return where placement requires it, and router-weighted aggregation before the residual update. Record both transfer directions. Let n_j be assignments to expert j

among E_x equal-rate experts and $\bar{n} = (\sum_j n_j)/E_x$. Imbalance is $\max_j n_j/\bar{n}$. Top-k routing totals k times the input-token count; heterogeneous experts require normalization by service rate. Dropping assignments requires a quality check.

For step time $T_{\text{step}}(B)$ and occupied-slot fraction $u(B)$, delivered rate is approximately $Bu(B)/T_{\text{step}}(B)$. Orca's iteration scheduling and PagedAttention address complementary costs [5], [36]. Larger batches reuse weights but may delay arrivals, so batch choice depends on the service objective.

5. Mapping Computation and State

5.1. Workload and Hardware Graphs

In workload graph $G_W = (V, E)$, vertices have compute and state demands. Edge e carries b_e bits on n_e occurrences, totaling $v_e = n_e b_e$. For data-dependent work, an instance is a realized trace; expected traffic supports planning rather than an execution-wise bound.

Hardware graph $H = (N, L)$ specifies supported operations and memory per node, and bandwidth B_l (bits/s) and propagation delay τ_l (seconds) per directed link. Placement $\phi : V \to N$, ordered directed routes ρ_e, and a schedule assign operations and transfers. A route connects the placed producer to its consumer; repeated transfers are counted in n_e. Node service times and transfer durations are in seconds, energy costs in joules, and normalized timing weights use the common divisor τ_{ref}. Precedence paths refer to the workload; physical routes refer to the hardware. Their bounds omit contention and waiting unless modeled explicitly, whereas realized completion and response times include the delays incurred by the executed schedule.

MKA's tuple $A = (G, T, S, M)$ records workload, logical embedding, substrate classes, and resources per node. Execution additionally specifies H, state ownership, and schedule.

Over period H_t, transfer count $N_e(H_t)$ gives rate $N_e(H_t)/H_t$. A stable homogeneous stream with job rate λ has long-run transfer rate $\lambda \mathbb{E}[n_e]$, hence link utilization

$$u_l = \frac{\lambda \sum_{e: l \in \rho_e} \mathbb{E}[n_e] b_e}{B_l}. \tag{18}$$

Persistent $u_l > 1$ is infeasible. Under ordinary stochastic operation, $u_l < 1$ is necessary for stability but may still yield excessive delay.

5.2. What a First Compiler Must Produce

Each persistent object has an owner, size, update rule, and consumers. Remote consumers create scheduled transfers; representation changes create conversions. Execution traces expose the resulting copies.

For $u = F_1(x)$ and $y = F_2(u)$ supported on one substrate, host-mediated calls may send u out and back. Keeping both operations and u resident avoids that round trip. At fixed operation costs and V_u intermediate bits,

$$\begin{aligned} \Delta E_{\text{resident}} = V_u(\varepsilon_{\text{out}} + \varepsilon_{\text{in}} - \varepsilon_{\text{local}}) \\ - \Delta E_{\text{state}} - \Delta E_{\text{control}}. \end{aligned} \tag{19}$$

Transfer costs are joules per bit; added state and control include keeping u live. Fusion requires supported operations and precision. A host-only nonlinearity still requires the return trip.

Emit placement, routes, buffers, and a replayable schedule. Verify write-before-read ordering, live capacity, links, and deadlock freedom at common operations, precision, and clocks.

5.3. Communication and Execution Lower Bounds

For a fixed routed instance, two immediate communication bounds are

$$\begin{aligned} T_{\text{prop}} &= \max_e \sum_{l \in \rho_e} \tau_l, \\ T_{\text{cong}} &= \max_l \frac{\sum_{e: l \in \rho_e} v_e}{B_l}. \end{aligned} \tag{20}$$

The Leighton–Maggs–Rao unit-packet schedule scales with congestion plus dilation [37]. Physical packets, buffers, and arbitration add timing constraints.

For required bits V_C and usable directional bandwidth B_C across hardware cut C, $T_{\text{cut}} = \max_C V_C/B_C$. Compute traffic after legal reduction, caching, replication, and multicast: aggregation can eliminate high-level graph transfers before the cut.

Let T_{cp} bound the precedence chain and T_{work} compatible compute demand. Interchangeable processors use total work divided by aggregate rate; heterogeneous operations require separate capacities. Then

$$T_{\text{LB}} = \max\{T_{\text{cp}}, T_{\text{work}}, T_{\text{cut}}, T_{\text{prop}}, T_{\text{cong}}\}, \tag{21}$$

with $T_{\text{realized}} \geq T_{\text{LB}}$. The index uses realized time. The ratio $T_{\text{realized}}/T_{\text{LB}}$ measures remaining modeled headroom; a large value may reflect the schedule or a loose bound.

5.4. Local Regularity Has a Global Cost

Fixed-density two-dimensional lattices with bounded-range links have diameter of order $\sqrt{N}$: local wiring simplicity entails longer global paths.

A balanced side-q hierarchy uses $O(\log q)$ ancestor hops; longer links and upper-level bandwidth carry physical area–time costs [38].

For symmetric traffic $w_{uv} \geq 0$, let $L_G = D_G - W$, with D_G containing weighted degrees. Assignment $x \in \{-1, 1\}^{|V|}$ has cut weight

$$\text{cut}(x) = \frac{1}{4} x^\top L_G x. \tag{22}$$

Spectral relaxations propose partitions [39], but a small isolated component may fail memory or compute balance. Multilevel partitioning and

constrained refinement can enforce those requirements [40].

Matched area or cost budgets, including wires, routers, and global links, isolate placement from added hardware.

For physical hierarchy, logical hop count must be paired with wire provision. The empirical Rent relation $P(B) = KB^r$ describes external terminal demand for partitions of logic blocks [41]. A two-dimensional region's perimeter grows as $\sqrt{B}$ at fixed density. When the fitted $r > 1/2$, terminal demand outgrows that perimeter, motivating extra routing layers, serialization, or longer links. A topology record should include wire span, bisection bandwidth, router area, and static power alongside graph distance.

5.5. A Controlled Example: Local Exchange Versus All-to-All

Consider an even-sided $q \times q$ mesh, $N = q^2$, with bandwidth B_0 per directed neighbor link. Both patterns below use the same compute work and outgoing budget b bits per node per round.

Local exchange sends $b/4$ bits to each interior neighbor; boundary nodes use at most that budget. A vertical bisection has q links and $qb/4$ bits in either direction, giving constant serialization bound $b/(4B_0)$.

Uniform all-to-all divides b among $N-1$ destinations. Assume distinct required messages prevent aggregation. The $N^2/4$ pairs crossing in one direction give

$$V_C = \frac{N^2 b}{4(N-1)},$$
$$T_{\text{cut}} = \frac{V_C}{qB_0} = \frac{q^3}{4(q^2-1)}\frac{b}{B_0}. \tag{23}$$

This grows as $\sqrt{N}\, b/(4B_0)$ at unchanged work and per-node payload because cut bandwidth grows only with side length. The model allows divisible payloads and omits headers; finite packets can increase execution time.

For global payload fraction φ and the remainder sent to neighbors,

$$\frac{T_{\text{cut}}}{b/B_0} = \frac{1-\varphi}{4} + \frac{\varphi q^3}{4(q^2-1)}. \tag{24}$$

On a 128×128 mesh, just 10% global payload raises the cut bound $13.70\times$. A locality advantage can disappear well before traffic becomes predominantly global. Sweeping φ at fixed work tests that boundary; express links and algorithmic reductions are charged for their resources.

5.6. Data Movement and the Active Bottleneck

With disjoint endpoint and link costs, the movement energy of a fixed instance is

$$E_{\text{move}} = \sum_e v_e \left(\varepsilon_{\text{src},e} + \sum_{l \in \rho_e} \varepsilon_l + \varepsilon_{\text{dst},e} \right), \tag{25}$$

Each ε is joules per delivered bit. The model charges copies and endpoint conversion as well as repeated reads; same-domain transfers still consume energy.

For F operations and $\mathcal{V}_k$ bytes through resource k, intensity $I_k = F/\mathcal{V}_k$ gives the multi-resource Roofline bound

$$\frac{F}{T} \le \min\{P_{\text{peak}},\ B_1 I_1, \ldots, B_m I_m\}, \tag{26}$$

with B_k in bytes/s [42]. Memory, network, and converters have their own traffic boundaries; a shared intensity axis requires explicit relationships between volumes.

Hong and Kung bound classical dense matrix-multiplication transfers by $\Omega(n^3/\sqrt{M})$ with M fast-memory words in a two-level model [43]. Different algorithms need corresponding bounds. Placement lowers required-transfer costs; algorithms can eliminate transfers. The active Roofline ceiling identifies the useful change.

6. An Exact Reference Planner and Its Evaluation

6.1. A Bounded Problem With Explicit Costs

We implement serial chains of n operations on k available execution sites. Every accepted site implements the same abstract operation. Its support and memory tests include the operation's workspace and both input and output buffers. A value remains at the producing site until consumed or transferred. Inputs begin at site 0; every plan returns its final result there.

A batch repeats one placement R times, sequentially. Setup enables each used site once per batch; changing kernel or operand work belongs in its recurring cost. Kernel and transfer costs recur, and local transfers have positive cost. All policies provision the same sites and incur the same static power P_s. Let d denote accumulated dynamic and setup energy and t batch duration. Total energy is $E = d + P_s t$. This model fixes overlap and power policy so that the placement comparison is unambiguous.

For operation i at site s, the dynamic cost and time are e_{is}, t_{is}. A transfer from previous owner r costs $c^E_{i-1}(r,s), c^T_{i-1}(r,s)$. If M is the set of already configured sites, the cost increment is

$$v_i(r,s) = (e_{is} + c^E_{i-1}(r,s),\ t_{is} + c^T_{i-1}(r,s)),$$
$$\Delta_i(r, M; s) = R v_i(r,s) + \mathbf{1}_{s \notin M}(H^E_s, H^T_s). \tag{27}$$

The initial state is $(r, M) = (0, \{0\})$ with label $(d, t) = (0, 0)$. A transition moves to $(s, M \cup \{s\})$. Unsupported or memory-infeasible choices are rejected before labels are propagated.

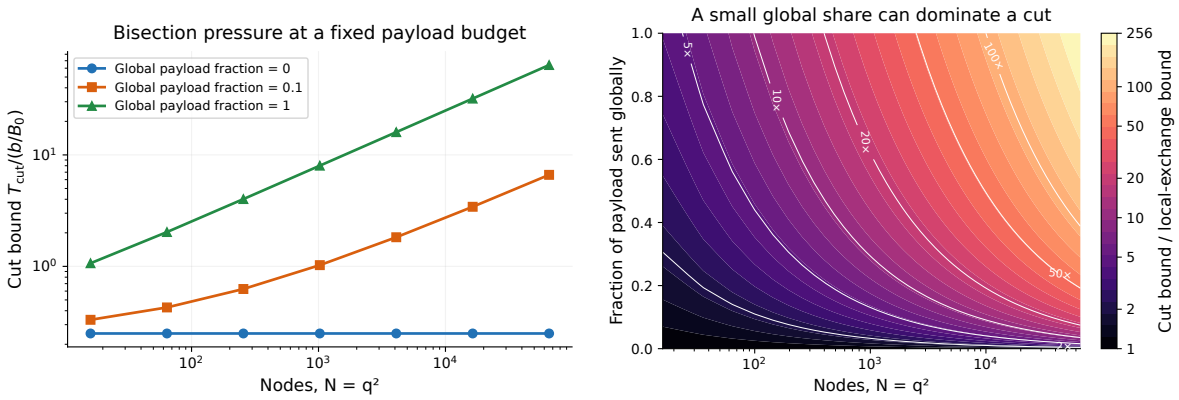


Fig. 7. Bisection bounds at fixed per-node payload and directed-link capacity. Left: local, 10%-global, and uniform all-to-all exchange. Right: the bound relative to local exchange as global fraction grows. Payloads are divisible. MKA impact: even a small global-traffic share can erase locality gains and justify hierarchy or a different interconnect.

6.2. Exactness and the Importance of Retaining State

At each layer and state, retain labels that are nondominated in (d, t):

$$\mathcal{F}_i(s, M') = \mathrm{ND} \bigcup_{r, M:\ M' = M \cup \{s\}} \{\ell + \Delta_i(r, M; s) : \ell \in \mathcal{F}_{i-1}(r, M)\}. \quad (28)$$

After adding the final transfer to site 0, choose the label minimizing $(d + P_s t)t$ subject to the batch deadline. This is a specialization of established multiobjective path methods [25] to the ownership and setup state required here.

Proposition 1 (Exact finite-chain planning)**.** *For the stated serial model with nonnegative additive costs, site-local feasibility, and constant $P_s \geq 0$, the recurrence retains a global energy–time optimum satisfying the deadline whenever one exists.*

Proof. Every legal assignment extends a legal prefix by one allowed transition. Two prefixes in the same layer, owner, and configured-set state admit the same suffixes, with identical suffix costs. If one prefix has no larger d or t, every common completion preserves that ordering. The objective $(d + P_s t)t$ is nondecreasing in both quantities, and a time deadline also preserves dominance. Induction therefore permits each pruning step without losing an optimum. The terminal transfer and final selection complete the argument. □

The configured-set state is essential: two prefixes with the same current owner can have different future setup costs. Pruning across those states would be unsound. General DAGs require additional state for live objects, joins, and resource overlap. The number of nondominated labels can grow exponentially; the recurrence offers exact finite-instance results rather than a polynomial bound for general mapping.

One frontier can also serve several fixed-power operating points. For plans a, b with dynamic energies d_a, d_b and durations t_a, t_b, their product difference, with $\Psi_a(P_s) = (d_a + P_s t_a)t_a$, is

$$\Psi_a(P_s) - \Psi_b(P_s) = d_a t_a - d_b t_b + P_s(t_a^2 - t_b^2). \quad (29)$$

Thus the preferred plan can change as persistent power increases. The retained (d, t) frontier supports that change by rescoring labels, without repeating the search.

6.3. Protocol and Comparison Policies

The released Cartesian suite has four families: uniform site preferences, alternating preferences, irregular support and costs, and tighter local memory. Each uses $n \in \{8, 16, 32\}$, inter-site cost scales $\{1, 4, 16\}$, reuse $R \in \{1, 16, 256\}$, and five deterministic seeds, giving 540 cases. Setup energy is $(0, 1800, 2500)$ and setup time $(0, 180, 250)$ in normalized units; $P_s = 1$. Base operation energy is drawn uniformly from integers 100–900 and duration from 20–140. Site multipliers vary by family. Standard memory capacities are $(10^6, 24, 32)$ storage blocks, reduced to $(10^6, 12, 16)$ in the capacity family. The complete generator, transfer matrices, and every generated instance are included. These normalized costs are synthetic and are not fitted to a particular device.

The deadline is 1.25 times the all-host duration, so a legal comparator always exists. We compare the joint planner with all-host execution, the best feasible fixed site, globally energy-optimal and time-optimal plans, and a greedy planner minimizing the full incremental energy–time product at each step. A stronger comparator chooses the best complete product among the two single-objective optima and the best fixed site. A boundary-blind ablation uses the exact planner with transfer costs omitted during selection, then evaluates the chosen plan under the full model. Every case, including ties and failures, is retained.

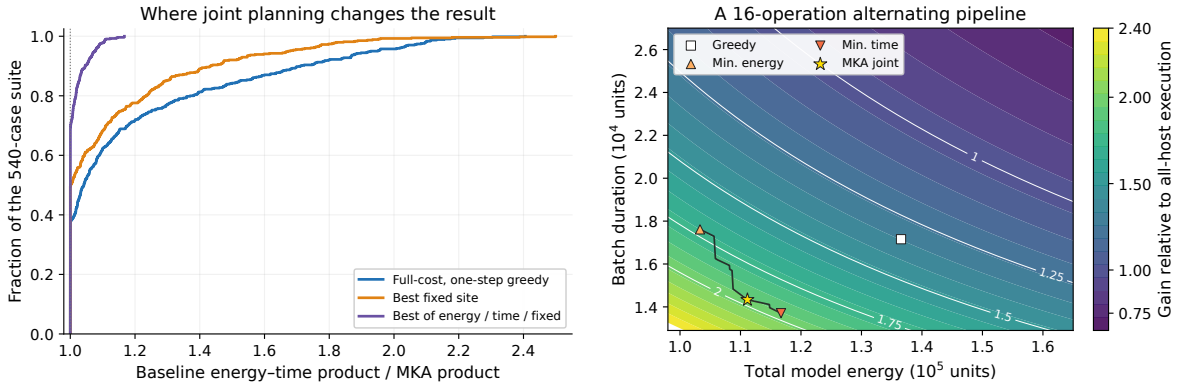


Fig. 8. Computed reference-planner results. Left: gain distributions across all 540 cases; the strong energy/time/fixed comparator often matches the joint optimum. Right: the specified alternating case with $n = 16$, link scale 4, $R = 16$, seed 0. The curve is its feasible energy–time frontier, and the background gives product gain against all-host execution. MKA impact: complete accounting avoids false local optima and recovers feasible energy–time gains.

Table 2
COMPUTED MODEL-LEVEL GAIN: BASELINE PRODUCT DIVIDED BY THE JOINT PLANNER'S PRODUCT. GEOMETRIC MEANS INCLUDE TIES. THE BOUNDARY-BLIND ROW USES ONLY ITS FEASIBLE CASES; DEADLINE FAILURES ARE COUNTED SEPARATELY.

Comparator	Feasible	Geo.	> 1.1	Max.
All host	540	1.428	315	3.502
Best fixed site	540	1.110	172	2.500
Energy optimum	540	1.035	62	1.603
Time optimum	540	1.050	98	1.454
Full-cost greedy	540	1.166	204	2.406
Best energy/time/fixed	540	1.011	12	1.167
Boundary-blind	351	1.174	126	2.360

6.4. Observed Planning Gains

The joint planner improves the product over full-cost greedy in 338 cases, including 204 gains above 10%, and ties it in 202. Its geometric-mean energy and time factors are 1.080 and 1.079, respectively; both improve in 234 cases. Against the best fixed site, both improve in 192 cases. These are placement and amortization gains on the same modeled resources.

The stronger comparator leaves a much smaller gap: 374 ties and 166 strict improvements, with a geometric-mean factor of 1.011. Most of the benefit comes from accounting for transfers and coordinating decisions across operations. Retaining both objectives then resolves a smaller set of trade-offs. The comparison does not establish an advantage over another exact optimizer given the same model and objective.

Boundary-blind selection violates the deadline in 189 cases, or 35%. In the example in Fig. 8, it alternates sites frequently and takes 26,030 time units, exceeding the 24,400-unit deadline. The joint plan uses site 2 for ten operations and site 1 for six, taking 14,334 units. Its product gain over greedy is 1.470, while the stronger comparator is already within 0.4% of it. A deadline sweep at 1.0, 1.5, and 2.0 times all-host duration produces 261, 144, and 111 boundary-blind failures, respectively; the full results accompany the main grid.

A separate enumerator sums primitive costs for all 3^n assignments on 36 instances with $n \in \{4, 6, 8\}$. Its complete nondominated sets match the recurrence in every case, covering 88,452 assignments before feasibility filtering. All 4,320 selected-plan replays preserve an exact modular-affine test computation; these are ownership and implementation checks, separate from the synthetic cost inputs. The largest retained layer in the main suite contains 518 labels.

6.5. How Much Model Error Would Erase a Gain?

Suppose each plan's true total energy and time lie within relative bounds $\epsilon_E, \epsilon_T < 1$ around its estimates. For estimated gain $\widehat{G}$, the worst-case ratio satisfies

$$G \geq \widehat{G} \frac{1 - \epsilon_E}{1 + \epsilon_E} \frac{1 - \epsilon_T}{1 + \epsilon_T}. \tag{30}$$

The result follows by decreasing both numerator quantities and increasing both denominator quantities to their allowed extremes. With separate 10% bounds, a gain must exceed 1.494 to remain above one. Eighty-six greedy comparisons pass that margin. The stronger comparator's gains do not. Applying this certificate to hardware requires calibrated bounds and a separate robust feasibility check for the deadline and memory constraints.

7. CHOOSING DESIGNS UNDER UNCERTAINTY

7.1. Ranking Feasible Mappings

For placement, routes, schedule, and substrate assignments x under scenario ξ, minimize

$$J(x, \xi) = \log \frac{E(x, \xi)}{E_{\text{ref}}} + \log \frac{T(x, \xi)}{T_{\text{ref}}}. \tag{31}$$

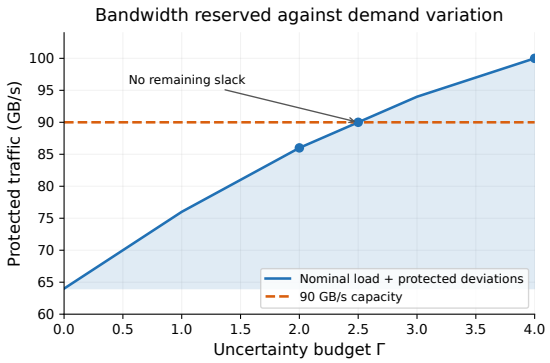


Fig. 9. Budgeted bandwidth protection. Fractional Γ permits a fractional deviation in the next-largest coefficient; capacity 90 GB/s is reached at $\Gamma = 2.5$. MKA impact: uncertainty becomes an explicit capacity-versus-efficiency decision instead of hidden safety margin.

This minimizes the energy–delay product under hard memory, quality, and timing constraints. Monetary cost can be a separate constraint or an explicitly normalized objective.

Group communicating operations, route candidates, and refine after scheduling. Ownership exposes copies; held-out traces test the combinatorial search's ranking model.

7.2. Bertsimas: Reserve Capacity for Uncertain Demand

Longer contexts or changed routes can overload a trace-fitted placement. Bertsimas and collaborators make protected uncertainty explicit [44], [45]:

$$\begin{aligned} &\min_{x\in\mathcal{X}} \quad \sup_{\xi\in\mathcal{U}} J(x,\xi), \\ &\text{subject to} \quad g_i(x,\xi) \le 0 \quad \text{for all } \xi \in \mathcal{U}. \end{aligned} \tag{32}$$

The predeclared set $\mathcal{U}$ covers operating variation against which every constraint must hold.

For uncertain coefficients $a_j = \bar{a}_j + \widehat{a}_j u_j$, let $\widehat{a}_j \ge 0$, $|u_j| \le 1$, and $\sum_j |u_j| \le \Gamma$. For nonnegative assignments x_j, the exact robust counterpart of $\sum_j a_j x_j \le b$ is

$$\begin{aligned} \sum_j \bar{a}_j x_j + \Gamma z + \sum_j p_j \le b, \\ z + p_j \ge \widehat{a}_j x_j, \quad z, p_j \ge 0. \end{aligned} \tag{33}$$

The Bertsimas–Sim capacity subproblem remains linear within the combinatorial search [44].

Four flows totaling 64 GB/s have possible increases of 12, 10, 8, and 6 GB/s. Protected loads at $\Gamma = 2, 2.5, 4$ are 86, 90, and 100 GB/s. A 90-GB/s link supports $\Gamma = 2.5$ with no slack; latency generally requires more headroom (Fig. 9).

An overload probability additionally requires distributional assumptions or testing [46]. Evaluate protection's capacity or latency cost against held-out overload rates; extra reservation has little value once service requirements are met.

The nominal price paid for protection can be recorded as

$$\mathrm{PoR}(\Gamma) = J(x_\Gamma, \bar{\xi}) - J(x_0, \bar{\xi}). \tag{34}$$

It is nonnegative when x_0 is a global nominal optimum and the robust feasible set is contained in the nominal one. Reporting this cost with held-out violations makes the reservation decision interpretable.

Context-dependent decisions [47] can choose batches from the current queue and memory. Tests must exclude future information and capture how policy changes feed back into queueing and observed service.

7.3. Rus: Generate Structures With Explicit Legality Checks

Daniela Rus's RoboGrammar generates robot designs through graph rules [48]; its multi-objective extension searches design and control jointly [49]. Decentralized reconfiguration links abstract rules to physical mechanisms [50]. We adapt this pattern to computing fabrics.

Rules $\mathcal{R}_S$ generate hardware from seed H_0. Accepted candidates also pass verification:

$$\mathcal{H}_S^{\mathrm{legal}} = \{H : H_0 \Rightarrow_{\mathcal{R}_S}^{*} H,\ \mathrm{Verify}_S(H) = 1\}. \tag{35}$$

Rules enforce ports and local connections; verification checks global timing, routing, thermal feasibility, and deadlock freedom.

Portability preserves required input–output behavior across hardware graphs and schedules. Analog behavior includes error tolerance; quantum interfaces enforce restrictions on copying unknown states.

Low-energy and higher-headroom candidates can both be useful; the contract selects among their operating margins.

A distributed realization also needs local control. A node updates from its own state, observations, and permitted neighbors:

$$s_i(t+1) = F_i(s_i(t), \{s_j(t) : j \in N_i\}, o_i(t)). \tag{36}$$

For a buffered dataflow implementation, credits limit sends to available capacity, ownership identifies the valid copy, and a configuration epoch prevents mixing old and new routes. The verifier must establish these invariants for the actual protocol. Control messages, polling, and barriers consume energy and appear in the execution trace. This protocol remains a target for a distributed backend.

7.4. Reconfiguration Has a Minimum Useful Duration

If switching costs E_{sw} joules and saves $\Delta P > 0$ at unchanged work rate for H seconds, energy improves when

$$H\Delta P > E_{\mathrm{sw}}. \tag{37}$$

Switching costs include state movement and calibration, with timing constraints enforced during interruptions. Frequent switches may consume the entire saving.

Adaptation is evaluated against a validated static baseline on held-out demand, with ownership and legality preserved through each transition.

8. Substrate Choices and Their Break-Even Conditions

Substrates act on different parts of the graph (Table 3). Analog in-memory arrays combine computation and storage; photonics can connect several compute stages. Recurring interface costs often decide which choice pays.

8.1. A Common Boundary-Cost Test

For a behavior-preserving replacement of U, let S_U be internal energy saved per invocation, $\Delta E_{\partial U}$ the signed change in boundary communication and conversion, H_{per} added recurring overhead, and H_{setup} setup reused R times. Energy improves exactly when

$$S_U > \Delta E_{\partial U} + H_{\text{per}} + \frac{H_{\text{setup}}}{R}. \tag{38}$$

Terms refer to identical completed work within this additive model; unchanged costs cancel. This is an energy break-even test. Timing, accuracy, reliability, and resource constraints still determine admissibility and the full energy–time comparison.

Conversion recurs unless residency or an eliminated crossing avoids it. Programming and calibration amortize over reuse; ordinary same-domain communication remains charged.

With recurring margin $m_E = S_U - \Delta E_{\partial U} - H_{\text{per}} > 0$, reuse must satisfy

$$R > \frac{H_{\text{setup}}}{m_E}. \tag{39}$$

For $m_E \le 0$ and nonnegative setup, no reuse count gives strict improvement.

At normalized baseline one and setup 12, recurring cost 0.24 first wins at $R = 16$; cost 0.60 needs $R > 30$; cost 1.10 never wins. Separating core cost 0.24 from boundary cost c (Fig. 10), required reuse diverges as $c \to 0.76$. This threshold follows from the chosen 0.24 core cost; it is not a universal analog limit.

A faster operation may expose another critical path or create a shared-tile queue. Sequential time savings must exceed added transfer, conversion, and waiting; concurrency requires a new schedule.

8.2. Joint Energy and Time Break-Even for a Reusable Batch

Suppose a baseline performs R sequential solves at e_0 joules and t_0 seconds each. A candidate uses e_1, t_1 plus batch setup H_E, H_T. With identical outputs, no overlap, and setup fully charged,

$$G_\Lambda(R) = \frac{R^2 e_0 t_0}{(H_E + R e_1)(H_T + R t_1)}. \tag{40}$$

Set $a = e_0 t_0 - e_1 t_1$, $b = H_E t_1 + H_T e_1$, and $c = H_E H_T$. For $a > 0$, the product improves when

$$R > R^\star_\Lambda = \frac{b + \sqrt{b^2 + 4ac}}{2a}. \tag{41}$$

This solves $aR^2 - bR - c > 0$. If $a \le 0$ and setup is nonnegative, improvement is impossible. Compound gain additionally needs $e_1 < e_0$, $t_1 < t_0$, and

$$R > \max\left\{ \frac{H_E}{e_0 - e_1}, \frac{H_T}{t_0 - t_1} \right\}. \tag{42}$$

For assumed $e_0 = 1$ J, $t_0 = 1$ s, $e_1 = 0.24$ J, $t_1 = 0.6$ s, $H_E = 12$ J, and $H_T = 4$ s, the first integer product gain is $R = 14$; both axes improve at $R = 16$.

8.3. Stage I: Spatial Digital Execution

A field-programmable gate array (FPGA) or coarse-grained reconfigurable array (CGRA) supports connected stages and local state. With frequency f_i, initiation interval II_i cycles, and replicas u_i, stage i has rate

$$r_i \le \frac{u_i f_i}{II_i}. \tag{43}$$

The slowest stage bounds a single-rate pipeline. Multirate graphs first normalize to a common output. Replication helps until another stage, memory port, or link limits execution.

At matched operations and precision, charge configuration, fill, transfers, and execution over reuse [11], [16]. Compilation is separate unless online. Routing and buffers penalize irregular graphs; setup penalizes small jobs.

8.4. Stage II: Analog Operations and Their Numerical Limits

Ambrogio et al.'s analog chip contains 35 million phase-change-memory devices across 34 tiles and reaches 12.4 TOPS/W chip-sustained performance [51]. Full-system comparisons also include its auxiliary operations and off-chip support.

Abdelaleim, Prakash, Irfanoglu, and Milutinović propose a resistive symmetric positive-definite (SPD) solver [52]. Version 2, dated 24 January 2026, reports analytical derivations and SPICE simulations, including circuit nonidealities. It distinguishes a passive subclass from a general active negative-resistance construction. These are circuit-model results, not measurements of fabricated solver hardware. The claimed size-independent analog phase is conditional on the circuit, matrix spectrum, and permitted component scaling. Full workload evaluation additionally charges programming, input/output, calibration, energy, component

Table 3
SIX SUBSTRATE REGIMES: INTENDED BENEFIT, EVIDENCE AVAILABLE, AND MATCHED SYSTEM-LEVEL TEST. EVIDENCE DESCRIBES THE CITED TECHNOLOGY, NOT A FABRICATED MKA SYSTEM.

	Implementation	Intended benefit	Available evidence	Matched system-level test
I	Reconfigurable digital fabric	Retain operations and intermediate state in a spatial pipeline.	Implemented dataflow precedents; MKA backend proposed.	Same work, precision and resources; include host, routing and configuration.
II	Analog computation	Use summation or relaxation for suitable numerical operations.	Analog-AI chip measurements; cited SPD solver is analytical/SPICE.	Same accuracy; include programming, readout, calibration and settling.
III	Processing near memory	Reduce transfers of stored weights or graph state.	Published PIM hardware precedents.	Saved traffic must exceed peripherals, updates and coordination.
IV	Photonic links	Add capacity at communication cuts.	Measured components and limited complete links.	Same delivered traffic; include source, tuning, interfaces and actual load.
V	Quantum-dot devices	Execute local quantum operations with shared control.	Small-array device demonstrations.	Same quantum task and fidelity; account for control and cooling.
VI	Entanglement-assisted links	Connect quantum states across modules.	Protocol theory and quantum-network precedents.	Same task and fidelity; include failed preparation and classical signaling.

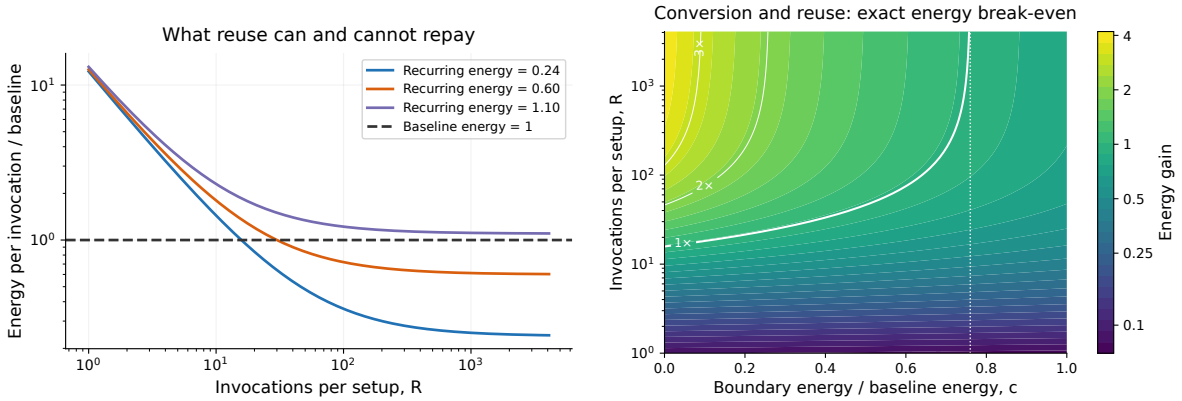


Fig. 10. Left: $E(R) = e_\infty + 12/R$ at three recurring costs. Right: $G_E = 1/(0.24 + c + 12/R)$ with per-invocation boundary cost c. At $c \geq 0.76$, no reuse count gives strict gain. MKA impact: the break-even boundary shows when a new substrate can repay conversion and setup at all.

count, accuracy, and conditioning. Dense instances require quadratically many coefficient settings or equivalent resources. These dependencies motivate a matched-tolerance comparison; they do not establish $O(1)$ end-to-end time for a complete digital-input/digital-output solve, unbounded gains, a computational-complexity-class transition, or universal superiority over digital execution.

The spectral requirement can be seen in an idealized relaxation model, analyzed separately from that circuit proposal:

$$\dot{x} = -\omega K(Ax - b), \qquad A \succ 0,\ K \succ 0. \tag{44}$$

Scaled matrices A, K are dimensionless SPD and $\omega > 0$ has units s^{-1}. At equilibrium $x^\star = A^{-1}b$. With $z = K^{-1/2}(x - x^\star)$ and $S = K^{1/2}AK^{1/2}$, $\dot{z} = -\omega S z$ gives

$$\|z(t)\|_2 \leq e^{-\omega\lambda_{\min}(S)t}\|z(0)\|_2. \tag{45}$$

A sufficient run duration for a relative reduction $\epsilon \in (0, 1)$ is

$$t \geq \frac{\log(1/\epsilon)}{\omega\lambda_{\min}(S)}. \tag{46}$$

Symmetry in transformed coordinates gives this bound; general non-normal dynamics can transiently grow.

The smallest eigenvalue in Eq. (46) can depend on dimension. For $K = I$ and $A_n =$

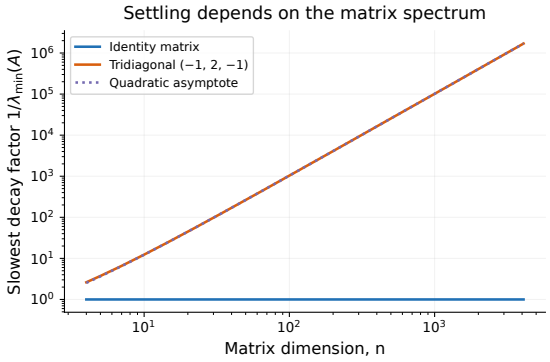


Fig. 11. Relaxation with $K = I$: the plotted factor multiplies $\log(1/\epsilon)/\omega$. The smallest eigenvalue determines dimension dependence. MKA impact: analog advantage depends on the matrix spectrum and requested accuracy, not device count alone.

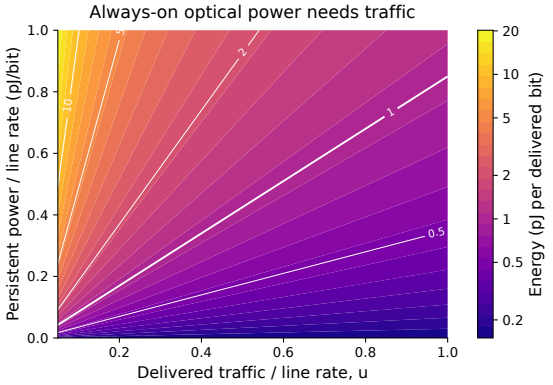


Fig. 12. Optical energy with assumed variable cost 0.15 pJ/bit. The 1-pJ/bit contour gives electrical break-even; persistent power is allocated over delivered bits. MKA impact: photonics becomes valuable only when traffic utilization is high enough to repay persistent optical power.

$\mathrm{tridiag}(-1, 2, -1)$,

$$\lambda_{\min}(A_n) = 4\sin^2\frac{\pi}{2(n+1)} \sim \frac{\pi^2}{(n+1)^2}. \quad (47)$$

This family settles quadratically in n; $A_n = I$ remains constant (Fig. 11). Comparisons include preconditioner construction and realization at fixed tolerance and permitted component scaling.

For $(A + \Delta A)(x + \Delta x) = b + \Delta b$, set $\delta_A = \|\Delta A\|/\|A\|$ and $\delta_b = \|\Delta b\|/\|b\|$. With nonzero b, compatible induced norms, and $\kappa(A)\delta_A < 1$,

$$\frac{\|\Delta x\|}{\|x\|} \le \frac{\kappa(A)}{1 - \kappa(A)\delta_A}(\delta_A + \delta_b). \quad (48)$$

Conditioning amplifies component error [53]. Residual $r = b - A\widetilde{x}$ gives $\|x - \widetilde{x}\|/\|x\| \le \kappa(A)\|r\|/\|b\|$, so report residual and conditioning together. Digital residual checks or refinement add their time and energy.

For a digital-input/digital-output analog solve,

$$E_{\text{analog}} = E_{\text{core}} + E_{\text{in}} + E_{\text{out}} + E_{\text{control}} + \frac{E_{\text{program}} + E_{\text{calibrate}}}{R}. \quad (49)$$

Inputs may use DACs or programmed conductances. Core energy includes static power during settling; readout returns every requested value through its finite interface.

Regularized least squares and covariance systems are suitable tests. Laplacians need a gauge or regularization to become nonsingular SPD; Newton Hessians require a positive-definiteness check.

8.5. Stage III: Processing Near Stored State

Processing-in-memory (PIM), exemplified by Aquabolt-XL [54], saves transfers at the cost of work near storage.

Eliminating ΔV off-chip bytes at baseline cost ε_{off} joules/byte gives net saving

$$\Delta E = \Delta V \varepsilon_{\text{off}} + \Delta E_{\text{other,saved}} - \Delta E_{\text{added}}. \quad (50)$$

Added costs include peripherals, updates, conversion, and coherence. Require $\Delta E > 0$ together with bank bandwidth, host coordination, and precision feasibility.

Weights are reused; KV is appended and read by request and head. Expensive writes may favor weights over KV, requiring separate placement.

8.6. Stage IV: Complete Photonic Links

Optics offers bandwidth density and distance-dependent transfer costs [55]. Daudlin et al. report 80-channel transmitter/receiver arrays, 800-Gb/s aggregate capability, and 5.3 Tb/s/mm^2. Separate front ends consume 50/70 fJ/bit at 10 Gb/s; the complete-link experiment uses four channels at 8 Gb/s [56]. System models retain those distinct measurement scopes.

For line rate R_l, utilization u, variable energy e_{var} per bit, and persistent power P_{on},

$$e_{\text{opt}}(u) = e_{\text{var}} + \frac{P_{\text{on}}}{uR_l}. \quad (51)$$

Persistent power covers the active source, receiver bias, tuning, and control. Full-rate front-end energy may contain static bias; gating adds wake-up costs.

Against an electrical comparator with energy $e_{\text{elec}} > e_{\text{var}}$, the energy break-even condition in this model is

$$u > u^\star = \frac{P_{\text{on}}}{R_l(e_{\text{elec}} - e_{\text{var}})}. \quad (52)$$

If $u^\star \ge 1$, full utilization still fails. At assumed $e_{\text{var}} = 0.15$ pJ/bit, $P_{\text{on}}/R_l = 0.30$ pJ/bit, and a 1-pJ/bit comparator, $u > 0.353$ is required (Fig. 12). Bursts and power gating alter this through wake-up costs.

Target a measured electrical cut bottleneck: a busy package link may repay optics where a local uncongested wire cannot. Include partitioning that avoids cut traffic as an alternative.

8.7. Stages V and VI: Quantum-Native Research Horizons

Semiconductor quantum dots support local quantum states and interactions [57], [58]. Borsoi et al.'s 16-dot germanium array uses shared control with terminal scaling

$$N_{\text{terminals}} = 6\sqrt{g} - 1 \tag{53}$$

for that architecture [59], giving 23 terminals at $g = 16$. Scaling also requires fidelity and cooling measurements.

Shared control limits independently adjustable parameters. Cross-capacitance maps, including rectangular or rank-deficient maps, support calibration. Useful quantum graphs require sufficient control and readout with calibration and cryogenic costs included.

Stage VI connects modules through teleportation using entanglement and classical communication, subject to no-cloning and causality [60]. Generation and management of entanglement are explicit resources [61], [62].

Let two independent links succeed with probability p per synchronous trial of duration τ, storing successes without decoherence. A swap succeeds with probability p_s; failure restarts both links. Then

$$\mathbb{E}[T_2] = \frac{\tau}{p_s}\frac{3-2p}{p(2-p)}. \tag{54}$$

Independent geometric trials give $\mathbb{E}[\max(X_1, X_2)] = 2/p - 1/[p(2-p)]$; $1/p_s$ counts swap cycles. For $p \ll 1$, the mean approaches $3\tau/(2pp_s)$. Swap duration, finite memory, and signaling add costs or change the model.

Quantum comparisons fix task and fidelity, then charge preparation failures, control, and transfer time.

9. Accuracy and Physical Limits

9.1. Accuracy at the Output

Let δ_v bound error at vertex v, ϵ_v locally introduced error, and L_{vu} sensitivity to predecessor u over the operating region. Then

$$\delta_v \le \epsilon_v + \sum_{u \in \text{pred}(v)} L_{vu}\delta_u. \tag{55}$$

Sum all DAG paths, including reconvergence, to allocate precision where sensitivity is high.

Bounds must cover reached perturbations. Cycles need convergence or stability analysis, correlated errors need a shared-source model, and nonlinear discrete outputs require application tests.

The Walden relation $P_{\text{ADC}} = F_W 2^{\text{ENOB}} f_s$ links converter power to effective bits, sampling rate, and implementation-dependent F_W [63], [64]. Precision changes require corresponding converter characteristics.

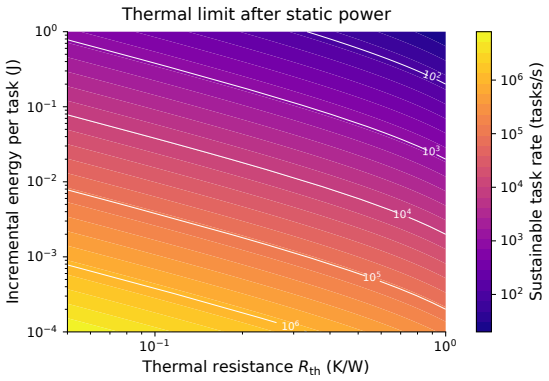


Fig. 13. Thermal rate bound at 40-K allowance and 20-W static power, as resistance and incremental task energy vary. MKA impact: lower energy per task converts directly into sustainable throughput and cooling headroom.

9.2. Failures and Retries

For failure events F_i, the union bound gives

$$\Pr\left(\bigcup_i F_i\right) \le \sum_i \Pr(F_i). \tag{56}$$

The bound allows dependence; multiplying success probabilities requires independence. Common voltage, temperature, or calibration can violate it. Count faults and retries.

Record numerical checks, communication errors, and retry policy; charge redundancy and retries to energy and time. Quantum error control respects no-cloning.

9.3. Thermal Headroom and Sustainable Throughput

A standard compact thermal model writes

$$C_{\text{th}}\dot{\theta} + G_{\text{th}}\theta = P(t), \tag{57}$$

where θ is rise above ambient. Package-specific resistance–capacitance models support local thermal analysis [65].

For thermal resistance R_{th}, allowed rise ΔT, static power P_s, and incremental task energy e, steady rate r satisfies

$$P_s + re \le \frac{\Delta T}{R_{\text{th}}}. \tag{58}$$

A positive sustainable rate requires $P_s < \Delta T/R_{\text{th}}$. For the feasible case,

$$r_{\max} = \frac{\Delta T/R_{\text{th}} - P_s}{e}. \tag{59}$$

At assumed $\Delta T = 40$ K, $R_{\text{th}} = 0.5$ K/W, and $P_s = 20$ W, 60 W remains dynamic. A 1-mJ task permits at most 60,000 tasks/s; halving energy doubles this bound at unchanged parameters.

At temperature T, resetting an unbiased unknown bit irreversibly under the usual isothermal assumptions has the Landauer dissipation bound $k_B T \ln 2$ [66]. An implementation erasing N_{irr} such

bits therefore has the conditional floor $N_{\mathrm{irr}} k_B T \ln 2$. The erasure count must be justified for the actual algorithm and implementation. Arithmetic operations, FLOPs, and attention pairs are not automatically irreversible bit erasures. Bennett's reversible-computation result makes this distinction explicit [67]. We infer no universal MKA ceiling or violation of the second law from an arithmetic count. A bound on Λ_W would require both a justified erasure lower bound and a compatible completion-time lower bound.

10. Execution Speed, Service, and Power

10.1. Harchol-Balter's Operational Perspective

Harchol-Balter's queueing research connects execution to waiting, variable jobs, parallelism, and response tails [68], [69]. Placement changes the service process managed by the runtime.

For one FCFS server with Poisson arrival rate λ and independent service requirements X, assume $m_1 = \mathbb{E}[X] > 0$, $m_2 = \mathbb{E}[X^2] < \infty$, and $\rho = \lambda m_1 < 1$. Pollaczek–Khinchine gives

$$\mathbb{E}[T_0] = m_1 + \frac{\lambda m_2}{2(1-\rho)}. \tag{60}$$

Long jobs delay those behind them; equal mean service can therefore yield different waiting times.

Scaling service to X/a, $a > 1$, at unchanged arrivals and policy gives

$$\mathbb{E}[T_a] = \frac{m_1}{a} + \frac{\lambda m_2}{2a(a-\rho)}. \tag{61}$$

Because $a - \rho > 1 - \rho$, $a\mathbb{E}[T_a] < \mathbb{E}[T_0]$ whenever $\lambda m_2 > 0$. Faster service also reduces waiting, amplifying mean response gain.

For exponential service times, this simplifies to

$$\frac{\mathbb{E}[T_0]}{\mathbb{E}[T_a]} = \frac{a-\rho}{1-\rho}. \tag{62}$$

At $\rho = 0.8$, $1.5\times$ acceleration gives $3.5\times$ mean response gain. Batching and multiple servers require their dependency models; tails are separate. Infinite second moment permits infinite mean waiting below saturation.

10.2. Power Cost of Service Acceleration

Harchol-Balter and collaborators jointly evaluate power and response time [70], [71]. At stable fixed admitted rate λ, measured energy per completion is $\bar{E} = \bar{P}/\lambda$, giving

$$\Lambda_{\mathrm{service}} = \frac{\lambda \tau_{\mathrm{ref}}}{\bar{P}\,\mathbb{E}[T]}. \tag{63}$$

At fixed λ, this minimizes the established power–response-time product [71], allocating observation-period energy per completion.

For power ratio $\pi_P = \bar{P}_1/\bar{P}_0$ at unchanged admitted rate, the M/M/1 case gives

$$G_{\mathrm{service}} = \frac{a-\rho}{\pi_P(1-\rho)}. \tag{64}$$

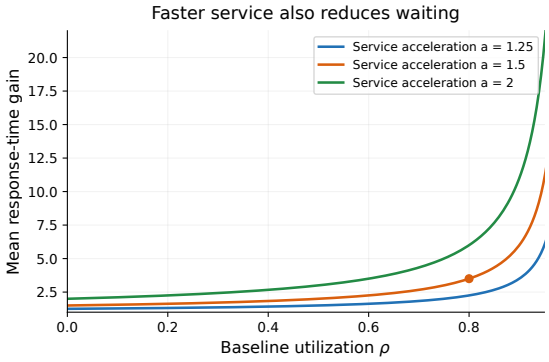


Fig. 14. Exact M/M/1 response gain at fixed arrival rate, uniform service acceleration, and baseline utilization up to 0.95. MKA impact: reducing service time near saturation can create superlinear user-visible latency gains.

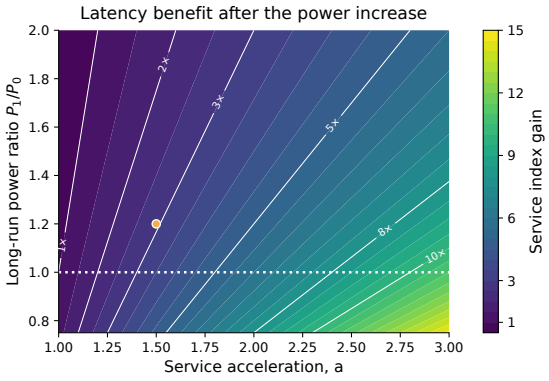


Fig. 15. Service gain at $\rho = 0.8$; dotted line marks unchanged power. At $a = 1.5$, $\pi_P = 1.2$, index gain is $2.92\times$ with higher request energy. MKA impact: the preferred operating point balances power and latency instead of maximizing either quantity alone.

At $\rho = 0.8$, $a = 1.5$, and $\pi_P = 1.2$, the index improves $2.92\times$: response improves $3.5\times$ while request energy rises 20%. A strict energy budget can reject this otherwise favorable trade-off.

10.3. A Latency Target Limits Usable Capacity

For FCFS M/M/1 service rate μ and $\lambda < \mu$, the stationary response tail is

$$\Pr\{T > \ell\} = e^{-(\mu-\lambda)\ell}. \tag{65}$$

Conditioning on the geometric queue length and mixing the resulting Erlang service sums yields rate $\mu - \lambda$ [68].

Requiring $\Pr\{T \le \ell\} \ge p$ gives

$$\lambda \le \lambda_{\max} = \mu - \frac{\log[1/(1-p)]}{\ell}. \tag{66}$$

A nonpositive bound admits no positive arrival rate. With $\mu = 100$ requests/s, deadline 100 ms, and $p = 0.99$, capacity is 53.95 requests/s; at $\mu = 150$, it is 103.95. Capacity expansion and fixed-demand energy answer different questions. Batched LLM service needs its dependencies modeled or replayed.

Measured demand and speed–power curves can favor fewer fast servers, more slow servers, or an intermediate allocation [70]. A controller minimizes power subject to the same demand and latency requirements.

10.4. Scheduling and Setup Can Preserve or Erase the Gain

Size-based scheduling and SOAP's age-based policy analysis can improve response on existing hardware [72], [73]. Record size information and prediction error, then evaluate long jobs through conditional slowdown $\mathbb{E}[T \mid X = x]/x$ and class-specific latency.

Jobs needing m_j simultaneous devices obey $\sum_{j \text{ active}} m_j \le K_{\text{on}}$ for K_{on} available devices, plus locality, memory, and non-preemption constraints. Multiserver-job scheduling weighs short-job priority against occupying device groups [74].

AutoScale accounts for returning servers to service [75]; deterministic setup analysis shows how its distribution affects waiting [76]. Replaying equal admitted traces while recording power states and response reveals whether device savings lower fleet power.

11. Datacenter Impact and Scientific Workloads

11.1. Datacenter Energy and Operating Costs

For average IT power P_{IT} in kW, annual PUE $\zeta = E_{\text{facility}}/E_{\text{IT}}$, and tariff c_e in dollars/kWh,

$$C_E = 8760 P_{\text{IT}} \zeta c_e. \tag{67}$$

PUE includes cooling and other facility overhead exactly once. Multiplying facility-boundary energy by PUE again would double-count that overhead. Equation (67) concerns electricity expenditure, not total operating cost or revenue. Varying tariffs or carbon intensities require time-resolved energy and rates.

At fixed admitted demand, measured IT energy-per-request gain α_{IT} gives, if average power reduces accordingly,

$$\frac{E_{\text{facility},1}}{E_{\text{facility},0}} = \frac{\zeta_1}{\alpha_{\text{IT}} \zeta_0}. \tag{68}$$

This converts IT gain to facility energy using PUE values ζ_0, ζ_1. Facility-boundary measurement captures it directly; response-time gains have separate service value.

The fleet example in Section 11.3 gives the monetary sensitivity under one declared set of assumptions. Figure 16 instead shows how power that remains unchanged limits the saving achievable from a local improvement.

Capacity expansion or deferred equipment purchases depend on memory, queueing, demand, and provisioning granularity; use measured capacity and costs. Operational carbon follows energy and grid intensity, with manufacturing impacts accounted separately.

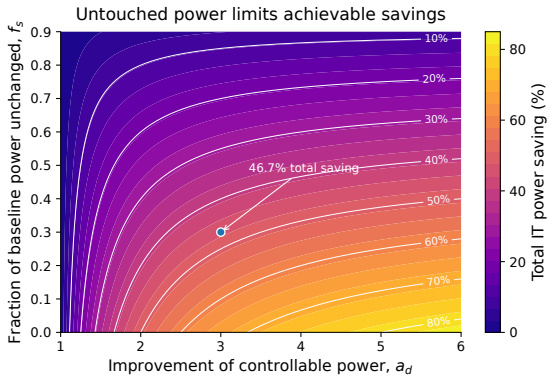


Fig. 16. IT power saving $100(1-f_s)(1-1/a_d)$ percent at fixed completed-work rate. The example $f_s = 0.30$, $a_d = 3$ gives 46.7% saving, not the 66.7% obtained when all power scales. MKA impact: persistent memory, interfaces and idle capacity limit how much local efficiency reaches the electricity bill. All factors are assumed sensitivities.

11.2. The Part of the Power Budget That Remains

If fraction f_s of baseline IT power stays fixed and the rest improves by a_d at unchanged work rate,

$$\frac{\bar{P}_1}{\bar{P}_0} = f_s + \frac{1-f_s}{a_d}, \qquad \alpha_{\text{IT}} = \left(f_s + \frac{1-f_s}{a_d} \right)^{-1}. \tag{69}$$

This applies Eq. (8) to power. Provisioned memory, interfaces, or other resources may retain their operating load.

For assumed $f_s = 0.30$ and $a_d = 3$, the resulting IT factor is 1.875, or 46.7% less power at the same work rate (Fig. 16). The local threefold factor is a sensitivity input, not a measured MKA result. Consolidation or gating can reduce the fixed fraction within memory and latency constraints; operating policy therefore determines financial value.

11.3. Prospective Six-Vector Optimization of Openly Released LLMs

Open-weight frontier models make model-specific systems optimization especially visible. Using published configurations current in September 2026, we compare Llama 4 Maverick, GLM-5.3, Kimi K3, and Qwen3.8-2.4T-A95B [77]–[80]. Maverick is a 400B/17B-active, 128-expert model with top-1 routing and a 1M-token context. GLM-5.3 uses the GLM-5.2 base architecture at 744B/40B active, 78 layers, 256 routed plus one shared expert, top-8 routing, DeepSeek Sparse Attention, and a 1M context. Kimi K3 scales to 2.8T/104B active, 93 layers, 896 routed plus two shared experts, top-16 routing, 69 KDA and 24 gated-MLA layers, and a 1M context. Qwen3.8-2.4T-A95B uses 2.4T/95B active parameters, 92 layers, 512 experts with 10 routed plus one shared active per token, 69 Gated DeltaNet and 23 full-attention layers, and 262K native context extensible to about 1.01M

tokens. These architectural differences directly change weight residency, context state, expert traffic, batching headroom, and power demand.

For serving model m, define a six-component control vector

$$z_m = (z_c, z_w, z_k, z_r, z_b, z_p), \tag{70}$$

where z_c selects compute kernels and numerical precision, z_w controls weight and expert residency, z_k controls KV and other request-state placement, z_r controls expert routing and interconnect placement, z_b controls batching, admission, and queueing, and z_p controls active capacity, power state, and the associated cooling load. These controls interact. Increasing batch can amortize weights while increasing KV residency and queueing; moving experts can reduce local memory pressure while increasing all-to-all traffic; lower precision can reduce traffic and power while tightening the quality constraint. The MKA operating point is therefore

$$z_m^\star \in \arg\max_{z \in \mathcal{Z}_m} \Lambda_m(z) = \arg\min_{z \in \mathcal{Z}_m} E_m(z) T_m(z), \tag{71}$$

subject to the model's quality, memory, reliability, and service-level requirements. For a finite nonempty admissible candidate set with positive costs, a minimizing configuration exists. Finding it requires measured or calibrated $E_m(z)$ and $T_m(z)$; the objective alone is not a search algorithm. The six coupled controls describe an operating-point optimization within the chosen model and hardware configuration, distinct from the six substrate regimes.

Table 4 gives a neutral BF16-equivalent scale view; actual released precisions differ and belong to the compute/precision decision vector. GQA cache follows Eq. (13). For compressed latent attention, a first-order context-dependent state proxy is

$$V_{\text{state}}^{\text{latent}} \simeq L_{\text{cache}} s (r_{\text{KV}} + d_{\text{rope}}) b_{\text{kv}}, \tag{72}$$

where L_{cache} counts layers retaining context-dependent latent state. KDA and Gated DeltaNet recurrent state is context-independent and is accounted separately. These footprints are architecture-derived proxies, not measured HBM traffic.

The contrast is operational. Maverick combines comparatively low active compute with a 1M GQA context, so request state and queueing can dominate. GLM-5.3 enlarges resident weights and expert routing while using compressed sparse-attention state. Kimi K3 pushes expert residency, top-16 routing, active compute, and power much further, while its KDA/MLA hybrid reduces context-dependent cache growth. Qwen3.8 likewise shifts pressure toward very large resident expert state and routing; its DeltaNet/full-attention hybrid moderates native-context KV growth despite 2.4T total parameters. Figure 17 makes the resulting pressure vectors explicit. These descriptors identify likely bottlenecks to measure. They do not establish a model's optimal configuration or a deployment speedup. The best admissible point depends on the measured request mix, hardware, and operating constraints; different models can sometimes share that point.

The fleet calculation makes the economic sensitivity explicit. With P_{fleet} in kW, fixed completed-request rate and quality, and unchanged PUE and electricity tariff,

$$S_{\text{yr}} = 8760 P_{\text{fleet}} \zeta c_e \left(1 - \alpha_{\text{IT}}^{-1}\right). \tag{73}$$

For $P_{\text{fleet}} = 5$ GW $= 5 \times 10^6$ kW, $\zeta = 1.20$, and $c_e = 0.07$ USD/kWh, every 10% verified reduction in system-level IT energy per completed request corresponds to approximately $368 million in annual electricity savings at fixed demand. A 30% reduction corresponds to $\alpha_{\text{IT}} = 1/0.70 \simeq 1.43$ and exceeds $1.1 billion annually. These are sensitivity thresholds rather than assumed MKA performance. Their purpose is to translate measured system-level efficiency into a transparent economic scale: even moderate verified gains can become material when sustained across a hyperscale serving fleet.

The reference planner in Section 6 already exercises a restricted subset of this idea by coordinating location, transfers, setup, and timing. Its feasible improvements show that even partial all-inclusive optimization can change the selected plan. The four-model case study identifies the next deployment-oriented step: calibrate all six vectors from real traces for a chosen model, solve Eq. (71), and verify the predicted energy–time improvement at the same quality and demand.

11.4. Applications Organized by the Cost That Can Change

The framework is intentionally broader than accelerator serving. It applies whenever the cost of moving, synchronizing, or maintaining state is comparable with the cost of the arithmetic performed on that state. The same completed-work principle therefore creates different engineering questions in different fields. In databases and search, a useful optimization may keep an index shard, embedding table, or reranking stage resident instead of moving intermediate results across machines. In molecular simulation and drug discovery, neighbor lists, graph messages, force calculations, and repeated linear-algebra kernels create a mixture of local reuse and global reductions. In robotics and autonomy, perception, sensor fusion, planning, and control form a latency-critical graph in which a locally efficient accelerator is useless if copying or scheduling causes the control deadline to be missed.

Finance and regulated computation combine streaming dataflow with numerical accuracy and re-

Table 4

PUBLISHED ARCHITECTURE SNAPSHOT USED IN THE OPEN-MODEL COMPARISON. BF16-EQUIVALENT WEIGHT FOOTPRINTS USE TWO BYTES PER PARAMETER; CONTEXT-STATE ENTRIES USE THE PUBLISHED ATTENTION ARCHITECTURE AND STATED CACHE/STATE PROXY. THE TABLE SHOWS WHY A SINGLE SERVING RECIPE IS UNLIKELY TO BE OPTIMAL ACROSS MODELS.

Model	Total / active	Context	Layers / attention	Experts / routing	BF16-equivalent reference state	Dominant MKA pressure
Llama 4 Maverick	400B / 17B	1M	48 layers; GQA, 40 Q / 8 KV	128 experts; top-1	800 GB eq.; 192 GiB KV	Long-context state, batching, residency, power gating
GLM-5.3	744B / 40B	1M	78 layers; DSA, latent KV rank 512	256 routed + 1 shared; top-8	1,488 GB eq.; 87.8 GiB state proxy	Expert residency, routing, compressed long-context state
Kimi K3	2.8T / 104B	1M	93 layers; 69 KDA + 24 gated MLA	896 routed + 2 shared; top-16	5,600 GB eq.; 27 GiB latent cache proxy	Extreme expert residency, routing, active compute and power
Qwen3.8-2.4T-A95B	2.4T / 95B	262K native	92 layers; 69 DeltaNet + 23 full attn.	512 experts; 10 routed + 1 shared	4,800 GB eq.; 23 GiB native KV	Massive residency and routing; hybrid attention limits native-context state

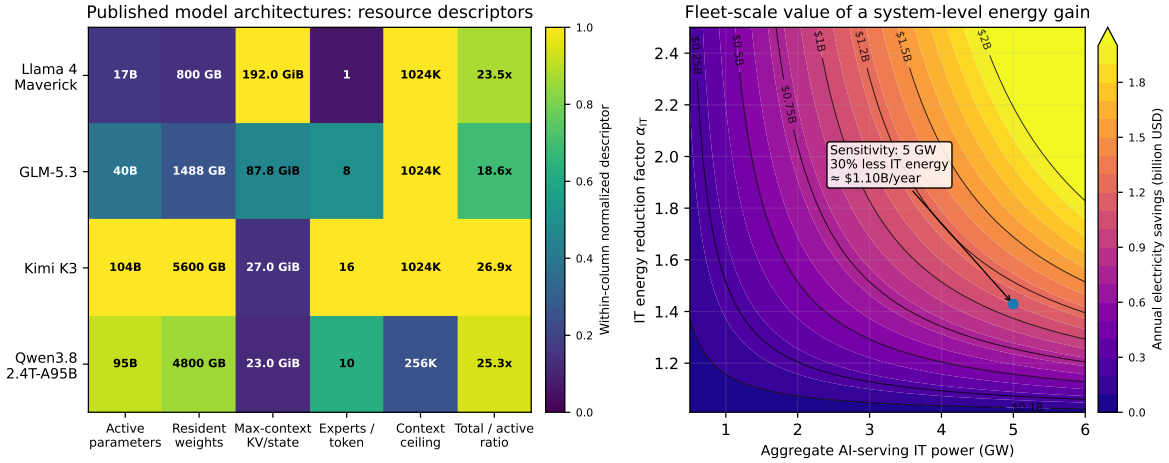


Fig. 17. Left: architecture-derived resource descriptors for four openly released MoE model families; raw values are printed in each cell and colors are normalized within each column. Right: annual electricity savings from Eq. (67) at PUE 1.20 and $0.07/kWh as aggregate AI-serving power and IT energy reduction vary. At 5 GW, each 10% verified reduction corresponds to about $368 million per year, while the marked 30% sensitivity point exceeds $1.1 billion. MKA impact: model-specific resource demands guide a calibrated six-control search, while verified energy gains determine fleet-scale savings.

producible decisions [15]. Pricing and risk aggregation can reuse resident state; covariance updates test whether setup is repaid. For $(\Sigma + \lambda I)w = b$, symmetric positive-semidefinite Σ and $\lambda > 0$ give an SPD test case. The result needs a residual, conditioning estimate, and deterministic replay record. In Monte Carlo risk, a g-fold sample-rate increase reduces sampling error by $1/\sqrt{g}$ at fixed duration for independent finite-variance samples, or finishes a fixed sample count earlier.

Genomics presents less regular locality. Variation graphs represent alternative sequences and structural variation [81]; traversal and alignment combine index lookups, dynamic programming, and sparse state access. For positive total graph traffic, a useful partition descriptor is

$$\ell_G = 1 - \frac{\sum_{e \in E_{\text{cut}}} n_e b_e}{\sum_{e \in E} n_e b_e}. \tag{74}$$

It records the fraction of traffic retained within partitions. High locality must be checked alongside bank balance and the maximum cut load. Read-mapping accuracy and the reference graph remain fixed when placements are compared.

Climate and scientific simulation add phase changes. In a d-dimensional stencil with local side length a and halo width h, the leading halo-to-volume ratio is approximately $2dh/a$ for $h \ll a$. Strong scaling therefore raises communication per local update. Coarse multigrid levels can favor different placements from fine levels. A useful test

follows the complete solve, including convergence, reductions, and remapping.

Telecommunications and edge inference expose hard timing limits. Factor-graph message passing underlies important decoding and inference methods [82]. LDPC decoding, channel estimation, beamforming, and sensor fusion offer distinct precision and locality tests. If I rounds are strictly sequential, $T \geq It_{\mathrm{round,LB}}$; reducing local arithmetic alone leaves the round count unchanged. Evaluation fixes bit-error or estimation quality and includes required host, memory, and radio-interface work.

Table 5 gives representative workloads, controlled comparisons, and end-to-end observables for all nine application areas. A common accounting method makes their results interpretable while preserving each domain's quality and timing requirements.

11.5. Algorithmic Changes and the Implementation Program

The substrate axis is only part of the research space. Sparsity, quantization, fusion, and asynchronous updates can change the work graph itself. Reversible computation trades erasure against space and control [67]; stochastic physical devices provide another route to sampling and optimization [83]. Each change is evaluated at the requested answer quality. This is consistent with the broader post-Moore opportunity in algorithms, software, and architecture [84].

A proposed three-year program has concrete gates. In year one, publish the graph/state schema, calibrated cost adapters, and a digital backend with matched placements and locality sweeps. In year two, test analog reuse and conditioning, PIM update traffic, and optical utilization on complete interfaces. In year three, integrate a multi-node service and one non-AI application, reproduce the result at a second institution, and release both successful and failing operating regions. Quantum experiments proceed through fidelity and preparation-rate gates appropriate to their own tasks.

12. Evaluation and Open Questions

12.1. From the Reference Result to a Hardware Claim

Implement Section 5's backend and compare placements on one device. Start with reusable local state, then sweep global payload using Eq. (24) through the point where routing erases the benefit.

Use a strong alternative and an ablation with MKA placement disabled, matching area, memory, and precision. A common legal clock isolates placement; also report each placement's best legal clock. Later sweeps vary analog reuse/conditioning, PIM updates, or optical load at fixed required output.

The *MKA certificate* records Table 6's configuration and measurements, including failed and non-improving runs. Workloads and primary comparisons are selected before evaluation. Existing graph/dataflow systems require the same comparison contract. Calibrate costs on separate runs and test predictions on held-out workloads; the index alone supplies no scaling forecast.

12.2. Paired Gain Estimation and Uncertainty

For independent paired workload blocks, define

$$y_i = \log \frac{E_{0,i}T_{0,i}}{E_{1,i}T_{1,i}}, \qquad \widehat{G} = \exp(\bar{y}). \tag{75}$$

This equally weighted geometric-mean gain applies to independent paired blocks. A weighted population needs matching weights and variance or resampling. Fix population and estimand before testing.

For approximately normal independent block log ratios, a one-sided lower confidence bound is

$$L = \bar{y} - t_{1-\eta,n-1}\frac{s_y}{\sqrt{n}}. \tag{76}$$

Positive L supports geometric-mean improvement. Block bootstrap is an alternative under its sampling assumptions. Power samples from one thermal run remain dependent; inference needs independent blocks or a time-series model.

Energy and time uncertainty are correlated. With $u = \log(E_0/E_1)$ and $v = \log(T_0/T_1)$,

$$\mathrm{Var}(y) = \mathrm{Var}(u) + \mathrm{Var}(v) + 2\,\mathrm{Cov}(u, v). \tag{77}$$

Paired resampling preserves covariance. Report systematic meter uncertainty separately: repetition reduces variability, not calibration bias.

For a planned detectable positive log-gain δ, block standard deviation σ_y, one-sided significance η, and power $1-\beta_s$, the normal approximation gives

$$n \gtrsim \frac{(z_{1-\eta} + z_{1-\beta_s})^2\sigma_y^2}{\delta^2}. \tag{78}$$

Here z_p is a normal quantile. Pilot data estimate variability; multiple workloads or candidate searches need a declared comparison strategy or held-out tests.

12.3. Quality and Rare-Failure Estimates

Assess actual output errors on matched inputs with a predeclared non-inferiority margin before comparing performance; equal nominal precision can yield different errors.

For n independent Bernoulli trials with no observed failures, the one-sided exact upper bound on failure probability at confidence $1-\eta$ is

$$p_U = 1 - \eta^{1/n}. \tag{79}$$

Since $\Pr(\text{zero failures}) = (1-p)^n$, 2,995 independent failure-free trials place the 95% upper bound below 10^{-3}. Correlation invalidates that count; sparse tail observations likewise need adequate replication.

Table 5
EXPANDED CROSS-DOMAIN RESEARCH PROGRAM. REPRESENTATIVE WORKLOADS ILLUSTRATE HOW THE SAME MKA RULE BECOMES A DIFFERENT PLACEMENT, STATE, OR SCHEDULING QUESTION IN EACH DOMAIN. A BENEFIT IS ESTABLISHED ONLY WHEN THE LISTED END-TO-END OBSERVATIONS IMPROVE AT MATCHED OUTPUT QUALITY.

Domain	Representative workloads	Candidate MKA mechanism	Controlled comparison	Decisive observations
AI serving and agents	MoE/long-context serving; RAG; tool-using agents	Weight/expert/KV placement; prefix reuse; routing; batching; fewer tool-boundary transfers	Same model, quality, context/request/tool traces and admitted demand	Bytes/token; HBM and network traffic; tail latency; joules/request; active fleet power
Finance and risk	Monte Carlo; streaming risk; pricing; fraud graphs	Resident aggregation; matrix reuse; streaming pipelines; graph partitioning	Same market trace, sample count, residual/error tolerance, and reporting deadline	Time-to-result; deterministic replay; setup amortization; joules/scenario
Databases, search, recommendation	Vector search; graph queries; joins; embedding lookup; reranking	Co-locate index shards and embedding state; fuse retrieval/rerank stages; cache hot subgraphs	Same corpus/index, recall or ranking quality, query mix, and tail-latency target	Bytes/query; cache hit rate; cross-shard traffic; p99 latency; joules/query
Genomics	Variation-graph mapping; sequence alignment; pangenome queries	Co-located graph partitions, indexes, and dynamic-programming state	Same reference graph, reads, scoring parameters, and mapping accuracy	Cut traffic; bank imbalance; reads/s/J; time-to-analysis
Molecular, materials, drug discovery	Molecular dynamics; GNN potentials; docking; sparse eigensolvers	Neighbor-list locality; resident coordinates/state; phase-specific dense/sparse kernels	Same molecule/system, timestep or convergence criterion, numerical tolerance, and ensemble size	Time/trajectory; neighbor and reduction traffic; energy/step; candidates/day/W
Climate and HPC	Stencils; multigrid; CFD; weather ensembles; sparse solvers	Halo locality; hierarchy; phase-specific placement and remapping	Same problem size, discretization, solver tolerance, and convergence criterion	Time-to-solution; global reductions; halo bytes; remapping cost; energy/solve
Telecom and edge	LDPC; MIMO; beamforming; channel estimation; sensor fusion	Local message passing; DSP/NPU pipelines; precision and state residency	Same channel/SNR trace, deadline, error target, and radio-interface scope	Iterations/rounds; BER or estimation error; missed deadlines; energy including interfaces
Robotics and autonomy	Perception-fusion-planning-control; multi-camera/ LiDAR pipelines	Sensor-local preprocessing; graph partitioning across CPU/GPU/NPU; deadline-aware state placement	Same sensor log, perception metric, route/task, and control deadline	End-to-end control latency; copied bytes; missed-deadline rate; joules/control cycle
Quantum-native tasks	Distributed sensing; modular quantum algorithms; entanglement-assisted computation	Shared control; quantum-state locality; entanglement routing and preparation scheduling	Same quantum task, fidelity, protocol, path length, and classical-control boundary	Preparation success; useful completion rate; fidelity; classical/cryogenic wall power

Table 6
EVALUATION RECORD AND COMPARISON ERRORS PREVENTED.

Item	Required evidence	Error prevented
Work and quality	Input or demand trace; model or algorithm version; quality tolerance; failure definition.	Comparing different useful work or silently relaxing accuracy.
Physical boundary	Included hosts, memory, interfaces, and overhead; calibrated input-power trace.	Moving energy outside the comparison or counting it twice.
Execution	Placement, routes, runtime and power states; release/completion events, clock resolution/error, and timing estimator.	Attributing software changes to devices or using incompatible timing boundaries.
Resources	Memory footprint, effective traffic, link provision, and physical or cost budget.	Comparing unequal hardware resources as a topology improvement.
Statistical decision	Independent paired blocks, uncertainty, primary endpoint, and tested comparisons.	Treating repeated samples or selected configurations as independent evidence.
Reproduction	Source, configurations, numerical inputs, and scripts for all reported results.	Providing plots that cannot be independently reconstructed.

12.4. Figure Provenance and Reproduction

The source package maps every figure to its equation, parameter values, generation script, and numeric data in `FIGURE_GUIDE.md`. Closed-form surfaces use the stated analytical models; planner plots use the complete synthetic suite, including ties and failures. The open-model panel uses architecture descriptors and storage proxies, not measured utilization or a computed six-control optimum. Its colors are normalized separately within each column. Economic surfaces vary assumed savings; external device results retain the measurement scope of the cited study. None of these plots is presented as a measured end-to-end MKA hardware result.

12.5. Scope of the Evidence

The 540-case study measures decisions in a synthetic cost model. Exact enumeration checks the implementation against that model; it does not calibrate the costs. The modular-affine replay tests common operation semantics and ownership

transfers. The planner does not yet schedule concurrent DAGs, implement a Honeycomb physical network, or run a real decoder. These boundaries define the next experiments rather than changing the interpretation of the reported ratios. All plot grids, generated instances, selected paths, failures, and verification code accompany the manuscript.

12.6. Operating Limits and Falsification

Consistent failure to improve the matched-budget energy–time frontier weakens placement. Excess routing or scheduling identifies compiler limits. A substrate fails where full interface and setup costs exceed its acceptable-quality benefit. Evaluate against the original workload criteria.

Conclusions depend on objective, tested workloads, legality checks, uncertainty set, and quantum-task semantics.

Report the first failing case alongside the best: the reuse, memory, or global-traffic threshold should explain the decision.

13. Conclusion

MKA proposes that the graph of useful work should remain the organizing object as computing technology changes. Its central claim is broader than the energy–time equation: a component or local hardware improvement becomes architectural progress only when its benefit survives data movement, state placement, interfaces and conversion, scheduling and queueing, accuracy, reliability, and service-level constraints at the boundary of completed useful work. The MKA Law supplies a quantitative objective for this completed-work survival principle, while the workload contract fixes the conditions under which the comparison is meaningful. This extends the Honeycomb direction initiated by Veljko Milutinović, its early development with Dragana Milutinović and B. Soucek, and the later mapping analysis with Borko Furht into a research program spanning physical substrates [6], [7], [9]. The enduring aim is to make architectural knowledge accumulate as devices change.

The reference planner establishes a first computational result within this program. For the stated serial-workflow model, it preserves an optimal admissible plan while accounting for data location, available memory, recurring transfers, and setup. Exhaustive checks verify the recurrence against direct enumeration. The experiments show that coordinated choices can improve both energy and completion time on unchanged modeled resources and recover feasible schedules that incomplete accounting misses. Comparisons with strong global alternatives show that much of the gain comes from representing the costs correctly and coordinating decisions beyond the next operation. These results support a concrete research direction: extend explicit ownership and full-cost planning to branching workflows, concurrent execution, and calibrated hardware models.

The wider scientific opportunity lies in the interaction between mechanisms. Keeping an intermediate value resident can eliminate a conversion and a transfer, but may consume memory needed by another operation. A faster device can reduce queueing, while its persistent power can alter the preferred schedule. A new interconnect can relieve a communication cut, yet change which partition is most efficient. MKA provides a common formulation in which these effects can be evaluated together. The communication bounds, accuracy conditions, and uncertainty models identify which change is worth implementing and which additional measurement could change the decision.

The six regimes turn this formulation into a long-term experimental agenda. Reconfigurable digital hardware provides an immediate platform for testing placement and state ownership. Analog operations and processing near memory offer ways to reduce suitable numerical and data-access costs; photonic links address transfers that remain expensive after partitioning. Quantum-dot and entanglement-assisted systems extend the agenda to tasks whose useful resources include quantum states and their preparation. Shared workload descriptions and explicit interfaces can make the findings from these experiments reusable. Each implementation contributes knowledge about the conditions under which a physical mechanism improves completed work.

The consequences extend from individual applications to computing infrastructure. Artificial intelligence services expose repeated access to large models and growing request state. Finance, genomics, and scientific simulation add streaming decisions, irregular graphs, and numerical convergence requirements. Telecommunications and edge inference place tight deadlines on iterative and sensor-driven computation. Their differences create complementary tests of the same architectural questions. The facility scenarios show how sustained system-level energy reductions could produce annual savings on a billion-dollar scale across a sufficiently large datacenter fleet under the stated assumptions. For scientific users, improved efficiency can instead support larger simulations, more samples, or faster analysis within the same available resources.

The open-model analysis motivates a joint optimization problem over six coupled serving controls: compute and precision, resident weights, KV and request state, expert and interconnect routing, batching and queueing, and active power with cooling. Llama 4 Maverick, GLM-5.3, Kimi K3, and Qwen3.8 occupy visibly different points in this resource space [77]–[80]. Their different demands motivate model-specific searches on the same

hardware fleet. Equation (71) turns that observation into a reproducible search problem. The fleet sensitivity in Section 11.3 translates verified IT energy reductions into electricity expenditure at unchanged demand, PUE, and tariff. It does not predict a saving from the architecture descriptors alone. The scientific claim is broader: optimizing all six vectors together exposes trade-offs that isolated kernel, memory, network, or scheduler metrics cannot resolve on their own.

This program builds on specific contributions from the researchers whose work we acknowledge. John L. Hennessy's compiler–machine co-design, his technology–architecture analysis with Norman Jouppi, and his quantitative and post-Moore work with David A. Patterson provide the evaluation discipline [1], [10]–[14]. Oskar Mencer's dataflow research provides an implementation precedent for predictable spatial execution [15], [16]. Mor Harchol-Balter's scheduling and server-power research connects execution choices to the service experienced by users [68], [70], [71], [75]. Dimitris Bertsimas's robust and prescriptive optimization informs decisions under uncertain demand [44], [45], [47]; Daniela Rus's graph grammars and decentralized reconfiguration inform the generation and control of realizable structures [48]–[50]. The analog-solver work of Osama Abdelaleim, Arun Prakash, Ayhan Irfanoglu, and Milutinović supplies a specific circuit proposal for the numerical branch of the roadmap [52]. Dan Shechtman's discovery of quasicrystals remains a methodological inspiration for examining structural assumptions. The relevance is not a direct physical analogy between crystallography and computer architecture, but the lesson that a successful prevailing structural model should not be mistaken for the full set of physically meaningful possibilities. MKA carries that attitude into architecture by treating topology, locality, and substrate boundaries as hypotheses to be measured and challenged rather than inherited constraints [85].

The next phase should connect the reference implementation to measured platforms and independent application studies. A shared representation of work and state, calibrated cost adapters, and reproducible comparisons would let researchers improve different parts of the system while testing their effect on the same completed workload. The long-term contribution sought is a cumulative science of architectural choice across substrates, with useful computation per energy and time budget as its common objective. We welcome the community to test and challenge it.

Declarations and Acknowledgements

No funding was received for this work. Code, synthetic benchmark inputs, selected plans, and complete plot data accompany the manuscript. The authors state that the manuscript is unpublished and is not under review at another journal or conference.

V.M. originated the Honeycomb research direction and provided senior scientific guidance. A.K. developed the six-regime roadmap and its industrial framing. D.K. led the MKA measurement framework, analytical development, and manuscript preparation. E.K. contributed figure preparation, presentation, and supporting research. The Milutinović and Katulevskiy families contributed equally to this work.

We are especially grateful to Professor John L. Hennessy for the intellectual influence of MIPS, VLSI architecture, technology–architecture analysis with Norman Jouppi, Stanford FLASH, and post-Moore work with David Patterson [10]–[14]. Their quantitative, workload-based approach guides our evaluation.

We thank Professor Mor Harchol-Balter of Carnegie Mellon University for her encouragement and the scheduling, server-farm power, and capacity-management research informing our service analysis [68], [70], [71], [75]. She met with the authors, shared and discussed prior findings, and encouraged them to develop and write this direction.

We acknowledge Professor Dimitris Bertsimas of MIT for robust optimization and contextual decision methods [44], [45], [47], which inform the uncertain-demand and budgeted-capacity analysis.

We acknowledge Professor Daniela Rus of MIT for decentralized reconfiguration, graph grammars, and joint design–control search [48]–[50]. Their application to MKA's generation and verification is developed by the authors.

We thank Oskar Mencer for dataflow research; Borko Furht and Dragana Milutinović for the parent Honeycomb analysis; and Osama Abdelaleim, Arun Prakash, and Ayhan Irfanoglu for the analog-solver work with V.M. Discussions of Dan Shechtman's quasicrystal work, whose discovery challenged long-standing assumptions about permissible order in solids, encouraged us to re-examine inherited structural assumptions in computing and to treat regularity and topology as experimentally testable design choices rather than fixed conventions [85].

Professors Hennessy, Harchol-Balter, Bertsimas, and Rus did not participate in preparing the manuscript and have not reviewed the final text. These acknowledgments recognize intellectual influence and encouragement without implying review or endorsement.

References

[1] J. L. Hennessy and D. A. Patterson, *Computer Architecture: A Quantitative Approach*, 6th ed. Cambridge, MA, USA: Morgan Kaufmann, 2017, ISBN: 978-0-12-811905-1.

[2] M. Horowitz, "Computing's energy problem (and what we can do about it)," in *2014 IEEE Int. Solid-State Circuits Conf. Digest of Technical Papers*, pp. 10–14, 2014, doi: 10.1109/ISSCC.2014.6757323.

[3] G. E. Moore, "Cramming more components onto integrated circuits," *Electronics*, vol. 38, no. 8, pp. 114–117, 1965.

[4] T. Dao, D. Y. Fu, S. Ermon, A. Rudra, and C. Ré, "FlashAttention: Fast and memory-efficient exact attention with IO-awareness," in *Advances in Neural Information Processing Systems*, vol. 35, 2022, pp. 16344–16359.

[5] W. Kwon, Z. Li, S. Zhuang, Y. Sheng, L. Zheng, C. H. Yu, J. E. Gonzalez, H. Zhang, and I. Stoica, "Efficient memory management for large language model serving with PagedAttention," in *Proc. 29th ACM Symp. Operating Systems Principles (SOSP)*, 2023, pp. 611–626, doi: 10.1145/3600006.3613165.

[6] D. Milutinović, V. Milutinović, and B. Soucek, "The Honeycomb Architecture," *Computer*, vol. 20, no. 4, pp. 81–83, Apr. 1987.

[7] V. Milutinović, "Mapping of neural networks on the honeycomb architecture," *Proc. IEEE*, vol. 77, no. 12, pp. 1875–1878, Dec. 1989.

[8] J. E. Hoch, *The Honeycomb Architecture: Prototype Analysis and Design*, Purdue University School of Electrical Engineering Technical Report TR-EE-87-28, Dec. 1987.

[9] V. Milutinović, D. Milutinović, and B. Furht, "On mapping of topological structures and mathematical algorithms onto specialized architectures for graph-based computing," *Journal of Big Data*, vol. 13, art. 34, 2026, doi: 10.1186/s40537-025-01326-z.

[10] J. L. Hennessy, N. P. Jouppi, S. Przybylski, C. Rowen, T. Gross, F. Baskett, and J. Gill, "MIPS: A microprocessor architecture," in *Proc. 15th Annual Symp. Microprogramming (MICRO)*, 1982, pp. 17–22.

[11] J. L. Hennessy, "VLSI processor architecture," *IEEE Transactions on Computers*, vol. C-33, no. 12, pp. 1221–1246, Dec. 1984.

[12] J. L. Hennessy and N. P. Jouppi, "Computer technology and architecture: An evolving interaction," *Computer*, vol. 24, no. 9, pp. 18–29, Sept. 1991, doi: 10.1109/2.84896.

[13] M. Heinrich, D. Ofelt, M. A. Horowitz, and J. L. Hennessy, "Hardware/software co-design of the Stanford FLASH multiprocessor," *Proceedings of the IEEE*, vol. 85, no. 3, pp. 455–466, Mar. 1997, doi: 10.1109/5.558720.

[14] J. L. Hennessy and D. A. Patterson, "A new golden age for computer architecture," *Communications of the ACM*, vol. 62, no. 2, pp. 48–60, Feb. 2019, doi: 10.1145/3282307.

[15] D. Oriato, S. Girdlestone, and O. Mencer, "Dataflow computing in extreme performance conditions," *Advances in Computers*, vol. 96, pp. 105–137, 2015, doi: 10.1016/bs.adcom.2014.11.002.

[16] N. Voss, B. Kwaadgras, O. Mencer, W. Luk, and G. Gaydadjiev, "On predictable reconfigurable system design," *ACM Transactions on Architecture and Code Optimization*, vol. 18, no. 2, art. 17, 2021, doi: 10.1145/3436995.

[17] N. Voss, M. Bacis, O. Mencer, G. Gaydadjiev, and W. Luk, "Convolutional neural networks on dataflow engines," in *Proc. 35th IEEE International Conference on Computer Design*, 2017, pp. 435–438, doi: 10.1109/ICCD.2017.77.

[18] A. Parashar et al., "Timeloop: A systematic approach to DNN accelerator evaluation," in *IEEE ISPASS*, 2019, pp. 304–315, doi: 10.1109/ISPASS.2019.00042.

[19] H. Kwon, P. Chatarasi, M. Pellauer, A. Parashar, V. Sarkar, and T. Krishna, "Understanding reuse, performance, and hardware cost of DNN dataflow: A data-centric approach," in *Proc. MICRO*, 2019, pp. 754–768, doi: 10.1145/3352460.3358252.

[20] Y. N. Wu, J. S. Emer, and V. Sze, "Accelergy: An architecture-level energy estimation methodology for accelerator designs," in *IEEE/ACM ICCAD*, 2019, pp. 1–8, doi: 10.1109/ICCAD45719.2019.8942149.

[21] T. Andrulis, J. S. Emer, and V. Sze, "CiMLoop: A flexible, accurate, and fast compute-in-memory modeling tool," in *IEEE ISPASS*, 2024, pp. 10–23, doi: 10.1109/ISPASS61541.2024.00012.

[22] J. Ragan-Kelley, A. Adams, S. Paris, M. Levoy, S. Amarasinghe, and F. Durand, "Decoupling algorithms from schedules for easy optimization of image processing pipelines," *ACM Trans. Graph.*, vol. 31, no. 4, art. 32, 2012, doi: 10.1145/2185520.2185528.

[23] M. Bansal, D. Sharlet, J. Ragan-Kelley, and S. Amarasinghe, "Lightweight and locality-aware composition of black-box subroutines," *Proc. ACM Program. Lang.*, vol. 9, PLDI, art. 189, pp. 1043–1067, 2025, doi: 10.1145/3729292.

[24] IEEE, "Systems and Architectures," *International Roadmap for Devices and Systems*, 2024 ed. [Online]. Available: https://irds.ieee.org/images/files/pdf/2024/2024IRDS_SA.pdf.

[25] E. Q. V. Martins, "On a multicriteria shortest path problem," *European Journal of Operational Research*, vol. 16, no. 2, pp. 236–245, 1984, doi: 10.1016/0377-2217(84)90077-8.

[26] G. M. Amdahl, "Validity of the single processor approach to achieving large scale computing capabilities," in *Proc. AFIPS Spring Joint Computer Conf.*, 1967, pp. 483–485, doi: 10.1145/1465482.1465560.

[27] R. Gonzalez and M. Horowitz, "Energy dissipation in general purpose microprocessors," *IEEE Journal of Solid-State Circuits*, vol. 31, no. 9, pp. 1277–1284, Sept. 1996, doi: 10.1109/4.535411.

[28] A. Vaswani et al., "Attention is all you need," in *Advances in Neural Information Processing Systems*, vol. 30, 2017.

[29] J. Ainslie, J. Lee-Thorp, M. de Jong, Y. Zemlyanskiy, F. Lebron, and S. Sanghai, "GQA: Training generalized multi-query transformer models from multi-head checkpoints," in *Proc. EMNLP*, 2023, pp. 4895–4901, doi: 10.18653/v1/2023.emnlp-main.298.

[30] N. Shazeer, "Fast transformer decoding: One write-head is all you need," arXiv:1911.02150, 2019.

[31] N. Shazeer, "GLU variants improve Transformer," arXiv:2002.05202, 2020.

[32] H. Touvron et al., "Llama 2: Open foundation and fine-tuned chat models," arXiv:2307.09288, 2023.

[33] M. Shoeybi et al., "Megatron-LM: Training multi-billion parameter language models using model parallelism," arXiv:1909.08053, 2019.

[34] D. Narayanan et al., "Efficient large-scale language model training on GPU clusters using Megatron-LM," in *Proc. Int. Conf. High Performance Computing, Networking, Storage and Analysis (SC)*, 2021, doi: 10.1145/3458817.3476209.

[35] W. Fedus, B. Zoph, and N. Shazeer, "Switch Transformers: Scaling to trillion parameter models with simple and efficient sparsity," *JMLR*, vol. 23, no. 120, 2022.

[36] G.-I. Yu, J. S. Jeong, G.-W. Kim, S. Kim, and B.-G. Chun, "Orca: A distributed serving system for transformer-based generative models," in *16th USENIX Symp. Operating Systems Design and Implementation*, 2022, pp. 521–538.

[37] F. T. Leighton, B. M. Maggs, and S. B. Rao, "Packet routing and job-shop scheduling in $O(\text{congestion} + \text{dilation})$ steps," *Combinatorica*, vol. 14, no. 2, pp. 167–186, 1994, doi: 10.1007/BF01215349.

[38] C. D. Thompson, "Area–time complexity for VLSI," in *Proc. 11th ACM Symp. Theory of Computing*, 1979, pp. 81–88, doi: 10.1145/800135.804401.

[39] F. R. K. Chung, *Spectral Graph Theory*. Providence, RI, USA: American Mathematical Society, 1997.

[40] G. Karypis and V. Kumar, "A fast and high quality multilevel scheme for partitioning irregular graphs," *SIAM Journal on Scientific Computing*, vol. 20, no. 1, pp. 359–392, 1998, doi: 10.1137/S1064827595287997.

[41] B. S. Landman and R. L. Russo, "On a pin versus block relationship for partitions of logic graphs," *IEEE Trans. Comput.*, vol. C-20, no. 12, pp. 1469–1479, 1971, doi: 10.1109/T-C.1971.223159.

[42] S. Williams, A. Waterman, and D. A. Patterson, "Roofline: An insightful visual performance model for multicore architectures," *Communications of the ACM*, vol. 52, no. 4, pp. 65–76, 2009, doi: 10.1145/1498765.1498785.

[43] J.-W. Hong and H. T. Kung, "I/O complexity: The red–blue pebble game," in *Proc. 13th ACM Symp. Theory of Computing*, 1981, pp. 326–333, doi: 10.1145/800076.802486.

[44] D. Bertsimas and M. Sim, "The price of robustness," *Operations Research*, vol. 52, no. 1, pp. 35–53, 2004, doi: 10.1287/opre.1030.0065.

[45] D. Bertsimas, D. B. Brown, and C. Caramanis, "Theory and applications of robust optimization," *SIAM Review*, vol. 53, no. 3, pp. 464–501, 2011, doi: 10.1137/080734510.

[46] D. Bertsimas, D. den Hertog, and J. Pauphilet, "Probabilistic guarantees in robust optimization," *SIAM Journal on Optimization*, vol. 31, no. 4, pp. 2893–2920, 2021, doi: 10.1137/21M1390967.

[47] D. Bertsimas and N. Kallus, "From predictive to prescriptive analytics," *Management Science*, vol. 66, no. 3, pp. 1025–1044, 2020, doi: 10.1287/mnsc.2018.3253.

[48] A. Zhao, J. Xu, M. Konaković Luković, J. Hughes, A. Spielberg, D. Rus, and W. Matusik, "RoboGrammar: Graph grammar for terrain-optimized robot design," *ACM Transactions on Graphics*, vol. 39, no. 6, art. 188, pp. 1–16, 2020, doi: 10.1145/3414685.3417831.

[49] J. Xu, A. Spielberg, A. Zhao, D. Rus, and W. Matusik, "Multi-objective graph heuristic search for terrestrial robot design," in *Proc. IEEE International Conference on Robotics and Automation (ICRA)*, 2021, doi: 10.1109/ICRA48506.2021.9561818.

[50] Z. Butler, K. Kotay, D. Rus, and K. Tomita, "Generic decentralized control for lattice-based self-reconfigurable robots," *The International Journal of Robotics Research*, vol. 23, no. 9, pp. 919–937, 2004, doi: 10.1177/0278364904044409.

[51] S. Ambrogio et al., "An analog-AI chip for energy-efficient speech recognition and transcription," *Nature*, vol. 620, pp. 768–775, 2023, doi: 10.1038/s41586-023-06337-5.

[52] O. Abdelaleim, A. Prakash, A. Irfanoglu, and V. Milutinović, "Analog solver circuit for linear symmetric positive-definite systems at a complexity independent of matrix size," arXiv:2508.04609v2, 24 Jan. 2026. Preprint, doi: 10.48550/arXiv.2508.04609.

[53] N. J. Higham, *Accuracy and Stability of Numerical Algorithms*, 2nd ed. Philadelphia, PA, USA: SIAM, 2002, doi: 10.1137/1.9780898718027.

[54] J. H. Kim et al., "Aquabolt-XL HBM2-PIM, LPDDR5-PIM with in-memory processing, and AXDIMM with acceleration buffer," *IEEE Micro*, vol. 42, no. 3, pp. 20–30, 2022, doi: 10.1109/MM.2022.3164651.

[55] D. A. B. Miller, "Attojoule optoelectronics for low-energy information processing and communications," *Journal of Lightwave Technology*, vol. 35, no. 3, pp. 346–396, 2017, doi: 10.1109/JLT.2017.2647779.

[56] S. Daudlin et al., "Three-dimensional photonic integration for ultra-low-energy, high-bandwidth interchip data links," *Nature Photonics*, vol. 19, pp. 502–509, 2025, doi: 10.1038/s41566-025-01633-0.

[57] D. Loss and D. P. DiVincenzo, "Quantum computation with quantum dots," *Phys. Rev. A*, vol. 57, pp. 120–126, 1998.

[58] R. Hanson et al., "Spins in few-electron quantum dots," *Rev. Mod. Phys.*, vol. 79, pp. 1217–1265, 2007.

[59] F. Borsoi et al., "Shared control of a 16 semiconductor quantum dot crossbar array," *Nature Nanotechnology*, vol. 19, pp. 21–27, 2024, doi: 10.1038/s41565-023-01491-3.

[60] C. H. Bennett, G. Brassard, C. Crépeau, R. Jozsa, A. Peres, and W. K. Wootters, "Teleporting an unknown quantum state via dual classical and EPR channels," *Phys. Rev. Lett.*, vol. 70, pp. 1895–1899, 1993.

[61] S. Wehner, D. Elkouss, and R. Hanson, "Quantum internet: A vision for the road ahead," *Science*, vol. 362, eaam9288, 2018.

[62] K. Azuma et al., "Quantum repeaters: From quantum networks to the quantum internet," *Rev. Mod. Phys.*, vol. 95, 045006, 2023.

[63] R. H. Walden, "Analog-to-digital converter survey and analysis," *IEEE Journal on Selected Areas in Communications*, vol. 17, no. 4, pp. 539–550, Apr. 1999, doi: 10.1109/49.761034.

[64] B. Murmann, "The race for the extra decibel: A brief review of current ADC performance trajectories," *IEEE Solid-State Circuits Magazine*, vol. 7, no. 3, pp. 58–66, 2015, doi: 10.1109/MSSC.2015.2442393.

[65] W. Huang, S. Ghosh, S. Velusamy, K. Sankaranarayanan, K. Skadron, and M. R. Stan, "HotSpot: A compact thermal modeling methodology for early-stage VLSI design," *IEEE Transactions on Very Large Scale Integration Systems*, vol. 14, no. 5, pp. 501–513, May 2006, doi: 10.1109/TVLSI.2006.876103.

[66] R. Landauer, "Irreversibility and heat generation in the computing process," *IBM J. Res. Dev.*, vol. 5, no. 3, pp. 183–191, 1961.

[67] C. H. Bennett, "Logical reversibility of computation," *IBM J. Res. Dev.*, vol. 17, no. 6, pp. 525–532, 1973.

[68] M. Harchol-Balter, *Performance Modeling and Design of Computer Systems: Queueing Theory in Action*. Cambridge, U.K.: Cambridge University Press, 2013, doi: 10.1017/CBO9781139226424.

[69] M. Harchol-Balter, "Open problems in queueing theory inspired by datacenter computing," *Queueing Systems*, vol. 97, pp. 3–37, 2021, doi: 10.1007/s11134-020-09684-6.

[70] A. Gandhi, M. Harchol-Balter, R. Das, and C. Lefurgy, "Optimal power allocation in server farms," in *Proc. ACM SIGMETRICS/Performance*, 2009, pp. 157–168, doi: 10.1145/1555349.1555368.

[71] A. Gandhi, V. Gupta, M. Harchol-Balter, and M. A. Kozuch, "Optimality analysis of energy-performance tradeoff for server farm management," *Performance Evaluation*, vol. 67, no. 11, pp. 1155–1171, 2010, doi: 10.1016/j.peva.2010.08.009.

[72] M. Harchol-Balter, B. Schroeder, N. Bansal, and M. Agrawal, "Size-based scheduling to improve Web performance," *ACM Transactions on Computer Systems*, vol. 21, no. 2, pp. 207–233, May 2003, doi: 10.1145/762483.762486.

[73] Z. Scully, M. Harchol-Balter, and A. Scheller-Wolf, "SOAP: One clean analysis of all age-based scheduling policies," *Proceedings of the ACM on Measurement and Analysis of Computing Systems*, vol. 2, no. 1, art. 16, Mar. 2018, doi: 10.1145/3179419.

[74] I. Grosof, Z. Scully, M. Harchol-Balter, and A. Scheller-Wolf, "Optimal scheduling in the multiserver-job model under heavy traffic," *Proceedings of the ACM on Measurement and Analysis of Computing Systems*, vol. 6, no. 3, art. 51, Dec. 2022, doi: 10.1145/3570612.

[75] A. Gandhi, M. Harchol-Balter, R. Raghunathan, and M. A. Kozuch, "AutoScale: Dynamic, robust capacity management for multi-tier data centers," *ACM Transactions on Computer Systems*, vol. 30, no. 4, art. 14, Nov. 2012, doi: 10.1145/2382553.2382556.

[76] J. K. Williams, M. Harchol-Balter, and W. Wang, "The M/M/k with deterministic setup times," *Proceedings of the ACM on Measurement and Analysis of Computing Systems*, vol. 6, no. 3, art. 56, Dec. 2022, doi: 10.1145/3570617.

[77] Meta, "Llama 4 Maverick 17B-128E Instruct model card," Apr. 2025. [Online]. Available: https://huggingface.co/meta-llama/Llama-4-Maverick-17B-128E-Instruct.

[78] Z.ai, "GLM-5.3 model card and GLM-5 series architecture," 2026. [Online]. Available: https://huggingface.co/zai-org/GLM-5.3; https://github.com/zai-org/GLM-5.

[79] Moonshot AI, "Kimi K3 model card," 2026. [Online]. Available: https://huggingface.co/moonshotai/Kimi-K3.

[80] Qwen Team, "Qwen3.8-2.4T-A95B model card," Aug. 2026. [Online]. Available: https://huggingface.co/Qwen/Qwen3.8-2.4T-A95B.

[81] E. Garrison et al., "Variation graph toolkit improves read mapping by representing genetic variation in the reference," *Nature Biotechnology*, vol. 36, pp. 875–879, 2018, doi: 10.1038/nbt.4227.

[82] F. R. Kschischang, B. J. Frey, and H.-A. Loeliger, "Factor graphs and the sum-product algorithm," *IEEE Trans. Inf. Theory*, vol. 47, no. 2, pp. 498–519, 2001, doi: 10.1109/18.910572.

[83] W. A. Borders et al., "Integer factorization using stochastic magnetic tunnel junctions," *Nature*, vol. 573, pp. 390–393, 2019, doi: 10.1038/s41586-019-1557-9.

[84] C. E. Leiserson et al., "There's plenty of room at the top: What will drive computer performance after Moore's law?" *Science*, vol. 368, no. 6495, eaam9744, 2020, doi: 10.1126/science.aam9744.

[85] D. Shechtman, "Quasi-periodic materials: A paradigm shift in crystallography," Nobel Lecture, Stockholm University, 8 Dec. 2011.

AUTHOR BIOGRAPHIES

Daniil Katulevskiy is a computer scientist and systems researcher working on artificial intelligence infrastructure, heterogeneous computing, distributed inference, accelerator performance, edge computing, and post-Moore computer architecture. He graduated in Computer Science from the University of California, San Diego. His technical work includes AI and performance engineering on Qualcomm and NVIDIA computing platforms, with particular emphasis on memory movement, accelerator utilization, model serving, and system-level efficiency. Daniil is the lead author and primary technical contributor to the present formulation of the Milutinović–Katulevskiy Architecture (MKA). His contributions include the completed-work measurement framework and MKA progress criterion, the workload and admissibility model, graph placement and communication analysis, the exact reference-planning formulation, heterogeneous-substrate break-even models, large-model serving analysis, uncertainty-aware validation, and the connection between system-level efficiency and datacenter economics. His broader research interests include scalable AI systems, secure and privacy-preserving computing, and new computational architectures that combine hardware, software, communication, and physical constraints within a common systems framework.

Elizaveta Katulevskaya is a researcher with interests in quantitative analysis, artificial intelligence, applied mathematics, and interdisciplinary scientific communication. She has studied through Stanford University Summer Session and Letovo School. Within the MKA research program, Elizaveta contributed to the supporting research base, scientific presentation, visualization, and organization of evidence across multiple computing substrates and application domains. Her work on the project has focused especially on making complex architectural comparisons interpretable across disciplines and on supporting the presentation of quantitative results.

Aleksey Katulevskiy is a researcher and entrepreneur working at the intersection of artificial intelligence, financial infrastructure, digital assets, and emerging technology. He is affiliated with the Polity Research Initiative on Applied Artificial Intelligence in Regulated Web3 Finance, where his work concerns the application of AI and computational infrastructure to regulated financial and digital-asset systems. Within the MKA program, Aleksey was a major contributor to the development of the six-regime substrate roadmap, the industrial and deployment strategy, the cross-sector application framework, and the economic interpretation of architectural efficiency. His broader interests include AI-enabled financial infrastructure, large-scale digital systems, technology commercialization, and the institutional deployment of advanced computing in regulated environments.

Veljko Milutinović is a computer architect whose research spans microprocessor and VLSI design, parallel and dataflow computing, graph-based architectures, and energy-efficient high-performance computation. He received his Ph.D. from the University of Belgrade and spent a substantial part of his early academic career at Purdue University before continuing his research and teaching through later appointments in Europe and the United States. He is a Life Fellow of the IEEE, a member of Academia Europaea, a member of the Serbian Academy of Engineering Sciences, and a Foreign Member of the Montenegrin Academy of Sciences and Arts. His earlier work includes the co-design of pioneering DARPA-supported GaAs computing systems and the development of graph-oriented and dataflow architectures, while his later research has emphasized energy-efficient computing, algorithm-to-hardware mapping, and new paradigms for high-performance computation.

Veljko originated the Honeycomb research direction that provides the architectural lineage for MKA. That line includes the 1987 *Computer* Honeycomb architecture paper with Dragana Milutinović and Branislav Souček, his 1989 *Proceedings of the IEEE* work on mapping neural networks onto Honeycomb, James E. Hoch's Purdue prototype analysis conducted under Veljko's supervision, and later work extending Honeycomb concepts to the mapping of graphs and mathematical algorithms. In the present MKA program, Veljko provides the senior scientific direction and the historical architectural foundation from which the six-regime roadmap develops.

Correspondence:
dkatulevskiy@gmail.com
elkatul@stanford.edu
alexkatul73@gmail.com
vm@etf.rs

Students' Awareness, Perceptions, and Use of Generative AI Tools in Higher Education

Radaković, Davorka; Steingartner, William; and Seleši, Dora

Abstract: ***The integration of digital technologies, content and processes is reshaping all sectors of education, from curriculum reform to teacher assessment and development. Among these technologies, generative artificial intelligence (AI) presents significant opportunities to improve the learning experience, especially in secondary and higher education, but it also raises questions about accuracy, academic integrity, privacy, and institutional guidance. This study examines how students perceive and use AI as a learning tool and how AI applications are integrated into academic processes and learning activities. A structured questionnaire was administered in two phases to students from five universities in four countries; the analysis dataset contained 227 consenting respondents. We report item-specific denominators, confidence intervals for key proportions, nonparametric comparisons for ordinal outcomes, effect sizes, Holm correction for multiple comparisons, and sensitivity analyses adjusted for recruitment phase and institution. Most respondents reported prior AI use, frequent use, and positive learning perceptions, while almost all users had encountered incorrect output. Only a minority knew that institutional AI-use guidelines existed, although most supported formal policies and explicit ethical-use instruction. Gender differences were generally small after multiplicity correction; women more often reported grammar/writing tools and problem-solving use. The findings support clearer institutional guidance, AI-literacy training, verification practices, and responsible-use policies rather than simple promotion or prohibition of AI tools.***

Index Terms: ***AI, ChatGPT, Generative AI, Higher Education, Teaching and Learning, Student Engagement***

1. Introduction

Over the past decades, the rapid and unprecedented growth of technologies applied in education and various other sectors has been clearly evident [20, 22, 43]. Digital technologies, digital content, and related processes exert a significant influence across all sectors of education—including primary and secondary schooling, higher education, and informal learning environments—shaping the entire educational value chain, from curriculum reform and instructional practices to assessment and teacher professional development, while actively engaging educators, learners, and institutional leaders.

Applications based on Artificial Intelligence have a huge contribution in everybody's daily life. In addition, AI (based on a model) offers a wide range of potential benefits for students and teachers. AI in education poses challenges, demanding research and practical application [12]. Effective data use, ethics, and AI advancement require skill development for teachers, administrators, and students.

Manuscript received September 10, 2026.

This work was supported by KEGA project 028TUKE-4/2026 – "Innovative approaches to teaching and researching formal methods in the context of modern software engineering", granted by the Cultural and Education Grant Agency of the Slovak Ministry of Education; ALFA Labs – trAnsformation pLan For Ai Living Labs, and CZ-1503-05-2425 (Umbrella) – Development of Computational Thinking. The authors gratefully acknowledge the financial support of the Ministry of Science, Technological Development and Innovation of the Republic of Serbia (Grants No. 451-03-33/2026-03/ 200125 & 451-03-34/2026-03/ 200125).

D. Radaković and D. Seleši are with the University of Novi Sad, Faculty of Sciences, Department of Mathematics and Informatics, Novi Sad, Serbia (e-mail: davorkar@dmi.uns.ac.rs, dora@dmi.uns.ac.rs).

W. Steingartner is with the Faculty of Electrical Engineering and Informatics, Technical University of Košice, Košice, Slovakia william.steingartner@tuke.sk).

Currently one of the most popular and widely used tools, ChatGPT, can assist students in: developing communication skills in a foreign language [13, 18]; in programming [39]; creating content, answering questions, explaining concepts, and giving suggestions [19, 49]; assist in conducting literature reviews [26]. Notably, more than a half of college students have used AI on assignments or exams [35].

Researchers are dedicated to developing educational visualization tools and studying their impact on the teaching process [27, 41, 42]. Since the education sector has benefited enormously through the integration of tools and platforms driven by digital technology, many authors analyze modern AI tools available to learners and educators [8, 9, 14, 18, 29, 31, 34, 38]. Further, the European Education Area (EEA) initiative [1] aims to strengthen national education systems by promoting resilience, inclusiveness, and the integration of digital learning and pedagogies, while supporting policymakers in designing, implementing, and evaluating initiatives related to digital education.

The primary objective of this study is to examine how AI can impact on teaching and learning in secondary and higher education. This study, therefore, aimed to examine students' attitudes toward AI through a cross-sectional analysis. Our research questions focus on students' access to AI tools, their perception of AI in education, the perceived usefulness of AI tools for learning, and potential gender-related differences in these perceptions.

We investigate the following research questions:

RQ1 What access, familiarity, use frequency, tools, and academic-task purposes do students report?

RQ2 How do students evaluate AI's learning effectiveness and support for independent learning, and what do they report about trust and incorrect output?

RQ3 What relationships connect AI-use intensity and purpose, complete-task generation, integrity attitudes, treatment preferences, detection awareness, and institutional governance?

RQ4 Which gender differences in AI access, use, learning perceptions, and responsible-use outcomes are supported after effect-size estimation, multiplicity correction, and sensitivity adjustment for recruitment phase and institution?

This paper is a significant extension and improvement of our preliminary work [44], in the following ways:

- We included students from Serbia and Slovenia in the second round of the study;
- We expanded the Results section;
- We added a Discussion section and Supplementary Material section.

The study that was inspired by research conducted by [7].

The main contributions of the article are as follows:

- providing an international empirical analysis of students' awareness, perception and use of AI tools in higher education, based on data collected from universities in four European countries;
- identifying patterns in how students integrate generative AI tools into their learning activities, including programming, problem-solving and academic writing;
- exploring gender differences in students' perceptions and use of AI tools in educational contexts;
- highlighting implications for educators and institutions regarding the responsible and effective integration of AI tools into teaching and learning.

The rest of this paper is organized as follows. Section 2 reviews the relevant literature. Section 3 describes the methodology. Section 4 presents the results and answers to research questions. Section 5 discusses the findings: general awareness and usage of AI tools; ethics, policy, and responsible use; and gender-related patterns. Finally, Section 6 concludes the paper and outlines directions for future research.

2. Related Work

This paper examines the use of AI tools in higher education and analyzes their potential impact on teaching and learning. The literature shows that researchers have extensively examined the transformative role of AI in teaching and learning.

As educational institutions increasingly rely on digital platforms and online learning systems, students and teachers increasingly interact through various online communication tools and social networks [15]. In such environments, similar security and privacy challenges arise and must be addressed to safeguard sensitive educational data.

The EU increasingly emphasizes digital transformation and the use of AI in education, as reflected in the AI Act and the Coordinated Plan on AI. One of the European Commission framework projects is SELFIE [3] (Self-reflection on Effective Learning by Fostering the use of Innovative Educational technologies). Digital learning technologies, in the context of DigCompOrg [2], constitute a key enabler for educational organizations, which can support their efforts to achieve their particular mission and vision for quality education.

Recent research has increasingly focused on the use of generative AI tools, particularly ChatGPT, in higher education. Several review studies summarize the opportunities and challenges associated with the use of ChatGPT in educational contexts [5]. Early reviews of ChatGPT in education highlight both the opportunities and the challenges associated with its use in educational contexts [21].

Qualitative research [23] provides an overview of the effectiveness of using ChatGPT as a virtual assistant. It helps improve student productivity in higher education by providing useful information and resources, supporting language development, facilitating collaboration, improving time efficiency, and offering motivation and support. Baidoo-Anu and Ansah discuss in [7] the benefits of ChatGPT and related generative AI technologies for advancing teaching and learning. They also identify several potential drawbacks, such as generating incorrect information, biases in training data that may reinforce existing biases, and privacy concerns. In addition, the study [33] emphasizes the potential of ChatGPT as a tool for enhancing English as a Foreign Language students' English language proficiency.

Some studies report less positive outcomes. For example, [32] found that ChatGPT did not improve performance across several disciplines, while Wu [50] warns that rapid automated answers may create an illusion of understanding. In addition, Haman [26] shows that ChatGPT may generate inaccurate or even non-existent references when used for literature searches, raising concerns about its reliability in academic research.

Currently, there are not many studies that observe using ChatGPT for teaching and learning in data science education, so paper [51] incorporates ChatGPT into a data science course and collects valuable insights from students, as well as sharing feedback from the instructor. The findings highlight the uniqueness of data science education while revealing new opportunities and challenges in integrating ChatGPT into the curriculum. Bin-Nashwan et al. [10] suggest to educators, policymakers, and technology developers that they have to work collaboratively to establish frameworks that promote responsible and ethical use of ChatGPT in education.

Focus on ChatGPT is also given through analysis of the significance and utilization of artificial intelligence in learning systems in higher STEM education [28]. In addition, the study [46] found that students viewed ChatGPT as an innovative set of AI tools that can be used in classroom settings, including programming and computer languages, promoting sustainable adoption of AI in education. In addition, the study [37] showed that the integration of GPT and innovative learning methods significantly improved academic performance and motivation.

Chen et al. [17] systematically reviews AI applications in education, including personalized learning, intelligent tutoring systems, and automated assessment. It also addresses technical, ethical, and pedagogical challenges, emphasizing the need for balanced integration of AI with human-centered teaching approaches.

Another overview of AI's integration into educational settings is given in [36]. Ng et al. discuss various applications such as intelligent tutoring systems, personalized learning platforms, and automated assessment tools, highlighting their potential to improve educational outcomes. Likewise, Lameras et al. [30] discuss the evolution of AI applications: from early computer-based tools to advanced systems such as intelligent tutoring, adaptive learning platforms, and humanoid robots, highlighting their impact on administrative efficiency, personalized instruction, and improved learning outcomes. Together, these studies demonstrate the growing role of AI technologies in shaping modern educational environments.

In addition, the review [31] categorizes AI applications in education into proactive (e.g. admissions, course scheduling, content generation) and reactive (e.g. knowledge delivery, performance assessment, outcome prediction) phases. It examines research trends over two decades, highlighting how AI methods support various educational stakeholders and discussing the evolution of solution approaches in the field. Besides that, Guo et al. in [25] examine teachers' acceptance of AI (TAAI) in educational settings. The instrument assesses five dimensions: perceived usefulness, perceived ease of use, behavioral intention, self-efficacy, and anxiety.

At the same time, the use of generative AI (GenAI) in teachers' daily practice offers substantial potential to enhance teaching, but also introduces new risks and responsibilities [11, 16, 40, 45]. The study [24] explores how AI simulations acting as both virtual students and mentors help preservice mathematics teachers develop teaching skills and professional identity through interactive practice and feedback.

Unlike the other reviews, Tan et ač. noticed and highlighted a gap in research [48] addressing the development needs of teachers as they integrate AI technologies into their teaching practices, emphasizing the need for future research to further explore the potential of AI in teacher professional development and to investigate how AI technologies can be applied in education from both the perspectives of student learning and teacher instruction.

Motivated by these findings, our study develops a questionnaire to examine students' awareness, perception, and usage of AI tools in teaching and learning, including ethical considerations.

3. Methodology

The research data in this study was collected through the primary data collection – a questionnaire [47]. Data were collected online using Google Forms [4] from students in two phases: first during the beginning of the summer semester of 2024/2025 (April 2025), and second during the end of the winter semester of 2025/2026 between January and February 2026. The study took a random sample of students with similar backgrounds: computer science, information technologies, AI, cybersecurity, mathematics and applied mathematics.

Data were therefore collected from student classes that were available to the researchers at the time from universities in Poland, Slovakia, Slovenia and Serbia. The questionnaire was written in English. Before taking part in the study, participants were informed of its general purpose and provided their informed consent.

The questionnaire includes closed-ended questions with binary response options (yes/no), Likert scales, check-boxes for multiple answers, and open questions with the opportunity for short written answers. This questionnaire design was intended to give a mix of both qualitative and quantitative data.

The AI tools questions used the phrase "AI tools" or "AI Assistants (Chatbots)" (e.g. "What AI tools have you used in an educational context?"), rather than to just emphasize "ChatGPT" since students use different AI tools.

The first section of the questionnaire gathered students' general demographic information such as age, gender, and educational status. The second section assessed perceived general perception, awareness, and usage of AI in students' field, while the third section measured attitude toward usage of AI tools. The fourth section observe the level of satisfaction while using them in teaching and learning, ethics, and confidence. Final part examines the consideration of the AI use for students who still have

not used AI.

The data corpus was analyzed using survey content analysis to identify themes and patterns, providing insights into the potential role of AI tools in education, and, descriptive statistics for the data collected (reported based on counts and ratios) is coded manually from Google spreadsheets and prominent themes identified. In addition, comparative analysis between male and female students was performed in order to examine potential gender-related differences in the perception and use of AI tools.

3.1 Participants

The study results presented in the article are obtained from a data sample of a total of 228 students from five universities: Cardinal Stefan Wyszyński University in Warsaw, Poland (42 students), Technical University of Košice, Slovakia (76 students), Matej Bel University, Banská Bystrica, Slovakia (11 students), University of Maribor, Faculty of Organizational Sciences, Kranj (10 students), University of Novi Sad (Faculty of Sciences, Faculty of Technical Sciences), Serbia (89 students).

In the first phase participated 84 students from two universities: Cardinal Stefan Wyszyński University in Warsaw, Poland (36 students), and Technical University of Košice, Slovakia (48 students). While, in the second phase participated 144 students from five universities: Cardinal Stefan Wyszyński University in Warsaw, Poland (6 students), Technical University of Košice, Slovakia (28 students), Matej Bel University, Banská Bystrica, Slovakia (11 students), University of Maribor, Faculty of Organizational Sciences, Kranj (10 students), University of Novi Sad (Faculty of Sciences, Faculty of Technical Sciences), Serbia (89 students). Detailed data related to the sample for each university is given in the Table 1; a visual summary of the unequal institutional composition is provided in Supplementary Fig. S2.

The highest level of education completed by 77.19% of the students is high school (pre-university entry, usually taken at age 18), 18.86% have bachelor's degree, and 3.95% have master's degree.

Table 1: Detailed data related to the sample for each university

Institution	n	%
University of Novi Sad, Serbia	88	38.8
Technical University of Košice, Slovakia	76	33.5
Cardinal Stefan Wyszyński University in Warsaw, Poland	42	18.5
Matej Bel University, Banská Bystrica, Slovakia	11	4.8
University of Maribor, Faculty of Organizational Sciences, Slovenia	10	4.4

70 students considered themselves female (30.70%), while 154 students identified as male (67.54%). Two students (0.88%) identified as non-binary/third gender, and two students (0.88%) preferred not to disclose their gender. There was some concern about the gender ratio; however, it is common in these types of the study. Table 2 presents the description of the participants by gender in this study (total and by Universities).

Table 2: Distribution of students by gender

Gender	n	%
Male	154	67.8
Female	69	30.4
Non-binary/third gender	2	0.9
Prefer not to say	2	0.9

Table 3: Distribution of students by age

Age range	n	%
18–20	99	43.6
21–23	93	41.0
24–26	15	6.6
27–30	13	5.7
31 or older	6	2.6
Prefer not to answer	1	0.4

In the present sample, the average age was 21.74, while the median age was 20 years (ranging from 18 to 46 years, $SD = 3.76$). Two students preferred not to disclose their age. Table 3 presents the distribution of the students by age range in this study.

25 students are major in mathematics, 5 in applied mathematics, 2 in teaching CS and mathematics, 55 in computer science, 114 in information technologies/informatics, 5 in cy-

bersecurity, 3 in AI, 11 in applied informatics, and 8 in management of information systems.

The results related to RQ1 indicate that 68.86% of all students consider that they have access to AI tools in their educational environment, 20.61% are not sure, while 10.53% consider that they do not have access. Correspondingly, similar ratio is between genders as shown in Table 4.

Table 4: Frequency of access to AI tools in your educational environment (by gender)

Response	Male	Female	All respondents
Yes	107	47	157
No	17	6	23
Not sure	31	16	47

4. Results

The survey questionnaire was crafted to gather in-depth insights from participants, including questions that explored the perceived benefits of using AI tools in learning and any challenges faced during their use.

4.1 General Awareness and Usage of AI tools

The survey was designed to explore the attitudes of students to the use of AI tools in their field and in their education. The revised analysis uses the respondent-level data consistently and reports denominators for each item. Access to AI tools in the educational environment was reported by 157 of 227 students (69.2%, 95% CI 62.9%–74.8%); 47 students were unsure and 23 reported no access. Moderate or extreme familiarity with AI tools was reported by 161 of 227 students (70.9%). Prior use of AI tools was reported by 214 of 227 students (94.3%, 95% CI 90.4%–96.6%).

Among the 214 AI users, 173 students (80.8%, 95% CI 75.0%–85.6%) used AI at least two to four times per week. ChatGPT was the most frequently reported assistant (197/214, 92.1%), followed by Gemini (97/214, 45.3%), Microsoft Copilot (67/214, 31.3%), Claude (48/214, 22.4%), and DeepSeek (20/214, 9.3%). Multiple tools could be selected. The leading task selections were solving problems (169/214, 79.0%), programming (150/214, 70.1%), teaching-related support (94/214, 43.9%), writing homework or tasks (63/214, 29.4%), and writing essays (51/214, 23.8%). These results confirm that the use of AI tools is not limited to one product or one task type.

A graphical version of the key AI-use, learning, integrity, and governance outcomes is provided in Supplementary Fig. S1.

To measure students' attitudes toward the effectiveness of AI in enhancing student learning outcomes, responses were obtained using a 7-point Likert scale. Among AI users, 168 of 214 respondents (78.5%) rated AI as at least slightly effective for learning. However, positive evaluations were accompanied by caution: 210 of 214 users (98.1%) reported encountering incorrect AI output, while 191 of 214 (89.3%) nevertheless reported at least some trust in AI tools. This combination suggests conditional reliance rather than uncritical confidence.

4.2 Ethics, Policy, and Responsible Use

The ethics questions show a clear governance gap. Only 53 of 214 AI users (24.8%) knew that their institution had AI-use guidelines, whereas 131 of 214 (61.2%) supported formal policies and 139 of 214 (65.0%) supported explicit teaching about ethical AI use. Nearly half of AI users (100/214, 46.7%) reported using AI to generate complete homework tasks, while 124 of 214 (57.9%) agreed or strongly agreed that AI-assisted dishonesty threatens educational integrity.

Students who reported complete-task generation gave lower integrity-threat ratings than students who did not in the unadjusted comparison, but this association was not retained in the adjusted dichotomized sensitivity model. Complete-task generation increased across the ordered use-frequency scale and was most common among daily users. In the adjusted model, programming and teaching purposes remained associated with complete-task generation. The association between guideline visibility and policy demand was also important: students who knew that guidelines existed more often supported formal policies than other respondents. The most frequent treatment preferences were

Table 5: Key learning and responsible-use outcomes

Outcome	n/N	%	95% CI
Prior AI use	214/227	94.3	90.4–96.6
Use at least 2–4 times weekly	173/214	80.8	75.0–85.6
At least slightly effective for learning	168/214	78.5	72.5–83.5
Somewhat/mostly trusts AI	191/214	89.3	84.4–92.7
Encountered incorrect output	210/214	98.1	95.3–99.3
Generated complete homework	100/214	46.7	40.2–53.4
Integrity threat: agree/strongly agree	124/214	57.9	51.2–64.4
Teach ethical use: yes	139/214	65.0	58.3–71.0
Institution has guidelines: yes	53/214	24.8	19.5–31.0
Formal policies should exist: yes	131/214	61.2	54.5–67.5

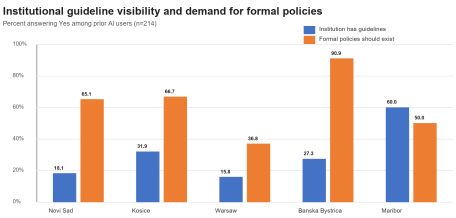


Figure 1: Institutional guideline visibility and demand for formal AI-use policies among prior AI users.

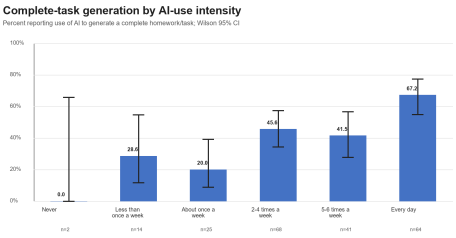


Figure 2: Complete-task generation by AI-use frequency, with Wilson 95% confidence intervals.

adaptation rather than punishment and recognizing a shift in learning and assessment. Detection awareness was not reliably associated with lower complete-task generation after within-family correction.

Figure 1 summarizes the most policy-relevant institutional result.

Figures 2 and 3 show the two ethical-behaviour results that are most directly actionable for teaching and policy design. Additional ethics analyses are retained in the supplementary material: the integrity-threat distribution by complete-task behaviour is shown in Supplementary Fig. S6, detection awareness and behaviour in Supplementary Fig. S7, and adjusted ethics associations in Supplementary Fig. S8.

4.3 Gender-related Patterns

The analysis aimed at evaluating RQ4 shows that some gender differences can be observed, but they are more limited than the descriptive figures alone suggest. Male and female respondents had similar access, use frequency, perceived learning effectiveness, and views of learner independence. None of the six primary ordinal comparisons remained significant after Holm correction. Women more often selected grammar and writing improvement tools and problem solving as a task, and these differences survived correction within their multiple-response families. Programming was more common among men before correction but not after correction. An adjusted sensitivity model suggested lower odds of moderate-or-higher familiarity among women, but this result depends on dichotomization and should be treated as exploratory. The detailed gender-stratified tool and task selections are shown in Supplementary Fig. S3, adjusted gender sensitivity models in Supplementary Fig. S4, and the primary ordinal gender comparisons in Supplementary Fig. S5. These figures are useful for transparency but are kept outside the main manuscript to preserve space and avoid overemphasizing exploratory subgroup findings.

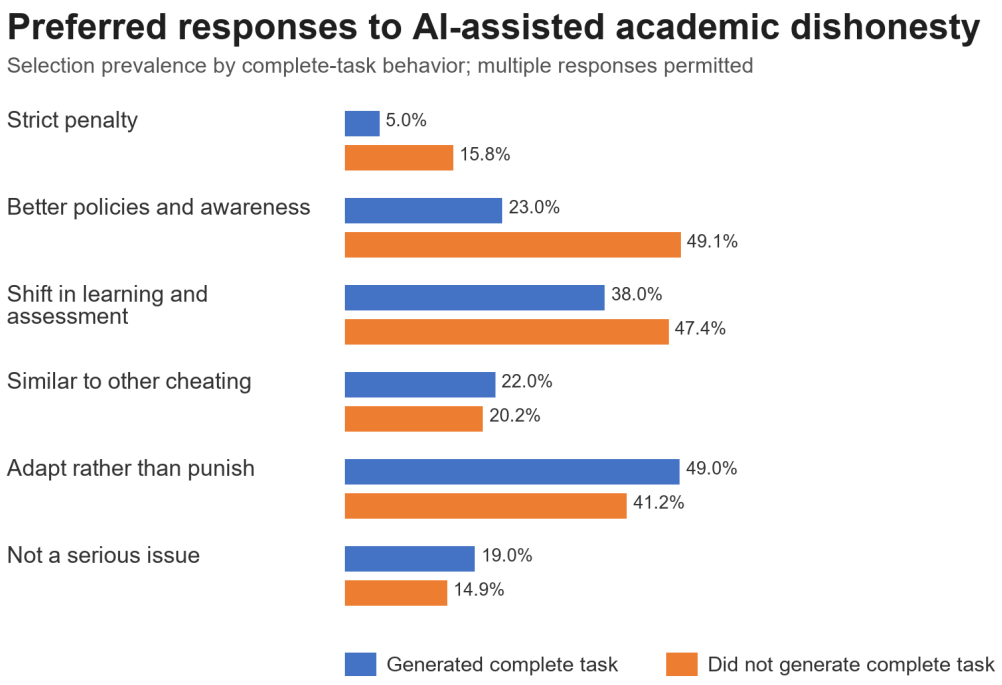


Figure 3: Preferred treatment of AI-assisted academic dishonesty; multiple selections were permitted.

5. Discussion

The results reached in this study bring the answers to the research questions presented in the Introduction. Findings regarding RQ1 show that most students in this technology-focused convenience sample have used AI tools and use them frequently for problem solving, programming, teaching-related support, homework or tasks, and writing. Regarding RQ2, students generally perceive AI as useful for learning, but this positive perception is accompanied by widespread experience of incorrect output. The results related to RQ3 show that ethical and institutional questions are not peripheral: many students report complete-task generation, most support ethical-use instruction, and many do not know whether institutional guidelines exist. RQ4 shows that gender-related differences are present in selected tool and task choices, but broad claims about gender differences are not supported after correction for multiple comparisons.

The use of technology and the Internet is continuously transforming education, leading to the introduction of new methods and virtual learning environments in teaching and learning processes. However, the extent to which these innovations are effectively implemented largely depends on teachers' expertise, students' AI literacy, institutional policy visibility, and the design of assessment. Universities and faculties can recommend innovative approaches and provide support through seminars, professional development programs, and the provision of appropriate technological infrastructure.

The findings suggest that universities should not treat AI only as a technical tool. They should define acceptable assistance, disclosure and citation expectations, responsibility for errors, use in examinations, handling of confidential data, and consequences for noncompliance. Training should require students to verify claims and code, compare outputs with authoritative sources, document substantial AI assistance, and avoid entering personal, assessment, research, or proprietary data into tools without authorization. Since detection awareness was

not associated with lower complete-task generation, prevention should rely more on clear rules, assessment design, and responsible-use education than on detection alone.

The study has limitations. The sample was recruited by convenience from five universities and is not representative of national student populations. Institution, country, recruitment phase, field of study, and gender composition are partly confounded. The questionnaire was investigator-developed and was not psychometrically validated; therefore, item groups should not be interpreted as latent constructs. The results are self-reported and cross-sectional, so they cannot establish causal effects or actual learning outcomes. Future studies should use preregistered hypotheses, balanced sampling, validated AI-literacy and adoption measures, qualitative coding of open responses, and longitudinal or intervention designs.

6. Conclusion

The results of our study present empirical evidence regarding the perception and use of AI tools by students in higher education. The findings show that AI tools are widely used and generally considered beneficial for supporting educational activities, programming, problem solving, and academic writing. At the same time, the results show that frequent use coexists with incorrect AI output, partial trust, complete-task generation, and limited awareness of institutional guidelines.

The statistical and ethics analyses suggest that universities should combine access to AI tools with explicit instruction in verification, disclosure, privacy, academic integrity, and acceptable assistance. Gender-related findings should be interpreted cautiously: selected tool and task differences were observed, but most primary comparisons were small and did not remain significant after multiplicity correction. These findings underline the importance of clear institutional policies and AI literacy rather than broad claims about student groups.

In this research, we have identified some of the challenges that higher education institutions and students may face as a result of the expansion and integration of new technologies into teaching and the pedagogical process. The findings of this study can serve for our future research, especially in areas that require analytical reasoning and structured thinking, such as theoretical computer science and formal methods. Future studies should use stronger sampling, validated measures, and longitudinal or intervention designs to evaluate whether AI-literacy and policy training improve responsible use and learning outcomes.

Ethics and data availability

Participation was voluntary. Respondents received information about the study purpose and provided consent before accessing the questionnaire. Four submissions that declined consent contained no study responses and were excluded. The study used an anonymous online questionnaire, collected no directly identifying personal data, involved adult university students, and presented no more than minimal risk to participants. According to the applicable institutional and national regulations for anonymous educational survey research, formal ethics committee approval was not required. The de-identified survey data in Excel format, analysis code, and aggregate Excel and PNG outputs are available from the corresponding author upon reasonable request.

References

[1] European Education Area. Last accessed 8 February 2025.

[2] European framework for digitally competent educational organisations – DigCompOrg, 2024. Last accessed 8 February 2025.

[3] SELFIE – a tool to support learning in the digital age, 2024. Last accessed 8 February 2025.

[4] AI in Education. `https://forms.gle/R5UtcyEjE5xTGkmT6`, 2025.

[5] Naya Abdallah, Rateb Katmah, Kinda Khalaf, and Herbert F. Jelinek. Systematic review of ChatGPT in higher education: Navigating impact on learning, wellbeing, and collaboration. *Social Sciences & Humanities Open*, 12:101866, 2025.

[6] Pavol Adamík, Matúš Mištrik, Alexander Krajči, and Michal Kvet. Security aspects of collecting and transmitting health data from Apple Watch to healthcare systems. *Lecture Notes in Networks and Systems*, 1524:37–43, 2026.

[7] David Baidoo-Anu and Leticia Owusu Ansah. Education in the era of generative artificial intelligence

(AI): Understanding the potential benefits of ChatGPT in promoting teaching and learning. *Journal of AI*, 7(1):52–62, 2023.

[8] Anton Bastian, Nils Buchholtz, and Gabriele Kaiser. Using AI chatbots to facilitate mathematics preservice teachers' noticing skills. *Frontiers in Education*, 10, 2025.

[9] Mohd. Abass Bhat, Chandan Kumar Tiwari, Preeti Bhaskar, and Shagufta Tariq Khan. Examining ChatGPT adoption among educators in higher educational institutions using extended UTAUT model. *Journal of Information, Communication and Ethics in Society*, 22(3):331–353, 2024.

[10] Saeed Awadh Bin-Nashwan, Mouad Sadallah, and Mohamed Bouteraa. Use of ChatGPT in academia: Academic integrity hangs in the balance. *Technology in Society*, 75:102370, 2023.

[11] Kyle Bittle and Omar El-Gayar. Generative AI and academic integrity in higher education: A systematic review and research agenda. *Information*, 16(4):296, 2025.

[12] Mohamed Bouteraa et al. Understanding the diffusion of AI-generative (ChatGPT) in higher education: Does students' integrity matter? *Computers in Human Behavior Reports*, 14:100402, 2024.

[13] Paula Caldas Brognoli and Melissa Bertolini Rodrigues. Uso do ChatGPT por estudantes italianos na Universidade degli Studi di Perugia-UniPG. *Revista Nova Paideia*, 7(1):332–350, 2025.

[14] Astrid Carolus, Martin J. Koch, Samantha Straka, Marc Erich Latoschik, and Carolin Wienrich. MAILS – Meta AI Literacy Scale: Development and testing of an AI literacy questionnaire. *Computers in Human Behavior: Artificial Humans*, 1(2):100014, 2023.

[15] Renáta Cenková and William Steingartner. Use of internet social networks in academic environment. *Journal of Information and Organizational Sciences*, 44(2):275–295, 2020.

[16] Sofia Chambers. GenAI use and risks in higher education: A preliminary review for research in New Zealand contexts. *Scope: Contemporary Research Topics*, 15:8–15, 2025.

[17] Lijia Chen, Pingping Chen, and Zhijian Lin. Artificial intelligence in education: A review. *IEEE Access*, 8:75264–75278, 2020.

[18] Merve Çildir. Collaborating with ChatGPT: An exploratory study of German language learning. *Alman Dili ve Kültürü Araştırmaları Dergisi*, 5(2):63–76, 2023.

[19] Joseph Crawford, Michael Cowling, and Kelly-Ann Allen. Leadership is needed for ethical ChatGPT: Character, assessment, and learning using artificial intelligence. *Journal of University Teaching and Learning Practice*, 20(3), 2023.

[20] Fernando de Oliveira Santini et al. Understanding students' technology acceptance behaviour: A meta-analytic study. *Technology in Society*, 81:102798, 2025.

[21] Ruiqi Deng, Maoli Jiang, Xinlu Yu, Yuyan Lu, and Shasha Liu. Does ChatGPT enhance student learning? A systematic review and meta-analysis. *Computers & Education*, 227:105224, 2025.

[22] Silvia Farias-Gaytan, Ignacio Aguaded, and Maria-Soledad Ramirez-Montoya. Transformation and digital literacy: Systematic literature mapping. *Education and Information Technologies*, 27(2):1417–1437, 2022.

[23] Fauzi Fauzi et al. Analysing the role of ChatGPT in improving student productivity in higher education. *Journal of Education*, 5(4):14886–14891, 2023.

[24] Selen Galiç, Mathias Tejera, Sebastián Parodi, and Zsolt Lavicza. Becoming a mathematics teacher with dual AI roles: student and mentor simulations in professional identity development. 2026.

[25] Shuchen Guo, Lehong Shi, and Xiaoming Zhai. Validating an instrument for teachers' acceptance of artificial intelligence in education, 2024.

[26] Michael Haman and Milan Školník. Using ChatGPT to conduct a literature review. *Accountability in Research*, 31(8):1244–1246, 2024.

[27] Djordje Herceg, Davorka Radaković, Mirjana Ivanović, and Dejana Herceg. Possible improvements of modern dynamic geometry software. *Computer Tools in Education*, 2:72–86, 2019.

[28] Jelena Ilić, Mirjana Ivanović, and Aleksandra Klašnja-Milićević. The impact of ChatGPT on student learning experience in higher STEM education. In *ITHET*, pages 1–9, 2024.

[29] Mirjana Ivanović et al. Current trends in AI-based educational processes—an overview. In *Artificial Intelligence for Knowledge Management*, pages 1–15. Springer, 2022.

[30] Petros Lameras and Sylvester Arnab. Power to the teachers: An exploratory review on artificial intelligence in education. *Information*, 13(1), 2022.

[31] Sruti Mallik and Ahana Gangopadhyay. Proactive and reactive engagement of artificial intelligence methods for education: a review. *Frontiers in Artificial Intelligence*, 6, 2023.

[32] Nimród Mike et al. The impact of ChatGPT on student performance in higher education. In *EGOV-CeDEM-ePart 2024*, volume 3737, 2024.

[33] Amr M. Mohamed. Exploring the potential of an AI-based chatbot in enhancing English as a foreign language teaching. *Education and Information Technologies*, 29:3195–3217, 2024.

[34] Ehsan Mohammadi et al. Is generative AI reshaping academic practices worldwide? *Information Processing & Management*, 63(1):104350, 2026.

[35] Jane Nam. 56% of college students have used AI on assignments or exams, 2023. Last accessed 4 March 2026.

[36] Davy Tsz Kit Ng et al. Conceptualizing AI literacy: An exploratory review. *Computers and Education: Artificial Intelligence*, 2:100041, 2021.

[37] José M. Nieto-Jalil et al. The impact of GPT, experiential learning, and reinforcement methods on complex problem solving. In *EDUNINE*, pages 1–6, 2025.

[38] Mieczyslaw L. Owoc, Agnieszka Sawicka, and Pawel Weichbroth. Artificial intelligence technologies in education: Benefits, challenges and strategies of implementation. Springer, 2021.

[39] Jay Rhald C. Padilla, Myron Darrel L. Montefalcon, and Alexander A. Hernandez. Language AI in programming. In *ICSPC*, pages 276–281, 2023.
[40] Birgit Pepin, Nils Buchholtz, and Ulises Salinas-Hernández. A scoping survey of ChatGPT in mathematics education. *Digital Experiences in Mathematics Education*, 11(1):9–41, 2025.
[41] Ján Perháč et al. OnlineProver: Experience with a visualisation tool for teaching formal proofs. *EPTCS*, 419:55–74, 2025.
[42] Ivan Pribela, Mirjana Ivanović, and Zoran Budimac. Svetovid—interactive development and submission system. *British Journal of Educational Technology*, 40(6):1076–1093, 2009.
[43] Mónica De La Roca et al. The impact of a chatbot working as an assistant in a course. *Computer Applications in Engineering Education*, 32(7):e22750, 2024.
[44] Davorka Radaković, William Steingartner. Gender and Emerging Digital Technologies in Education. *In: Alvarez, I., Arias-Oliva, M., Dediu, AH., Silva, N. (eds) Ethical and Social Impacts of Information and Communication Technology. ETHICOMP 2025. Lecture Notes in Computer Science, Springer, Cham*, 15939: 141–152, 2026.
[45] Sandra Saúde, João Paulo Barros, and Inês Almeida. Impacts of generative artificial intelligence in higher education. *Social Sciences*, 13(8):410, 2024.
[46] Carlos Alexandre Gouvea da Silva et al. ChatGPT: Challenges and benefits in software programming for higher education. *Sustainability*, 16(3):1245, 2024.
[47] Hamed Taherdoost. Data collection methods and tools for research. *International Journal of Academic Research in Management*, 10(1):10–38, 2021.
[48] Xiao Tan, Gary Cheng, and Man Ho Ling. Artificial intelligence in teaching and teacher professional development: A systematic review. *Computers and Education: Artificial Intelligence*, 8:100355, 2025.
[49] H. Holden Thorp. ChatGPT is fun, but not an author. *Science*, 379(6630):313, 2023.
[50] Yi Wu. Integrating generative AI in education: How ChatGPT brings challenges for future learning and teaching. *Journal of Advanced Research in Education*, 2(4):6–10, 2023.
[51] Yong Zheng. ChatGPT for teaching and learning: An experience from data science education. In *SIGITE '23*, pages 66–72. ACM, 2023.

Davorka Radaković holds a PhD in Computer Science and is an Assistant Professor at the the University of Novi Sad, Faculty of Sciences, Department of Mathematics and Informatics, Novi Sad, Serbia. She received her B.Sc. in Mathematics in 2001, M.Sc. (former Mr, 2 years) in Informatics in 2010, Ph.D. degree in Computer Science in 2019, and M.Sc. in Teaching Informatics in 2023 from the University of Novi Sad (Serbia), Faculty of Sciences. Her scientific research is software engineering and education.
E-mail: davorkar@dmi.uns.ac.rs
OrcidID: 0000-0001-8480-3211

William Steingartner works as Associate Professor of Computer Science at the Department of Computers and Informatics of the Faculty of Electrical Engineering and Informatics, Technical University of Košice, Slovakia. He defended his PhD thesis "The *Rôle* of Toposes in Informatics" in 2008. His main fields of research are the semantics of programming languages, category theory, compilers, data structures and recursion theory. He also works with type theory and software engineering.
E-mail: william.steingartner@tuke.sk
OrcidID: 0000-0002-2852-9403

Dora Seleši is Full Professor at the University of Novi Sad, Faculty of Sciences, Department of Mathematics and Informatics. She received her B.Sc. in Mathematics in 2000, M.Sc. (former Mr, 2 years) in Mathematics in 2004, Ph.D. degree in Mathematics in 2007 from the University of Novi Sad (Serbia), Faculty of Sciences. Currently she is Head of Studies for Applied Mathematics. Her research includes probability theory, stochastic analysis and stochastic differential equations, statistics and data science.
E-mail: dora@dmi.uns.ac.rs
OrcidID: 0000-0003-2261-0590

Supplementary Material

Data Reconciliation and Analysis Population

The combined worksheet contained 231 submissions. Four records declined consent and contained no study responses, leaving 227 participants. Rrespondent-level data contain 154 men, 69 women, two non-binary/third-gender respondents, and two respondents who preferred not to state gender. Phase 1 contains 84 records and phase 2 contains 143, including one March 2026 timestamp. Institution counts are Novi Sad 88, Košice 76, Warsaw 42, Banská Bystrica 11, and Maribor 10. All descriptive results retain the four non-binary or undisclosed-gender records. Male–female inferential comparisons exclude them because the individual cells are too small for stable estimation. Multiple-response percentages use the complete item denominator and can sum to more than 100%. Missing responses are not imputed.

Questionnaire and Codebook

Table S1 lists the questionnaire items relevant to this study. The wording follows the supplied workbook headers, while the first column uses short semantic labels rather than spreadsheet column letters. Coding questions and open-text items are retained for reproducibility but were not used for unsupported qualitative inference in the revised manuscript.

Table S1: Questionnaire items, response formats, and analysis coding

Item	Item wording	Response format and analysis use
Consent	Consent to participate	Yes or No; No ended the survey.
Age	What is your age?	Open numeric; one nonnumeric refusal treated as missing.
Gender	What is your gender?	Male; Female; Non-binary/third gender; Prefer not to say.
Education level	Highest education completed	Elementary school; High school; Bachelor's; Master's.
Institution type	Type of institution	Categories as presented in the form.
Institution name	Name of institution	Open text; standardized into five institutions.
Field of study	Field of study or subject area	Open text; used descriptively.
Country	Country of study	Open text; used descriptively.
AI access	Access to AI tools in the educational environment	No; Not sure; Yes; ordered 0–2.
Field role of AI	Perceived role of AI in the respondent's field	Open text; excluded from formal qualitative inference.
AI familiarity	Familiarity with artificial intelligence	Five ordered categories; stored 1–5.
Prior AI use	Prior use of AI tools in work or daily life	Yes or No; determines user/non-user branch.
Tool categories used	Types of AI tools used	Multiple response: assistants; writing; search; video; presentations; image; notetakers; music; other.
AI-use frequency	Frequency of AI-tool use	Never; less than weekly; once weekly; 2–4 times weekly; 5–6 times weekly; daily; ordered 0–5.
Assistant brands	AI assistants/chatbots used	Multiple response including ChatGPT, Gemini, Copilot, Claude, DeepSeek, and other.
Assistant choice reasons	Reasons for using AI assistants	Multiple response: free; popular; recommended; other.
Willingness to pay	Willingness to pay for AI assistants	No; Maybe; Yes.
Teaching/learning use	Use of AI to support teaching or learning	Yes or No; four AI-user records missing.
AI-use purpose	Primary tasks for which AI is used	Multiple response: problem solving; programming; teaching; homework/tasks; essays; other.
Adoption motivation	Motivation for adopting AI in learning	Open text; excluded from formal qualitative inference.
Educational AI tools	AI tools used in an educational context	Multiple response using the categories in tool categories used.
Learning effectiveness	Effectiveness of AI in enhancing learning outcomes	Seven ordered categories from absolutely ineffective to absolutely effective; stored 1–7.
Independent learning	Whether AI helps students become independent learners	No; Maybe; Yes; ordered 0–2.
Complete-task generation	Use of AI to generate complete homework tasks	Yes or No; primary behaviour outcome for the ethics analyses.

Table S1 continued

Item	Item wording	Response format and analysis use
Integrity-threat attitude	AI-assisted dishonesty as a serious integrity threat	Five-point agreement item; stored 1–5.
Ethical-use teaching	Whether students should be taught ethical AI use	No; Maybe; Yes; ordered 0–2.
Treatment preferences	How AI-assisted dishonesty should be treated	Multiple response; descriptive.
Plagiarism-detection awareness	Awareness of plagiarism-detection software	Yes or No.
AI-detection awareness	Awareness of AI-detection tools	Yes or No.
Institutional guideline visibility	Institution has AI-use guidelines	No; I do not know; Yes.
Formal policy demand	Formal AI policies should exist	No; I do not know; Yes; ordered 0–2 for exploratory comparison.
Replacement expectation	Whether AI will replace teaching tasks or roles	No; I do not know; Yes.
Trust in AI output	Trust in AI-generated output	Five ordered categories from none to full trust; stored 1–5.
Trusted-output contexts	Contexts in which AI output is trusted	Multiple response: source verification; basic tasks; understood process; tool-dependent; human oversight; other.
Incorrect AI answers	Encountered incorrect AI answers	Yes or No.
Correction attempts	Asked AI to correct an incorrect answer	Yes or No.
Coding use	Use of AI for coding	Intended Yes/No; one observed Maybe response retained descriptively.
Coding-use frequency	Frequency of AI use for coding/software engineering	Rarely; Sometimes; Frequently; Always; some nonstandard form-version entries.
Software-engineering tasks	Software-engineering tasks using AI	Multiple response: requirements; design; coding; debugging; testing; documentation; other.
Coding prompt style	Best description of coding prompt use	Basic or proficient prompt-use description.
Debugging support	How AI help is requested for debugging	Multiple response: paste errors; ask for explanations; step-by-step prompts; other.
Advanced prompting	Advanced prompting used	Multiple response: none; few-shot; chain-of-thought; role-based; constraints.
AI work checking	Whether AI work is checked	Always or sometimes in the implemented dataset.
AI-use struggles	Biggest struggle with AI	Multiple response and other text: mistakes; overreliance; ethics; other.
Help/failure example	Example where AI helped or failed	Open text.
Desired AI skills	AI skills the respondent wants to learn	Open text.
Prompt-pattern knowledge	Knowledge of prompt patterns	Yes or No.
Non-user consideration	Intention to consider AI	No; Maybe; Yes; non-user branch only.
Non-user education intent	Intention to use AI in education	No; Maybe; Yes; non-user branch only.

Complete Gender Comparisons

Multiple-response educational tools

Table S2: Educational tool selections by gender among AI users

Option	Female n/N	Male n/N	OR	95% CI	Holm p
AI assistants/chatbots	57/65	140/146	0.31	0.10–0.92	0.256
Grammar/writing improvement	33/65	44/146	2.39	1.31–4.36	0.042
Search engines	22/65	64/146	0.66	0.36–1.21	1.000
Video generation/editing	2/65	4/146	1.13	0.20–6.31	1.000
Presentations	10/65	17/146	1.38	0.59–3.20	1.000
Image generation	9/65	20/146	1.01	0.43–2.36	1.000
Notetakers/meeting assistants	4/65	14/146	0.62	0.20–1.96	1.000
Music generation	0/65	1/146	0.00	0.03–18.42	1.000

Multiple-response task purposes

Table S3: Task-purpose selections by gender among AI users

Option	Female n/N	Male n/N	OR	95% CI	Holm p
Solving problems	59/65	108/146	3.46	1.38–8.66	0.028
Programming	39/65	109/146	0.51	0.27–0.95	0.142
Teaching	29/65	65/146	1.00	0.56–1.81	1.000
Writing homework/tasks	17/65	45/146	0.79	0.41–1.53	1.000
Writing essays	13/65	37/146	0.74	0.36–1.50	1.000

Assistant brands

Table S4: Assistant-brand selections by gender among AI users

Option	Female n/N	Male n/N	OR	95% CI	Holm p
ChatGPT	62/65	132/146	2.19	0.61–7.91	1.000
Gemini	31/65	66/146	1.11	0.62–1.99	1.000
Microsoft Copilot	23/65	44/146	1.27	0.68–2.36	1.000
Claude	12/65	36/146	0.69	0.33–1.44	1.000
DeepSeek	6/65	13/146	1.04	0.38–2.87	1.000

Adjusted Sensitivity Models

Binary outcomes were analyzed with logistic regression adjusted for recruitment phase and institution. These are sensitivity analyses, not replacements for the ordinal primary analyses. Holm correction was applied across the seven models.

Table S5: Phase- and institution-adjusted gender sensitivity models

Outcome	n	Events	Adjusted OR	95% CI	Holm p
Access: yes	223	154	0.92	0.49–1.73	1.000
Familiarity: moderate or higher	223	158	0.34	0.18–0.66	0.009
Use 5–6 times weekly or daily	211	104	1.05	0.58–1.91	1.000
Effective or absolutely effective	211	120	0.79	0.43–1.45	1.000
Independent learner: yes	211	95	1.14	0.62–2.12	1.000
Integrity threat: agree/strongly agree	211	122	0.46	0.25–0.86	0.089
Generated complete homework: yes	211	99	0.76	0.41–1.42	1.000

Ethics and Governance Analyses

Analyses in this section include the 214 prior AI users who received the ethics and governance items listed in Table S1. Site comparisons retain the three response categories for institutional guideline visibility and formal policy demand and use 10,000-permutation chi-square tests. Holm correction was applied across the two site outcomes and separately within the purpose, treatment-preference, and detection-awareness families. Odds ratios compare the first named group with its reference group. Adjusted models were restricted to 211 male or female respondents and included recruitment phase and institution; the complete-task model additionally included familiarity and all five task purposes.

Table S6: Institutional guideline visibility and formal policy demand by institution

Institution	n	Guidelines visible: Yes	%	Formal policies: Yes	%
Novi Sad	83	15	18.1	54	65.1
Košice	72	23	31.9	48	66.7
Warsaw	38	6	15.8	14	36.8
Banská Bystrica	11	3	27.3	10	90.9
Maribor	10	6	60.0	5	50.0

Institutional guideline visibility: Cramer's $V = 0.191$, permutation $p = 0.047$, Holm $p = 0.047$. Formal policy demand: $V = 0.226$, permutation $p = 0.005$, Holm $p = 0.009$. Small-site percentages are unstable and institution overlaps with country, phase, and program mix.

Table S7: Core behavior, attitude, intensity, and policy associations

Analysis	Comparison	Effect	95% CI	p
Integrity-threat rating by complete-task behavior	Yes vs No; median 4 (3–4) vs 4 (3–5)	$r = -0.208$	-0.352–-0.061	0.006
Use frequency by complete-task behavior	Yes vs No; median 4 (3–5) vs 3 (2–4)	$r = 0.321$	0.186–0.460	< 0.001
Policy support by guideline visibility	77.4% vs 55.9%	OR 2.70	1.32–5.51	0.006
Policy support by guideline visibility, adjusted	211 respondents	OR 2.73	1.23–6.08	0.014
Integrity-threat agreement by complete-task behavior, adjusted	211 respondents	OR 0.78	0.42–1.44	0.424

Table S8: Complete-task generation by AI-use frequency

Frequency	Yes/n	%	Wilson 95% CI
Never	0/2	0.0	0.0–65.8
Less than once a week	4/14	28.6	11.7–54.6
About once a week	5/25	20.0	8.9–39.1
2–4 times a week	31/68	45.6	34.3–57.3
5–6 times a week	17/41	41.5	27.8–56.6
Every day	43/64	67.2	55.0–77.4

Table S9: AI-use purposes and complete-task generation

Purpose	Selected %	Unselected %	Crude OR	Adjusted OR (95% CI)	Holm p
Solving problems	50.9	31.1	2.29	1.39 (0.60–3.22)	0.889
Programming	56.0	25.0	3.82	2.94 (1.35–6.40)	0.046
Teaching	57.4	38.3	2.17	2.54 (1.29–5.02)	0.046
Writing homework/tasks	74.6	35.1	5.43	2.82 (1.20–6.62)	0.088
Writing essays	72.5	38.7	4.20	1.03 (0.40–2.67)	0.947

Displayed Holm p values refer to the seven-term adjusted complete-task model family (frequency, familiarity, and five purposes). Crude within-purpose results are available in the accompanying workbook.

Table S10: Preferred treatment of AI-assisted dishonesty (item AB)

Preference	Overall %	Complete-task %	Other %	OR (95% CI)	Holm p
Strict penalty	10.7	5.0	15.8	0.28 (0.10–0.79)	0.070
Better policies and awareness	36.9	23.0	49.1	0.31 (0.17–0.56)	< 0.001
Shift in learning and assessment	43.0	38.0	47.4	0.68 (0.39–1.18)	0.852
Similar to other cheating	21.0	22.0	20.2	1.12 (0.58–2.16)	0.934
Adapt rather than punish	44.9	49.0	41.2	1.37 (0.80–2.35)	0.852
Not a serious issue	16.8	19.0	14.9	1.34 (0.65–2.74)	0.934

Table S11: Detection awareness and complete-task generation

Awareness item	Aware %	Unaware %	Crude OR (95% CI)	Holm p
Plagiarism-detection software	48.9	29.2	2.33 (0.92–5.87)	0.166
AI-detection tools	48.6	37.1	1.60 (0.76–3.37)	0.267

Adjusted odds ratios were 2.23 (95% CI 0.81–6.11; Holm $p = 0.240$) and 1.77 (0.77–4.06; Holm $p = 0.240$), respectively, after controlling for use frequency, gender, phase, and institution. Awareness was not associated with less complete-task generation.

Supplementary Figures

The remaining generated figures are provided below for transparency and reproducibility.

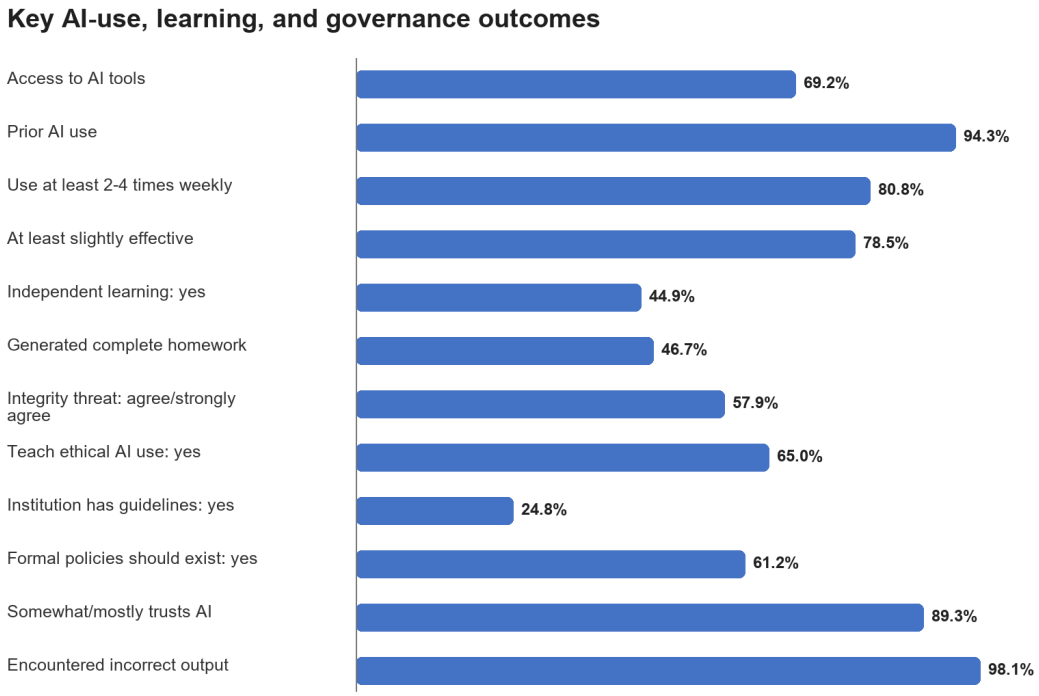


Figure S1: Key AI-use, learning, integrity, and governance outcomes. Percentages use item-specific denominators.

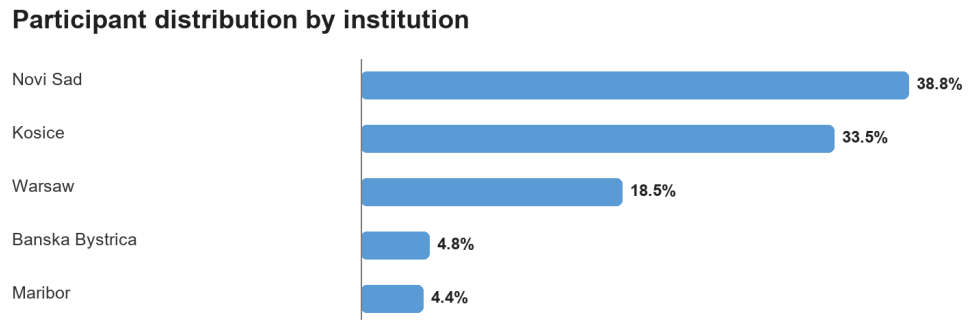


Figure S2: Unequal institutional composition of the convenience sample.

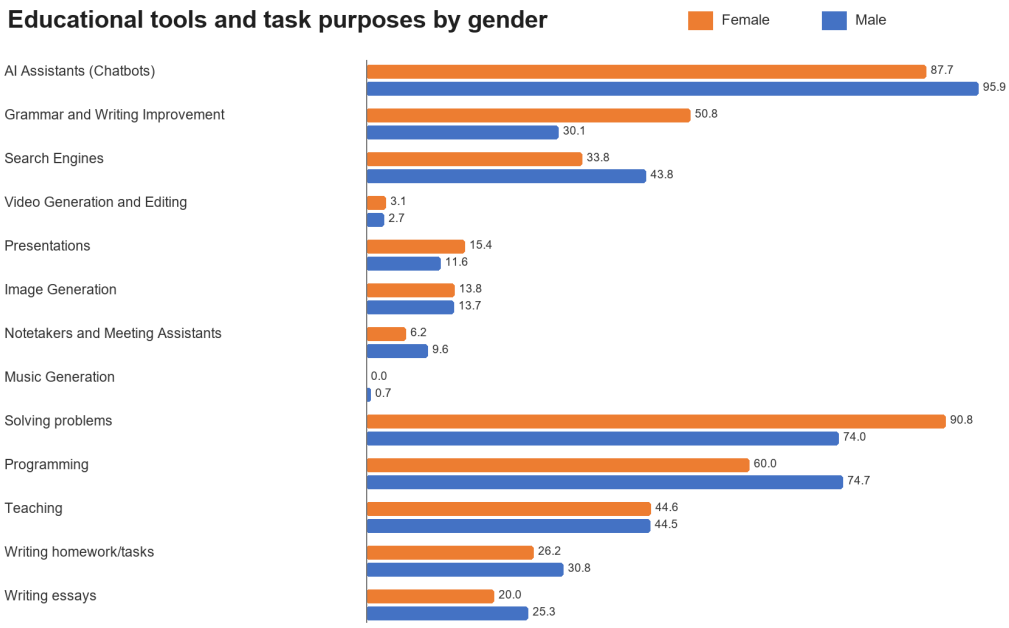


Figure S3: Educational-tool and task-purpose selections among male and female AI users. Inferential results are reported in Tables 2 and 3.

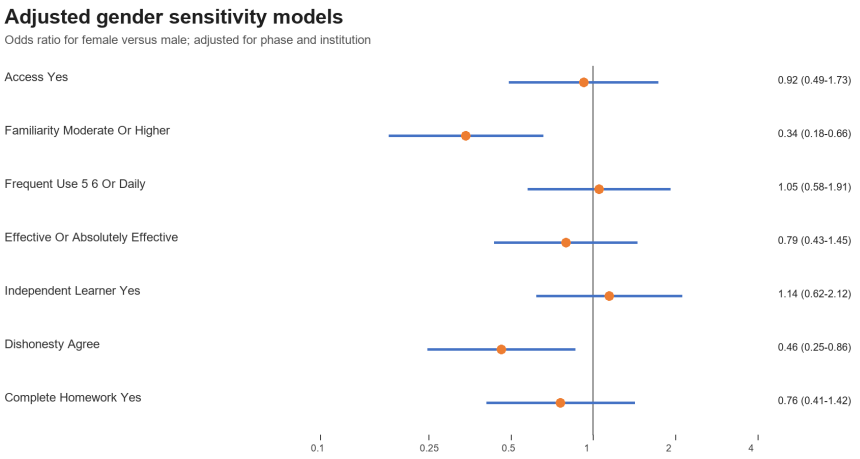


Figure S4: Adjusted odds ratios for female versus male respondents. Models adjust for recruitment phase and institution.

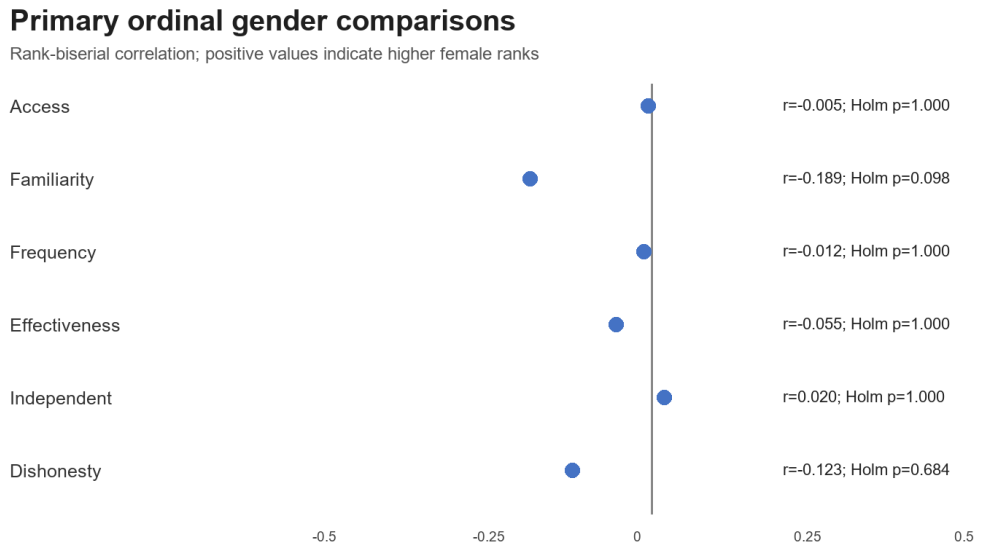


Figure S5: Rank-biserial effects for the six primary ordinal gender comparisons. Positive values indicate higher female ranks.

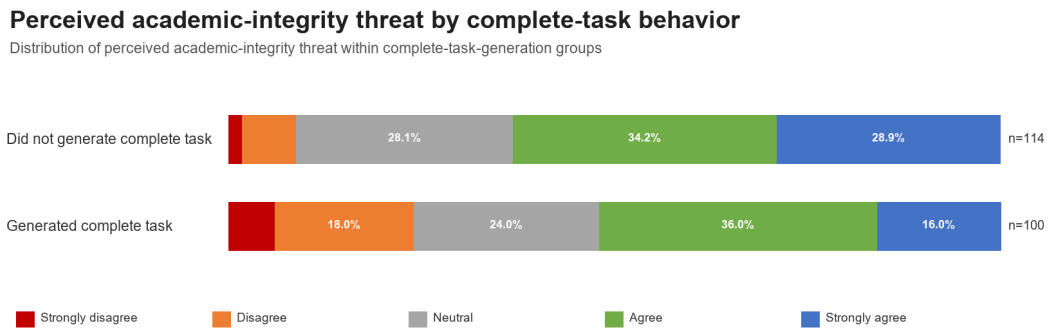


Figure S6: Integrity-threat response distribution according to complete-task behavior.

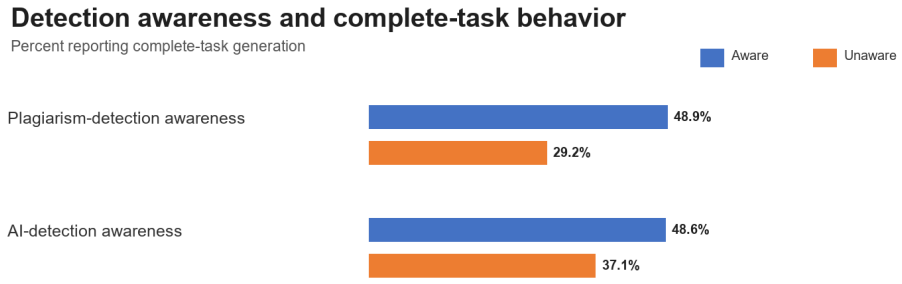


Figure S7: Complete-task behavior by awareness of plagiarism- and AI-detection tools.

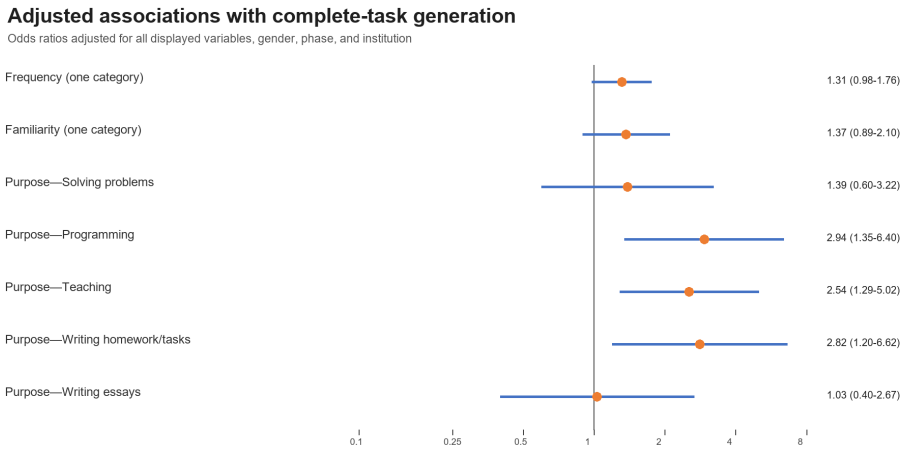


Figure S8: Adjusted associations of use intensity and task purposes with complete-task generation.

ForiC-a Fast and Effective Method for Predicting the Origin of Bacterial Chromosome Replication

Stojićević, Marko

Abstract: Identifying the origin of replication (oriC) is fundamental to understanding bacterial genome organization and replication mechanisms. Traditional methods, such as GC-skew and cumulative GC-skew (CGC) analyses, are computationally simple but often imprecise. Advanced approaches, including DnaA-box detection and comparative genomics, can improve accuracy but are frequently resource-intensive and dependent on external biological information. This paper presents ForiC, a novel method that bridges this gap by combining the simplicity of GC- and CGC-based analyses with improved oriC-region localization performance. Beyond using CGC inflection points, ForiC integrates three complementary signals within a deterministic selection rule: the CGC global maximum, the CGC inflection point associated with that maximum, and its alignment with the GC global minimum. The resulting framework operates directly on the input genome sequence, without DnaA-box motifs, homology searches, or external databases. ForiC was calibrated and evaluated on a curated set of 36 annotated bacterial genomes using five phylum-stratified training/test splits. On one pre-defined stratified test subset selected as a representative balanced test subset, ForiC reached a hit accuracy of 90.91% using a 1% window and a symmetrically expanded two-window reporting interval, with complementary localization metrics (center-distance error, intersection-over-union) and one-window results reported in the main text. ForiC was benchmarked against classical skew-based methods (GC, CGC) and against a biologically informed reference tool (OriFinder). OriFinder achieves higher positional precision through homology-based refinement but may report multiple candidate oriC regions and depends on external databases such as BLAST. In contrast, ForiC consistently produces a single, directly interpretable prediction per genome and runs substantially faster without external resources, making it complementary to reference-assisted tools and particularly suitable for large-scale, database-independent analyses. ForiC is available at http://www.markostojicevic.com/foric.

Index Terms: ***algorithms, bioinformatics, computational genomics, GC skew, oriC prediction***

1. INTRODUCTION

DNA replication is one of the most important biological processes. This ensures the inheritance of attributes that define the phenotype of cells and the organism. In this complicated process, genetic information must be accurately copied to avoid mutations and ensure the survival of living organisms. Complicated as it is, the DNA replication process tends to differ in all three domains of life (the Bacteria, the Eukarya and the Archaea) and between organisms in the same domain [1], [2], [3], [4], [5].

Bacteria usually contain circular DNA chromosomes with a specific locus designated as the origin of replication (oriC). oriC contains three functional elements: a DNA repeat, an AT-rich region and regulatory protein recognition sites [6]. Moreover, the analysis of bacterial oriC regions revealed high-scale diversity, increasing the complexity of oriC sequence recognition [7], [8]. To better understand the mechanism of DNA replication initiation, it is important to identify the position of the replication origin.

DNA asymmetry is the most universal method of putative oriC identification in bacterial chromosomes [9]. The analysis revealed that the leading strand contains more Guanine (G) and Thymine (T), whereas the lagging strand contains more Cytosine (C) and Adenine (A) [10], [11]. The

[1] Manuscript submitted on August 2, 2026. Marko Stojićević, University of Belgrade, Faculty of Mathematics, Department of Informatics and Computer Science, Belgrade, Serbia, mstojicevic@gmail.com, ORCID: 0009-0005-8771-2012.

asymmetry of the base pairs that have been found in the strands is strong enough to identify the location of the oriC position via mathematical methods [12]. There are three main methods that measure nucleotide asymmetry: GC skew, cumulative GC (CGC) skew and the Z curve.

Although these approaches are widely used, each exhibits specific limitations. Basic GC skew is sensitive to window size and can produce multiple ambiguous asymmetry transitions, which reduces positional accuracy [13], [14]. Cumulative GC skew smooths the signal but often yields broad candidate regions rather than precise oriC locations, and its accuracy decreases in genomes where the GC skew signal itself is weak or where local compositional biases distort the cumulative curve [15]. The Z-curve provides richer compositional information but requires careful interpretation of multiple curve components, and in practice its accuracy typically depends on integration with additional biological signals such as DnaA-box motifs [16], [17]. These limitations motivate the development of methods that preserve the computational simplicity of skew-based analyses while increasing localization accuracy through additional, intrinsically derived features.

The GC skew method is based on GC asymmetry and is calculated as (G-C)/(G+C) [10], [13], [14]. This method is based on the frequency of base pairs C and G in a given sequence of the genome. The genome is divided into equal-sized windows, and for each window, the imbalance between guanine (G) and cytosine (C) nucleotides is quantified as the GC skew value. These values are often represented in the form of a graph, where the x-axis shows the position (of the window) within the genome and the y-axis shows the GC skew value. The selected window size strongly influences the GC skew signal, affecting how clearly genomic patterns can be visualized and interpreted. This highlights the necessity of carefully selecting an appropriate window size to facilitate accurate analysis of the GC skew.

The second method, CGC skew, resolves most of the flaws of the GC skew method and is calculated as a sum of (G-C)/(G+C) in adjacent windows from an arbitrary start to a given point in a sequence [18]. CGC skew also uses sliding windows, but each value reflects the running sum of GC differences from all preceding windows. As a result, CGC skew plots are typically smoother and easier to interpret than standard GC skew plots, since the cumulative transformation reduces abrupt changes in curve curvature.

The third method is the Z-curve method, which represents a DNA sequence as a three-dimensional mathematical curve obtained by cumulating nucleotide counts along the genome [16], [19], [20], [21]. Its three coordinates capture complementary aspects of base composition: purine/pyrimidine (R/Y) asymmetry, amino/keto (M/K) asymmetry, and strong/weak hydrogen-bond (S/W) asymmetry, corresponding to the A+G vs. C+T, A+C vs. G+T, and G+C vs. A+T distinctions, respectively. Unlike the previous methods, the Z-curve does not require a sliding-window strategy; characteristic turning points of these curves have been used to infer candidate oriC positions.

Deeper analysis in various studies indicates that the switch in asymmetry does not always correspond to the position of oriC [22]. Various software tools implement methods such as GC, CGC, Z-curve, and other skew techniques [23]. Among the bacterial oriC predictors, one of the most widely used tools is OriFinder [17]. This algorithm comprises several steps, with the primary components being the determination of the Z-curve and analysis of DnaA-box distributions in the given bacterial genome. The predicted genomes are deposited in DoriC [24], a database of predicted oriC positions. In recent years, more species-specific indicator genes and functional elements of oriCs have been integrated into updated frameworks, leading to newer versions of OriFinder [25].

Recent studies have also introduced alternative oriC prediction frameworks, including open-source and machine-learning-based approaches [26], [27], [28]. However, these methods differ substantially in scope, target datasets, or implementation strategies. Therefore, OriFinder was retained in the present study as the principal comparative benchmark due to its broad recognition and established use in bacterial oriC prediction.

More specifically, existing models present several practical drawbacks. Basic GC and CGC skew methods often lack precision in predicting the exact location of oriC, frequently identifying broad candidate regions rather than the precise position of the oriC site. OriFinder, while regarded

as a highly accurate tool, depends heavily on BLAST for functional annotation of genes in unannotated genomes. This reliance introduces two key limitations: (1) the computational overhead of performing sequence alignments, which increases processing time and resource demands, especially for large-scale analyses, and (2) the dependence on the completeness and accuracy of external databases, which may vary over time and across research environments.

This study aims to propose a new, more effective method for oriC prediction that overcomes these limitations by utilizing an enhanced GC asymmetry-based approach. The ForiC (Fast oriC) platform eliminates the need for BLAST or extensive biological information such as DnaA box motifs. Instead, the method leverages a combination of GC and CGC skew analyses as the foundation. These methods are further refined by incorporating a mathematical framework that analyzes higher-order structural properties of CGC skew curves, rather than relying solely on local maxima, minima, or direct asymmetry transitions. In particular, the approach uses inflection points as informative features and combines them with additional genomic signals to improve oriC localization. Key parameters of the method, specifically the window size and prediction interval length, were calibrated empirically by evaluating candidate values on stratified training partitions and selecting the configuration with the most consistent performance. This approach differs from other methods by providing a faster, database-independent, and more versatile tool for bacterial origin prediction.

2. *Methods and Materials*

2.1. *ForiC method description*

The ForiC method incorporates an initial input-validation phase to ensure that only biologically plausible genome sequences are subjected to detailed analysis. This preliminary screening includes basic sequence-length and GC-content sanity checks, together with simple compositional descriptors such as entropy, GC asymmetry, and nucleotide-count variance. These measures are intended to exclude trivial, highly atypical, or low-information inputs before the main prediction stage. Full definitions and implementation details of the preliminary validation criteria are provided in Additional file 1.

After this screening step, the validated genome sequence is processed using the core ForiC framework based on sliding-window GC and CGC skew analysis. Similar to GC and CGC skew methods, ForiC requires only the genome sequence as input and does not depend on external annotations or prior biological features. The computational complexity of the ForiC algorithm can be described as O(n), where n denotes the number of base pairs in the genome sequence.

A defining feature of the ForiC algorithm is the use of specific inflection points (IPs) within the CGC graph (positions where the numerically approximated curvature changes sign) as an auxiliary positional cue alongside the CGC global maximum and the GC global minimum. The inflection point is not intended to replace the CGC extremum traditionally used to bracket oriC, but to refine the predicted interval when the global maximum alone may not unambiguously localize the oriC-containing region. Among these, the first inflection point following the CGC global maximum, denoted $IPCGC_{max}$, is particularly significant. This point was empirically observed to occur near annotated oriC regions in many genomes. To increase localization accuracy, the algorithm cross-references $IPCGC_{max}$ with the x-axis position of the GC global minimum. By combining these features, the ForiC algorithm is designed to improve oriC localization by integrating complementary signals from GC and CGC analyses. Figure 1 shows the CGC and GC graphs, highlighting the global extrema and inflection points used in the algorithm.



Figure 1: *Escherichia coli GC and CGC graph with CGC inflection points.* The graph shows (for the genome *CP006027 Escherichia coli O145:H28 str. RM13514*) plots for AT, CAT, GC and CGC measures, with noted global extreme values (minimum and maximum). The CGC plot also includes inflection points (shown as dots), where those of interest are those after the CGC global maximum value. The dashed black line denotes that the x-axis values overlap between the CGC inflection point and the GC global minimum value, whereas the dashed orange line denotes the oriC position.

It is appropriate to define an inflection point:

An inflection point on a differentiable curve $y = f(x)$ is a point (x_0, y_0) where the second derivative $f''(x)$ changes sign. At an inflection point, the concavity of the curve changes from concave up to concave down, or vice versa [29].

Importantly, the CGC skew does not define a curve $y = f(x)$ but rather generates a finite set of values $(x_1, x_2, \dots, x_n)$. These values can visually resemble the function's graph, but the method itself does not generate the (continuous) function's equation $f(x)$. Considering that there is no analytical definition that describes the set of x_i values, calculating the second derivative is not possible. The second derivative can be calculated only as a numerical approximation to obtain the inflection points.

The precondition of any of the calculations is to have a smooth function. The smoothness of the function depends on the number of windows used in the CGC method; as the window tends to be smaller, the number of windows (the set of x_i values) increases, leading to a smoother function. A point x_i is an inflection point whenever there is a polarity change in the numerical approximation of the second derivative. This approximation can be performed in multiple steps - first, to approximate the first derivative based on a set of x_i values and then, to calculate the approximated second derivative on the basis of the first derivative.

The first step is to approximate the first derivative using the sequence of values x_i. For each pair of consecutive windows, a new temporary variable q_i is defined as:

$$q_i = \frac{x_{i+1} - x_i}{m}, \quad i = 1, \dots, n-1$$

where m represents the size of the sliding window. The sequence $q_1, q_2, \dots, q_{n-1}$ represents an approximation of the first derivation of the function f: $f'(x)$.

The second derivative is then approximated by applying the same finite-difference procedure to consecutive values of q_i:

$$p_i = \frac{q_{i+1} - q_i}{m}, \quad i = 1, \dots, n-2$$

where m denotes the size of the sliding window.

Equivalently, this expression can be written directly in terms of the original values:

$$p_i = \frac{x_{i+2} - 2x_{i+1} + x_i}{m^2}$$

The sequence $p_1, p_2, \dots, p_{n-2}$ therefore represents an approximation of the second derivative, $f''(x)$.

On the basis of the previous equations, a point x_i is the inflection point whenever the condition $p_i * p_{i+1} < 0$ is met. The number of inflection points differs for different CGC calculations, but the number tends to decrease as the window size increases and vice versa.

CGC skew analysis is commonly used in conjunction with other genomic features to predict oriC in bacterial genomes. While various computational methods exist for oriC prediction, the transition points in CGC skew analysis, such as global maximum and global minimum values, are often considered indicative of genomic regions corresponding to oriC and the terminus, respectively. Based on the above, only inflection points around the CGC's global maximal value were important.

2.2. *ForiC Algorithm*

The algorithm applies a sliding-window strategy combining GC skew, CGC skew, and derived CGC inflection-point features. Specifically, it calculates graph inflection points, analyzes the derived data (GC, CGC, inflection points), and evaluates the correlation between these values. These steps are systematically implemented to improve the prediction precision. The ForiC algorithm is currently applied to circular bacterial genomes, and its structure is presented in the pseudocode shown in Figure 2.

Following the definition of these features, the initial stage of the algorithm involves calculating the GC skew and the CGC skew and identifying inflection points within the CGC data, which serve as critical features for subsequent analyses. As the algorithm progresses, the positioning of the oriC exhibits two distinct scenarios: either oriC is located near the origin of the circular genome's coordinate system or elsewhere within the genomic sequence.

The algorithm uses different measures for predicting the oriC position: (1) the position of the CGC inflection point associated with the CGC global maximum, (2) the overlap of the x-axis values between the CGC inflection points and the GC global minimum position, and (3) the position of the CGC global maximum.

The ForiC algorithm supports variable window sizes, allowing for adjustable granularity in the analysis of bacterial genomes. Window sizes were defined as proportions of the genome length. This flexibility facilitates interval-based predictions for oriC regions, with the output defined as a predicted genomic interval. In the standard one-window setting (1W), the predicted interval spans a single window length. In the extended two-window setting (2W), the reported oriC interval is not defined as two separately selected neighboring windows. Instead, the interval is symmetrically expanded around the initially predicted 1W interval center to a total span of two window lengths, allowing improved capture of oriC regions that extend beyond a single window. All parameters, including window size and interval length, were systematically evaluated to ensure robust performance. These features enhance the algorithm's adaptability across diverse genomic contexts while maintaining computational efficiency.

The algorithm applies specific rules to determine the oriC position based on the distribution of inflection points and CGC global maximum values.

Because ForiC currently supports only circular bacterial genomes, predictions near the coordinate-system boundary (the junction between the final and initial genomic positions) require special handling. A boundary-centered oriC interval is assigned when either (i) the x-axis alignment between IPCGCmax and the GC global minimum occurs within the last two CGC windows, or (ii) the CGC global maximum lies within the first window. In such cases, the predicted oriC interval is centered on coordinate 0 and reported as a wrap-around interval across the genome boundary. In the standard one-window prediction setting, this corresponds to an interval extending by half a window length on each side of coordinate 0.

If neither condition is met, the algorithm applies the standard prediction rule by identifying the informative CGC inflection point located after the CGC global maximum and aligned with the GC minimum. The predicted oriC interval is then assigned to the immediately following window, while the aligned inflection point serves as a positional marker rather than the predicted interval itself.

```
Input: genome (string), window size (integer), interval mode (1W or 2W)
Output: Predicted oriC interval
FUNCTION CalculateOriginInterval(genome: String, windowSize: Integer, intervalMode: String) : array of Integer
var
    inflectionPoints, ipAfterCGCMax, ipAlignedWithGCMin, oricPosition: array of Integer;
    gcSkew, cgcSkew: array of Real;
begin
    // Step 1: Calculate initial skew values
    gcSkew := CalculateGCSkew(genome, windowSize);
    cgcSkew := CalculateCGCSkew(genome, windowSize);
    inflectionPoints := CalculateInflectionPoints(cgcSkew);

    // Step 2: Identify Inflection points of interests
    ipAfterCGCMax := GetFirstInflectionPointAfterCGCMax(cgcSkew, inflectionPoints);
    ipAlignedWithGCMin := GetInflectionAlignedWithGCMin(gcSkew, inflectionPoints);

    // Step 3: Define prediction center
    if (ipAfterCGCMax = nil or (ipAlignedWithGCMin["leftPosition"] > (genomeLength – 2*windowSize)) or
    (ipAlignedWithGCMin["leftPosition"] < windowSize))
    begin
        oricPosition["leftPosition"] := genomeLength – windowSize/2;
        oricPosition["rightPosition"] := windowSize/2;
    end
    else
    begin
        oricPosition["leftPosition"] := ipAlignedWithGCMin["leftPosition"] + windowSize;
        oricPosition["rightPosition"] := oricPosition["leftPosition"] + windowSize;
    end

    // Step 4: Expand interval if intervalMode is 2W
    if (intervalMode = "2W")
    begin
        oriCPosition := ExpandSymmetrically(oriCPosition, windowSize, genomeLength);
    end

    return oriCPosition;

end;
```

Figure 2: Pseudocode of the ForiC algorithm for interval-based oriC prediction in circular bacterial genomes.

2.2.1. *Metrics for Assessing Algorithmic Prediction Performance*

The performance of the ForiC algorithm was evaluated primarily using hit accuracy, reflecting its ability to detect the oriC region. Unlike standard classification models that assign discrete labels, ForiC predicts a genomic interval expected to contain the oriC region. Consequently, classification-based metrics such as precision, recall, sensitivity, and specificity are not directly applicable, as they require clearly defined positive and negative instances at the same resolution. Localization-oriented metrics described below were therefore adopted instead.

Accuracy (A) was defined as the proportion of correctly predicted oriC intervals relative to the total number of genomes:

$$A = \frac{N_{correct}}{N_{total}}$$

where $N_{correct}$ represents the number of genomes for which the annotated oriC position falls within the predicted interval, and N_{total} is the total number of genomes.

However, this metric alone may favor larger prediction windows, as broader intervals have a higher probability of overlapping the annotated oriC region. It is well established that accuracy alone may provide a misleading assessment in tasks where prediction granularity varies [30]. Therefore, additional localization-oriented metrics were introduced to complement detection performance and assess spatial precision.

The circular distance error between the centers of the predicted and reference oriC intervals was computed as:

$$d = \min(|c_p - c_r|, L - |c_p - c_r|)$$

where c_p and c_r are the centers of the predicted and reference (experimentally confirmed) intervals, respectively, and L is the genome length. This formulation follows the standard definition of distance on circular range [31] and quantifies how close the prediction is to the true oriC location, independently of the window size.

The intersection-over-union (IoU) score, commonly used in localization tasks to jointly assess detection and spatial precision [32], [33], was used to assess the degree of overlap between predicted and reference intervals:

$$IoU = \frac{|\,Pred \cap Ref\,|}{|\,Pred \cup Ref\,|}$$

This metric penalizes overly large predictions and rewards precise localization.

Each metric was computed separately for each genome based on the predicted and annotated oriC intervals. Hit accuracy was then obtained as the proportion of genomes with overlapping predicted and reference intervals, while circular distance error and IoU values were summarized across genomes using mean and median values, with particular emphasis on the median as a robust measure against outliers.

Due to the circular nature of bacterial genomes, intervals may span the coordinate

origin. Such cases were handled by representing intervals in a circular coordinate system, and for overlap-based calculations, intervals crossing the genome boundary were split into segments to ensure correct computation of intersection and union lengths.

Together, these metrics enable a balanced evaluation by distinguishing between detection performance (whether the oriC region is captured) and localization precision (how accurately it is localized).

2.3. *ForiC Implementation*

The ForiC's source code is publicly accessible and can be downloaded via a publicly available http://bioinfo.matf.bg.ac.rs/home/downloads.wafl?cat=Software&project=ForiC.

Comprehensive instructions for the installation process, detailed usage options, and test cases are included in the provided link. Furthermore, ForiC is available online for testing purposes at http://www.markostojicevic.com/foric.

2.4. *Materials*

The dataset used in this study was retrieved from the NCBI database (https://www.ncbi.nlm.nih.gov/nuccore/) in April 2026. Genome records were selected using a stringent query designed to include only bacterial chromosomes with experimentally validated origins of replication, annotated with the feature key “rep_origin” in GenBank format. Although the query targeted a broad range of bacterial genomes annotated with “rep_origin”, only 85 records were retrieved, reflecting the limited availability of such annotations.

From this set, only circular chromosomes were retained, as cumulative GC skew patterns are most consistently defined and interpreted in circular genomes with bidirectional replication [34], [35]. Additional filtering was applied to ensure a consistent and unambiguous benchmark: duplicate entries cross-listed in NCBI under multiple accession codes (i.e., the same sequence indexed under more than one accession), as well as genomes with multiple or ambiguous “rep_origin” annotations that could not be mapped to a single continuous *oriC* region, were excluded. Furthermore, entries with computationally inferred *oriC* annotations (e.g., based on GC-skew analysis or automated annotation tools), sequence similarity-based annotations, or annotations lacking sufficient descriptive information were also excluded. The orientation of the *oriC* annotation (direct or complement) was not used as an exclusion criterion, as strand asymmetry in nucleotide composition remains consistent regardless of annotation direction [35].

Following this filtering procedure, 52 genomes were retained and classified into two groups. The primary dataset consisted of 36 genomes with experimentally validated *oriC* annotations and was used for parameter calibration and testing. The extended dataset comprised 16 additional genomes with curated but non-experimental *oriC* annotations, which were used exclusively to evaluate the robustness and generalizability of the method.

The dataset shows an uneven taxonomic distribution, with a predominance of Pseudomonadota genomes, reflecting the current availability of experimentally validated oriC annotations.

The primary dataset, used for parameter calibration and testing, is summarized in Table 1. The extended dataset is reported separately to avoid mixing annotations of different confidence levels. The complete list of genomes, including excluded entries and the specific reasons for their omission, is provided in Additional file 2. The relatively small size of the primary dataset reflects the limited availability of experimentally validated *oriC* annotations rather than an arbitrary selection bias, highlighting a broader limitation in current *oriC* benchmarking studies.

Table 1. Summary of the primary dataset (36 genomes), categorized by phylum

Phylum	N	Average length (bp)	Min. length (bp)	Max. length (bp)
Actinomycetota	1	9,239,851	9,239,851	9,239,851
Bacillota	3	3,128,281	1,991,969	5,346,304
Cyanobacteriota	1	8,680,294	8,680,294	8,680,294
Pseudomonadota	30	5,110,205	2,991,126	7,257,956
Spirochaetota	1	1,139,590	1,139,590	1,139,590
Total	**36**	**5,048,631**	**1,139,590**	**9,239,851**

2.4.1. *Preparation of training and test datasets*

The accuracy of ForiC was assessed by comparing predicted *oriC* positions to annotated reference positions obtained from GenBank records.

The primary dataset was divided into five stratified subsets based on phylum-level taxonomic distribution to improve representation of diverse bacterial groups across both training and test partitions. Five stratified splits were constructed to balance robustness of performance estimation against overreliance on any single data partition. In each split, 25 genomes (69.4%) were allocated to the training subset and 11 genomes (30.6%) to the testing subset. A 70:30 partitioning scheme was selected instead of the commonly used 80:20 ratio to retain a more informative test subset given the relatively limited dataset size, while preserving adequate data for parameter calibration.

Stratification at the phylum level was applied to preserve taxonomic diversity across subsets and reduce potential bias toward overrepresented groups. Within each phylum, genomes were allocated to the training and test partitions using a manual balancing procedure designed to preserve representation while approximating the target 70:30 proportion. For phyla represented by only a small number of genomes, exact proportional allocation was not always feasible, and assignments were adjusted to ensure that each phylum was represented in both training and test partitions whenever possible. Because the allocation was deterministic and manually balanced rather than stochastic; no random seed was used, and the resulting splits were used consistently throughout all calibration and evaluation analyses, and their full composition is provided in Additional file 2.

Table 2. Distribution of phyla per dataset, training/testing set

Phylum	**DATASET 1**		**DATASET 2**		**DATASET 3**		**DATASET 4**		**DATASET 5**	
	Training set	Testing set	Training set	Testing set	Training set	Testing set	Training set	Testing set	Training set	Testing set
Actinomycetota	0	1	1	0	1	0	0	1	1	0
Bacillota	2	1	2	1	2	1	2	1	2	1
Cyanobacteriota	1	0	0	1	1	0	1	0	0	1
Pseudomonadota	21	9	21	9	21	9	21	9	21	9
Spirochaetota	1	0	1	0	0	1	1	0	1	0
Total	**25**	**11**	**25**	**11**	**25**	**11**	**25**	**11**	**25**	**11**

The partitioning enabled systematic evaluation of different parameter configurations, including window size and GC/CGC-related thresholds. Different window lengths, defined as proportions of the genome size, were evaluated, along with alternative strategies for interpreting predicted regions, including one-window predictions and symmetrically expanded two-window reporting intervals.

Candidate window sizes (0.1%, 0.5%, 1%, 1.5%, and 2%) were calibrated using the training portions of each stratified subset. The final window-size setting was selected based on the most consistent overall performance across the training subsets and subsequently evaluated unchanged on the corresponding test subsets.

Evaluating parameter configurations across multiple stratified subsets reduced dependence on a single data partition and provided a more

reliable assessment of method performance across diverse genomic contexts.

Given the relatively limited size of the curated benchmark dataset (36 genomes) and the interval-localization nature of the task, performance was summarized using descriptive multi-metric comparisons across the five stratified subsets rather than formal null-hypothesis significance testing. This approach was considered more appropriate for emphasizing consistency of performance across multiple predefined partitions than inference from small-sample statistical tests.

3. RESULTS

3.1. Quality of ForiC Predictions and Performance Analysis

To validate the effectiveness of the ForiC method, a systematic evaluation was conducted on a dataset of bacterial genomes partitioned into five stratified datasets. The algorithm was applied using sliding window sizes ranging from 0.1% to 2% of the genome length. Performance was assessed using hit accuracy, defined as the proportion of cases in which the annotated oriC position falls within the predicted interval. All parameter tuning was performed exclusively on training subsets, while testing subsets were used for independent performance assessment, ensuring that the reported results were not influenced by parameter selection.

Performance was evaluated across all five stratified datasets and window sizes. Detailed accuracy values for both training and testing subsets are provided in Tables 3a and 3b, while additional evaluation metrics are summarized in Additional file 3. Across datasets, identical hit-accuracy values were frequently observed at 1%, 1.5%, and 2% window sizes, indicating that performance had reached a plateau once the reporting interval became sufficiently broad. Therefore, a window size of 1% was selected as the smallest configuration that consistently achieved maximal or near-maximal hit accuracy while preserving better spatial resolution than larger windows.

Among the evaluated splits, Dataset 3 exhibited one of the most balanced training–testing profiles at the selected 1% window size, with an absolute difference of 2.91 percentage points between training (88.00%) and testing (90.91%) hit accuracy under the expanded-interval criterion (Table 3b). At this configuration, Dataset 3 also maintained moderate center-distance values, typically in the range of approximately 6–30 kb (Additional file 3). Other datasets showed either larger train–test discrepancies at the same window size or less stable behavior across alternative window-size settings.

To illustrate these trends more clearly, Figure 3 presents the performance behavior for the test subset of Dataset 3. As shown in Figure 3a, hit accuracy for single-window predictions generally increased with window size. In contrast, when using the expanded prediction interval, hit accuracy increased up to a window size of 1% and then reached a plateau, with no further improvement observed for larger windows. Expanding the prediction interval to twice the window size, constructed by symmetric extension around the predicted center, increased hit accuracy under a broader interval criterion. In this configuration, hit accuracy reached up to 90.91% (10/11 genomes) in the testing subset, compared to substantially lower values for single-window predictions (e.g., 54.55% at 1% window size).

Additional evaluation metrics were used to assess localization performance. The center distance, summarized using the median, provided a robust measure of localization, whereas Intersection over Union (IoU), reported as the mean, remained low across configurations, generally on the order of 10^{-3}, reflecting the large size difference between predicted reporting intervals and experimentally annotated oriC regions (Additional file 3). The center distance remained unchanged between one-window and expanded-interval configurations, as the larger interval was generated by symmetric extension around the original prediction center. In contrast, IoU values were lower for the expanded interval configuration. Although a slightly lower center distance was observed at larger window sizes (e.g., 2%), this effect was not consistent across datasets and did not translate into consistently better overall performance. Accordingly, a window size of 1% was selected because it

provides the best balance between detection accuracy and spatial resolution.

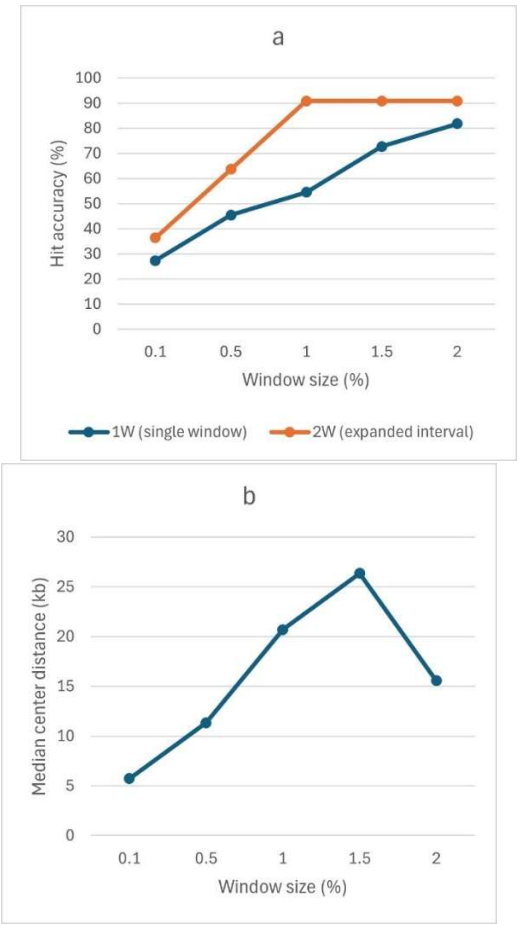


Figure 3: Effect of window size on oriC detection performance using the ForiC method, evaluated on the test subset of Dataset 3. (a) Hit accuracy for single-window (1W) predictions and for prediction intervals expanded to twice the window size (2W), constructed by symmetric extension around the predicted center. (b) Localization performance measured as median center distance between predicted and annotated oriC positions across different window sizes.

Based on these results, the 1% window-size configuration was selected for further analysis and comparison with existing methods.

Table 3a. Hit accuracy (%) for *oriC* prediction across five stratified datasets using single-window (1W) prediction intervals. Results are shown separately for training and testing subsets, along with the difference between them.

	Dataset 1			Dataset 2			Dataset 3			Dataset 4			Dataset 5		
Window size (%)	Training	Test	Absolute gap (pp)	Training	Test	Absolute gap (pp)	Training	Test	Absolute gap (pp)	Training	Test	Absolute gap (pp)	Training	Test	Absolute gap (pp)
0.1	40.00%	63.64%	23.64%	44.00%	54.55%	10.55%	56.00%	27.27%	28.73%	40.00%	63.64%	23.64%	52.00%	36.36%	15.64%
0.5	60.00%	45.45%	14.55%	52.00%	63.64%	11.64%	60.00%	45.45%	14.55%	52.00%	63.64%	11.64%	64.00%	36.36%	27.64%
1	60.00%	36.36%	23.64%	52.00%	54.55%	2.55%	52.00%	54.55%	2.55%	40.00%	81.82%	41.82%	68.00%	18.18%	49.82%
1.5	64.00%	81.82%	17.82%	76.00%	54.55%	21.45%	68.00%	72.73%	4.73%	72.00%	63.64%	8.36%	76.00%	54.55%	21.45%
2	72.00%	72.73%	0.73%	80.00%	54.55%	25.45%	68.00%	81.82%	13.82%	72.00%	72.73%	0.73%	84.00%	45.45%	38.55%

Table 3b. Hit accuracy (%) for *oriC* prediction across five stratified datasets using prediction intervals expanded to twice the window size (2W). Results are shown separately for training and testing subsets, along with the difference between them.

	Dataset 1			Dataset 2			Dataset 3			Dataset 4			Dataset 5		
Window size (%)	Training	Test	Absolute gap (pp)	Training	Test	Absolute gap (pp)	Training	Test	Absolute gap (pp)	Training	Test	Absolute gap (pp)	Training	Test	Absolute gap (pp)
0.1	56.00%	90.91%	34.91%	64.00%	72.73%	8.73%	80.00%	36.36%	43.64%	60.00%	81.82%	21.82%	72.00%	54.55%	17.45%
0.5	76.00%	90.91%	14.91%	80.00%	81.82%	1.82%	88.00%	63.64%	24.36%	76.00%	90.91%	14.91%	84.00%	72.73%	11.27%
1	88.00%	90.91%	2.91%	92.00%	81.82%	10.18%	88.00%	90.91%	2.91%	88.00%	90.91%	2.91%	96.00%	72.73%	23.27%
1.5	88.00%	90.91%	2.91%	92.00%	81.82%	10.18%	88.00%	90.91%	2.91%	88.00%	90.91%	2.91%	96.00%	72.73%	23.27%
2	88.00%	90.91%	2.91%	92.00%	81.82%	10.18%	88.00%	90.91%	2.91%	88.00%	90.91%	2.91%	96.00%	72.73%	23.27%

3.2. ForiC Comparison with Existing Models

3.2.1. Comparison with GC Skew and CGC Skew Baselines

To ensure fair comparison, all methods were evaluated under a unified framework by representing their outputs as prediction intervals of matched length and assessing them using identical scoring criteria. This avoids bias toward methods producing broader predictions and enables direct comparison of detection and localization performance.

Further analysis revealed that ForiC generally outperformed the baseline methods, namely the GC and CGC skew analyses. Table 4 presents a comparative analysis of ForiC against these methods on the testing dataset. For fair comparison, GC- and CGC-based predictions were evaluated using the same window sizes as the final ForiC configuration and comparable interval-length settings. For each method, the primary candidate window was identified from the corresponding skew signal, and the final prediction interval was defined either as a single window (1W) or as an interval of twice the window size (2W). In the extended evaluation (2W), ForiC used symmetric expansion around its predicted center, whereas GC and CGC used adjacent two-window intervals derived from their respective skew extrema. This avoids bias due to unequal interval size while preserving the native prediction logic of each method.

Under these conditions, the GC method demonstrated a maximum accuracy of 72.73%, whereas CGC showed substantially lower performance (Table 4, Additional file 4). In contrast, ForiC achieved a higher accuracy of 90.91%. In addition, ForiC demonstrated better localization, as reflected by lower center-distance values compared with GC and CGC methods. Notably, CGC predictions showed very large localization errors, with a median center distance of 896.33 kb, indicating substantial displacement from validated oriC regions. This likely reflects a known limitation of CGC skew, in which the global maximum may occur far from oriC in genomes with asymmetric compositional patterns or irregular skew profiles.

A representative example further illustrates these differences. Although the final evaluation is based on the expanded-interval configuration, this example demonstrates localization capability using a single-window prediction. Using a window size of 1% and a single-window prediction interval, the genome CP006262 (length: 5,402,276 bp) was analyzed. The experimentally validated oriC region is located between positions 4,544,042 and 4,544,273. The GC method predicted oriC between positions 4,483,826 and 4,537,848, whereas the CGC method generated a prediction substantially displaced from the annotated oriC region (3,241,320–3,295,342). In contrast, ForiC localized the oriC region within the interval 4,537,848–4,591,870, demonstrating higher positional precision.

Under the evaluated framework, ForiC showed higher hit accuracy and lower center-distance values than GC and CGC methods.

3.2.2. Comparison with OriFinder

In addition to intrinsic skew-based baselines, ForiC was compared with OriFinder 2022 (Additional file 4), a widely used oriC prediction tool that integrates compositional asymmetry, motif detection, and homology-based signals within a ranking framework for putative oriC regions.

For fair comparison, all OriFinder predictions were converted into fixed-length evaluation windows centered on each reported candidate, matching the window size used for ForiC. This removes biases arising from differences in predicted interval lengths and enables direct comparison of localization metrics.

When two OriFinder candidate regions were reported, each candidate was evaluated independently, and a prediction was considered correct if at least one normalized window overlapped the validated oriC region (Additional file 4).

OriFinder achieved 100% accuracy under the candidate-based criterion, meaning that at least one reported candidate overlapped the validated oriC region in all analyzed genomes. However, multiple candidate regions were returned in 54.55% of cases, whereas only 45.45% of genomes received a single prediction. In contrast,

Table 4. Comparison of ForiC, GC, and CGC methods for oriC prediction on the test subset of Dataset 3. Results are reported for both single-window (1W) and symmetrically expanded two-window (2W) prediction intervals using a window size of 1% of the genome length. For GC and CGC methods, the primary candidate window was identified based on the corresponding skew signal, and prediction intervals were constructed using the same strategy as for ForiC. Metrics include hit accuracy (%), median center distance (kb), and Intersection over Union (IoU).

	Prediction interval length of 1 window (1W)			Prediction interval length of 2 windows (2W)		
Method	**Accuracy (%)**	**Median center distance (kb)**	**Mean IoU**	**Accuracy (%)**	**Median center distance (kb)**	**Mean IoU**
ForiC	54.55%	20.70	0.0050	90.91%	20.70	0.0035
GC	27.27%	27.36	0.0010	72.73%	27.36	0.0034
CGC	9.09%	896.33	0.0020	18.18%	896.33	0.0018

Table 5. Comparison of ForiC and OriFinder performance on the test subset of Dataset 3. Results are reported for both single-window (1W) and symmetrically expanded two-window (2W) prediction intervals, using a window size corresponding to 1% of the genome length. Accuracy is defined as the overlap between predicted and experimentally validated oriC regions. For OriFinder, accuracy is calculated using a candidate-based criterion (i.e., at least one predicted candidate overlaps the validated region). For distance-based metrics, the minimum distance among candidate predictions was used. All interval-based metrics were computed using normalized prediction windows of identical size to ensure comparability.

	Prediction interval length of 1 window (1W)			Prediction interval length of 2 windows (2W)		
Method	**Accuracy (%)**	**Median center distance (kb)**	**Mean IoU**	**Accuracy (%)**	**Median center distance (kb)**	**Mean IoU**
ForiC	54.55%	20.70	0.0050	90.91%	20.70	0.0035
OriFinder	100%	0.04	0.0078	100%	0.04	0.0039

ForiC consistently generated one prediction per genome and achieved 54.55% accuracy in the single-window (1W) setting and 90.91% under the expanded two-window (2W) criterion. OriFinder also showed markedly lower median center distance (0.04 kb versus 20.70 kb), indicating substantially higher positional precision (Table 5).

OriFinder also achieved higher IoU values in the single-window setting, although this difference decreased after interval expansion. Overall, the two methods reflect different practical trade-offs: OriFinder provides strong candidate recovery and high positional precision but depends on external resources (e.g., BLAST searches) and may return multiple predicted regions, whereas ForiC provides a single intrinsically derived prediction without requiring prior annotation or database searches.

To further compare the behavior of both methods outside the benchmark dataset, an additional genome (AE000511, *Helicobacter pylori 26695*) was analyzed. This genome was not part of the training or evaluation datasets and does not have a precisely annotated oriC region in NCBI records. Both methods produced broadly similar predicted oriC locations, with interval centers at 1,642,783 for ForiC and 1,609,074 for OriFinder, indicating consistency in identifying candidate regions. However, substantial differences were observed in end-to-end execution time. The ForiC prediction was completed locally in 0.75 seconds on a standard workstation (Intel Core i7, 2.2 GHz, 8 GB RAM, SSD), whereas OriFinder, executed through its

public web interface, required approximately 1,680 seconds (28 minutes).

This difference likely reflects the more complex OriFinder workflow, which includes external searches such as BLAST-based homology detection and motif analysis, together with web-server execution overhead. In contrast, ForiC performs all computations directly on the input sequence without external dependencies.

Although limited to a single external genome, this example suggests that ForiC may be useful for rapid screening and large-scale analyses where lightweight execution is advantageous.

3.2.3. Robustness Under Local Sequence Perturbation

To further assess the robustness of the methods, selected genomes from the dataset were modified by removing nucleotide regions encompassing the experimentally validated oriC locations. The selected genomes include representatives from both the primary dataset (experimentally validated oriC annotations) and the extended dataset, allowing evaluation under both well-characterized and less certain annotation conditions. The specified subset of genomes was analyzed to illustrate different prediction behaviors under this perturbation.

The results (Table 6) indicate that ForiC maintains stable predictions even when the original oriC signal is partially removed, typically shifting the predicted region toward nearby compositional features. In contrast, OriFinder exhibits increased variability, often producing a larger number of candidate regions or shifting predictions to alternative high-scoring locations. In some cases, the number of reported candidates increased substantially, reflecting a stronger dependence on specific sequence features and external signals. While the table focuses on prediction stability and candidate variability, a complete overview of predictions for both original and modified sequences is available in Additional file 5.

For genomes CP026681 and AE015450, ForiC predicted oriC-centered intervals at coordinate 0 in both original and modified sequences, consistent with the circular genome representation in which coordinate 0 corresponds to the sequence boundary (wrap-around origin).

These findings suggest that ForiC appeared more stable in these selected perturbation examples to local disruptions in oriC-associated signals, while OriFinder's predictions are more sensitive to the presence of specific functional elements and homology-based cues.

3.3. Exploratory Consistency Analysis on the Extended Dataset

To further examine method behavior on genomes lacking consistently validated oriC annotations, an exploratory analysis was performed on the extended dataset. Because reliable reference coordinates were not uniformly available, formal accuracy assessment was not possible. Instead, positional agreement and prediction consistency across methods were examined.

In this analysis, ForiC was compared with GC skew, cumulative GC skew (CGC), and OriFinder to examine their behavior under less certain annotation conditions. The results are summarized in Table 7.

OriFinder demonstrated the highest level of agreement with ForiC, achieving 100% agreement and a very low mean distance of 3.26 kb, indicating that its predicted oriC locations are consistently close to those identified by ForiC. However, it produced an average of 1.73 candidate regions per genome, reflecting ambiguity in its predictions.

GC skew exhibited a relatively high agreement rate (80%), but with a substantially larger mean distance (202.65 kb), indicating the presence of significant outliers and reduced stability. CGC skew showed lower agreement (26.67%) and moderate distances, suggesting weaker consistency with ForiC predictions.

These exploratory results indicate that OriFinder and GC predictions were often positioned near ForiC predictions, whereas CGC showed lower positional consistency. Because experimentally validated references were unavailable for this dataset, these findings should be interpreted descriptively rather than as comparative accuracy rankings.

Table 6. Robustness analysis of ForiC and OriFinder predictions on selected genomes from the primary and extended datasets. Modified sequences were generated by removing genomic regions encompassing the experimentally validated oriC locations. Reported values correspond to the centers of predicted oriC regions, allowing direct comparison of prediction stability. While ForiC maintains relatively stable predictions with minor shifts in location, OriFinder exhibits increased variability, often producing a higher number of candidate regions after modification, reflecting its sensitivity to local sequence features and external signals.

Genome	Dataset	Removed region (bp)	ForiC center (original)	ForiC center (modified)	OriFinder candidates (original)	OriFinder candidates (modified)
CP006027	Primary	4,600,000-4,700,000	4,719,832	3,044,508	2	1
CP009273	Primary	3,870,000-3,940,000	3,959,847	3,899,997	2	3
CP026681	Extended	0-3,000	0	0	1	21
AE015450	Extended	0-7,000	0	0	2	4

Table 7. Comparative analysis of ForiC, GC skew, CGC skew, and OriFinder on the extended dataset. As experimentally validated oriC annotations are not consistently available, the evaluation is based on positional agreement relative to ForiC predictions. Mean and median distances represent circular distances between the centers of predicted oriC regions and the corresponding ForiC prediction centers. Agreement is defined as the percentage of predictions located within a window corresponding to 1% of the genome length centered at the ForiC prediction. For OriFinder, the minimum distance among candidate predictions was used, and the average number of candidates per genome is reported.

Metric	GC	CGC	OriFinder
Mean distance to ForiC (kb)	202.65	100.08	3.26
Median distance to ForiC (kb)	4.99	41.12	3.56
Agreement with ForiC (%)	80.00	26.67	100.00
Mean number of candidates	1	1	1.73

4. Discussion

The performance metrics obtained during method development and testing indicate that ForiC achieves higher interval-based hit accuracy than GC and CGC skew baselines on the evaluated datasets. The method achieved a testing accuracy of 90.91%, exceeding that of the baseline approaches. Combined with comparable computational efficiency, these results suggest that the proposed method showed encouraging performance on held-out subsets and highlight the importance of integrating multiple genomic characteristics in oriC prediction.

Despite the high hit accuracy achieved, a key challenge remains in balancing interval size and positional precision. The improved performance observed with two-window reporting intervals indicates that the method can tolerate moderate positional uncertainty, which may be advantageous in practical screening applications. This improvement is also consistent with cases in which the oriC region spans neighboring sliding-window boundaries rather than being fully contained within a single window. However, the reported prediction intervals are intentionally designed for regional localization rather than exact base-pair delineation of oriC boundaries. The observed median center distance (20.70 kb) indicates that predicted oriC locations are typically placed within the reported interval,

supporting the stability of the approach. This reflects the design of the method, which prioritizes reliable regional localization of oriC-containing regions over nucleotide-level boundary prediction. Accordingly, ForiC is best viewed as a screening-oriented interval-localization framework that complements, rather than replaces, higher-resolution reference-assisted approaches. This interpretation is further supported by performance trends across different window sizes (Figure 3), where hit accuracy reaches a plateau beyond 1% of the genome length, while no consistent improvement in localization is observed. These results suggest diminishing returns with increasing window size, indicating that further interval expansion is unlikely to provide substantial benefit and may reduce spatial resolution.

While the previous analysis highlights the trade-off between interval size and positional precision, it is also important to consider how the proposed method compares with reference-based approaches such as OriFinder. While OriFinder demonstrated high positional precision and strong agreement with ForiC predictions, it frequently reported multiple candidate regions, requiring additional selection to identify the most likely oriC location. In contrast, ForiC consistently produced a single prediction per genome, providing a clear and directly interpretable result. This reflects a trade-off between positional resolution and decisiveness, where ForiC offers a streamlined and unambiguous prediction framework.

Performance varied across the taxonomically stratified benchmark subsets, which may reflect differences in genomic composition across bacterial lineages. Broader validation across additional phyla and larger balanced datasets is therefore required before stronger generalizability claims can be made.

The proposed method offers several practical advantages for computational genomics. In addition to improving upon simple compositional approaches such as GC and CGC skews, ForiC provides a computationally efficient and self-contained framework that does not rely on external databases or prior annotation. While reference-based tools such as OriFinder can achieve higher positional precision in certain cases, this is partly attributable to their use of external biological resources and functional annotations. In contrast, ForiC focuses on identifying oriC-containing regions using intrinsic genomic signals, including compositional asymmetry and structural transitions reflected by inflection points. These features are consistent with known characteristics of bacterial replication origin regions, which are often associated with shifts in nucleotide composition and replication-related genome organization. The ability of ForiC to detect such signals across diverse genomes suggests that the method may capture biologically meaningful patterns related to DNA replication, although the precise role of individual features, particularly inflection points, requires further investigation.

Additional analyses on the extended dataset provide exploratory evidence of positional consistency across methods. Despite the absence of consistently validated oriC annotations, ForiC predictions showed general positional agreement with both compositional methods and OriFinder, indicating that the method captures biologically relevant signals beyond the training dataset.

Furthermore, robustness experiments involving the removal of oriC-associated regions suggested that ForiC predictions were comparatively stable in the selected perturbation experiments. These findings indicate that ForiC is less sensitive to local disruptions in sequence features and may provide more consistent predictions under uncertain or incomplete annotation conditions.

Nevertheless, several limitations should be acknowledged. The relatively modest number of genomes used in this study may limit the generalizability of the findings, highlighting the need for larger and more diverse datasets in future work. In addition, while ForiC effectively identifies oriC-containing regions, further improvements are needed to refine positional precision without compromising computational efficiency. The preliminary screening stage could also be extended in future work by incorporating sequence descriptors that account for nucleotide-order information. The proposed method relies on genomic features such as inflection points, whose biological interpretation and consistency across

different bacterial groups warrant further investigation. While these features contribute to improved prediction performance, their role in oriC localization is not yet fully understood, and the isolated contribution of the inflection-point feature was not separately evaluated in the present study. Future work should explore the relationship between inflection point patterns and underlying replication mechanisms, assess their independent predictive value, and evaluate their stability across larger and more diverse genomic datasets.

In conclusion, ForiC provides a lightweight intrinsic framework for oriC-region prediction that complements existing reference-assisted methods and may be particularly useful for rapid large-scale bacterial genome screening.

5. Acknowledgments, Statements and Declarations

5.1. Acknowledgments

I would like to thank my mentor, Dr Nenad Mitić, for reviewing the manuscript and providing valuable suggestions that significantly improved its quality.

5.2. Author Contributions

The author is solely responsible for all aspects of this manuscript. This includes conceptualization, methodology, formal analysis, investigation, data curation, writing – original draft, writing – review & editing, visualization, and project administration.

5.3. Data Availability Statement

The code developed for the method, which was used to produce the results presented in this study, is freely available at the following link: http://bioinfo.matf.bg.ac.rs/home/downloads.wafl?cat=Software&project=ForiC.

The application itself is publicly accessible at http://markostojicevic.com/foric/.

The data are provided within the manuscript or supplementary information files. Additionally, the data and code can be obtained from the author upon reasonable request via email.

5.4. Funding

No funds, grants, or other support were received for conducting this study.

5.5. Competing Interests

The author(s) declare that they have no competing interests.

5.6. Ethics Approval and Consent to Participate

Not applicable.

5.7. Consent for Publication

Not applicable.

6. References

[1] S. P. Bell and K. Labib, “Chromosome Duplication in *Saccharomyces cerevisiae*,” *Genetics*, vol. 203, no. 3, pp. 1027–1067, Jul. 2016, doi: 10.1534/genetics.115.186452.

[2] C. Lou *et al.*, “sefOri: selecting the best-engineered sequence features to predict DNA replication origins,” *Bioinformatics*, vol. 36, no. 1, pp. 49–55, Jan. 2020, doi: 10.1093/bioinformatics/btz506.

[3] M. O’Donnell, L. Langston, and B. Stillman, “Principles and Concepts of DNA Replication in Bacteria, Archaea, and Eukarya,” *Cold Spring Harb. Perspect. Biol.*, vol. 5, no. 7, pp. a010108–a010108, Jul. 2013, doi: 10.1101/cshperspect.a010108.

[4] N. P. Robinson and S. D. Bell, “Origins of DNA replication in the three domains of life,” *FEBS J.*, vol. 272, no. 15, pp. 3757–3766, Aug. 2005, doi: 10.1111/j.1742-4658.2005.04768.x.

[5] R. R. Spinks *et al.*, “DnaB helicase dynamics in bacterial DNA replication resolved by single-molecule studies,” *Nucleic Acids Res.*, vol. 49, no. 12, pp. 6804–6816, Jul. 2021, doi: 10.1093/nar/gkab493.

[6] M. Wolanski, R. Donczew, A. Zawilak-Pawlik, and J. Zakrzewska-CzerwiÅ„ska, “oriC-encoded instructions for the initiation of bacterial chromosome replication,” *Front. Microbiol.*, vol. 5, Jan. 2015, doi: 10.3389/fmicb.2014.00735.

[7] F. Bleichert, M. R. Botchan, and J. M. Berger, “Mechanisms for initiating cellular DNA replication,” *Science*, vol. 355, no. 6327, p. eaah6317, Feb. 2017, doi: 10.1126/science.aah6317.

[8] T. T. Richardson, D. Stevens, S. Pelliciari, O. Harran, T. Sperlea, and H. Murray, “Identification of a basal system for unwinding a bacterial chromosome origin,” *EMBO J.*, vol. 38, no. 15, p. e101649, Aug. 2019, doi: 10.15252/embj.2019101649.

[9] P. Mackiewicz, “Where does bacterial replication start? Rules for predicting the oriC region,” *Nucleic Acids Res.*, vol. 32, no. 13, pp. 3781–3791, Jul. 2004, doi: 10.1093/nar/gkh699.

[10] J. R. Lobry, “Asymmetric substitution patterns in the two DNA strands of bacteria,” *Mol. Biol. Evol.*, vol. 13, no. 5, pp. 660–665, May 1996, doi: 10.1093/oxfordjournals.molbev.a025626.

[11] E. R. M. Tillier and R. A. Collins, “The Contributions of Replication Orientation, Gene Direction, and Signal Sequences to Base-Composition Asymmetries in Bacterial Genomes,” *J. Mol. Evol.*, vol. 50, no. 3, pp. 249–257, Mar. 2000, doi: 10.1007/s002399910029.

[12] A. Marín and X. Xia, “GC skew in protein-coding genes between the leading and lagging strands in bacterial genomes: New substitution models incorporating strand bias,” *J. Theor. Biol.*, vol. 253, no. 3, pp. 508–513, Aug. 2008, doi: 10.1016/j.jtbi.2008.04.004.

[13] J. Lu and S. L. Salzberg, “SkewIT: The Skew Index Test for large-scale GC Skew analysis of bacterial genomes,” *PLOS Comput. Biol.*, vol. 16, no. 12, p. e1008439, Dec. 2020, doi: 10.1371/journal.pcbi.1008439.

[14] M. J. McLean, K. H. Wolfe, and K. M. Devine, “Base Composition Skews, Replication Orientation, and Gene Orientation in 12 Prokaryote Genomes,” *J. Mol. Evol.*, vol. 47, no. 6, pp. 691–696, Dec. 1998, doi: 10.1007/PL00006428.

[15] K. Arakawa and M. Tomita, “The GC Skew Index: A Measure of Genomic Compositional Asymmetry and the Degree of Replicational Selection,” *Evol. Bioinforma.*, vol. 3, p. 117693430700300006, Jan. 2007, doi: 10.1177/117693430700300006.

[16] H. Luo, C.-L. Quan, C. Peng, and F. Gao, “Recent development of Ori-Finder system and DoriC database for microbial replication origins,” *Brief. Bioinform.*, vol. 20, no. 4, pp. 1114–1124, Jul. 2019, doi: 10.1093/bib/bbx174.

[17] F. Gao and C.-T. Zhang, “Ori-Finder: A web-based system for finding oriC s in unannotated bacterial genomes,” *BMC Bioinformatics*, vol. 9, no. 1, p. 79, Dec. 2008, doi: 10.1186/1471-2105-9-79.

[18] A. Grigoriev, “Analyzing genomes with cumulative skew diagrams,” *Nucleic Acids Res.*, vol. 26, no. 10, pp. 2286–2290, May 1998, doi: 10.1093/nar/26.10.2286.

[19] H.-Y. Ou, F.-B. Guo, and C.-T. Zhang, “Analysis of nucleotide distribution in the genome of *Streptomyces coelicolor* A3(2) using the Z curve method,” *FEBS Lett.*, vol. 540, no. 1–3, pp. 188–194, Apr. 2003, doi: 10.1016/S0014-5793(03)00263-1.

[20] R. Zhang and C.-T. Zhang, “A Brief Review: The Z-curve Theory and its Application in Genome Analysis,” *Curr. Genomics*, vol. 15, no. 2, pp. 78–94, Apr. 2014, doi: 10.2174/1389202915999140328162433.

[21] R. Zhang and C.-T. Zhang, “Single replication origin of the archaeon Methanosarcina mazei revealed by the Z curve method,” *Biochem. Biophys. Res. Commun.*, vol. 297, no. 2, pp. 396–400, Sep. 2002, doi: 10.1016/S0006-291X(02)02214-3.

[22] A. Zawilak, “Identification of a putative chromosomal replication origin from Helicobacter pylori and its interaction with the initiator protein DnaA,” *Nucleic Acids Res.*, vol. 29, no. 11, pp. 2251–2259, Jun. 2001, doi: 10.1093/nar/29.11.2251.

[23] N. V. Sernova and M. S. Gelfand, “Identification of replication origins in prokaryotic genomes,” *Brief. Bioinform.*, vol. 9, no. 5, pp. 376–391, Sep. 2008, doi: 10.1093/bib/bbn031.

[24] M.-J. Dong, H. Luo, and F. Gao, “DoriC 12.0: an updated database of replication origins in both complete and draft prokaryotic genomes,” *Nucleic Acids Res.*, vol. 51, no. D1, pp. D117–D120, Jan. 2023, doi: 10.1093/nar/gkac964.

[25] M.-J. Dong, H. Luo, and F. Gao, “Ori-Finder 2022: A Comprehensive Web Server for Prediction and Analysis of Bacterial Replication Origins,” *Genomics Proteomics Bioinformatics*, vol. 20, no. 6, pp. 1207–1213, Dec. 2022, doi: 10.1016/j.gpb.2022.10.002.

[26] Y. Li and F. Gao, “OriV-Finder: a comprehensive web server for bacterial plasmid replication origin analysis,” *Nucleic Acids Res.*, vol. 53, no. W1, pp. W451–W456, Jul. 2025, doi: 10.1093/nar/gkaf341.

[27] S. M. Azim, Md. R. Haque, and S. Shatabda, “OriC-ENS: A sequence-based ensemble classifier for predicting origin of replication in S. cerevisiae,” *Comput. Biol. Chem.*, vol. 92, p. 107502, Jun. 2021, doi: 10.1016/j.compbiolchem.2021.107502.

[28] Z. Van Meel and J. A. Baaijens, “ORCA: Predicting replication origins in circular prokaryotic chromosomes,” Mar. 30, 2024, *Bioinformatics*. doi: 10.1101/2024.03.28.587133.

[29] J. Stewart, *Calculus: concepts and contexts*, Fourth edition, Enhanced edition. Boston, MA, USA: Cengage, 2019.

[30] D. Powers, “Evaluation: From Precision, Recall and F-Measure to ROC, Informedness, Markedness & Correlation,” *J. Mach. Learn. Technol.*, vol. 2, pp. 37–63, 2011.

[31] K. V. Mardia and P. E. Jupp, *Directional Statistics*, 1st ed. in Wiley Series in Probability and Statistics. Wiley, 1999. doi: 10.1002/9780470316979.

[32] M. Everingham, L. Van Gool, C. K. I. Williams, J. Winn, and A. Zisserman, “The Pascal Visual Object Classes (VOC) Challenge,” *Int. J. Comput. Vis.*, vol. 88, no. 2, pp. 303–338, Jun. 2010, doi: 10.1007/s11263-009-0275-4.

[33] M. A. Rahman and Y. Wang, “Optimizing Intersection-Over-Union in Deep Neural Networks for Image Segmentation,” in *Advances in Visual Computing*, vol. 10072, G. Bebis, R. Boyle, B. Parvin, D. Koracin, F. Porikli, S. Skaff, A. Entezari, J. Min, D. Iwai, A. Sadagic, C. Scheidegger, and T. Isenberg, Eds., in Lecture Notes in Computer Science, vol. 10072. , Cham: Springer International Publishing, 2016, pp. 234–244. doi: 10.1007/978-3-319-50835-1_22.

[34] P. Worning, L. J. Jensen, P. F. Hallin, H. Stærfeldt, and D. W. Ussery, “Origin of replication in circular prokaryotic chromosomes,” *Environ. Microbiol.*, vol. 8, no. 2, pp. 353–361, Feb. 2006, doi: 10.1111/j.1462-2920.2005.00917.x.

[35] E. P. C. Rocha, “The replication-related organization of bacterial genomes,” *Microbiology*, vol. 150, no. 6, pp. 1609–1627, Jun. 2004, doi: 10.1099/mic.0.26974-0.

Reviewers:

Australia
Abramov, Vyacheslav; Monash University
Begg, Rezaul; Victoria University
Bem, Derek; University of Western Sydney
Betts, Christopher; Pegacat Computing Pty. Ltd.
Buyya, Rajkumar; The University of Melbourne
Chapman, Judith; Australian University Limited
Chen, Yi-Ping Phoebe; Deakin University
Hammond, Mark; Flinders University
Henman, Paul; University of Queensland
Palmisano, Stephen; University of Wollongong
Ristic, Branko; Science and Technology Organisation
Sajjanhar, Atul; Deakin University
Sidhu, Amandeep; University of Technology, Sydney
Sudweeks, Fay; Murdoch University

Austria
Derntl, Michael; University of Vienna
Hug, Theo; University of Innsbruck
Loidl, Susanne; Johannes Kepler University
Linz Stockinger, Heinz; University of Vienna
Sutter, Matthias; University of Innsbruck

Brazil
Parracho, Annibal; Universidad Federal
Fluminense Traina, Agma; University of Sao Paulo
Traina, Caetano; University of Sao Paulo
Vicari, Rosa; Federal University of Rio Grande

Belgium
Huang, Ping; European Commission

Canada
Fung, Benjamin; Simon Fraser University
Grayson, Paul; York University Gray,
Bette; Alberta Education
Memmi, Daniel; UQAM
Neti, Sangeeta; University of Victoria
Nickull, Duane; Adobe Systems, Inc. Ollivier-Gooch,
Carl; The University of British Columbia Paulin,
Michele; Concordia University
Plaisent, Michel; University of Quebec Reid, Keith;
Ontario Ministry of Agriculture Shewchenko,
Nicholas; Biokinetics and Associates Steffan,
Gregory; University of Toronto Vandenberghe,
Christian; HEC Montreal

Czech Republic
Kala, Zdenek; Brno University of Technology
Korab, Vojtech; Brno University of technology
Lhotska, Lenka; Czech Technical University

Finland
Lahdelma, Risto; University of Turku
Salminen, Pekka; University of
Jyvaskyla

France
Cardey, Sylviane; University of Franche-Comte
Klinger, Evelyne; LTCI – ENST, Paris
Roche, Christophe; University of Savoie
Valette, Robert; LAAS - CNRS

Germany
Accorsi, Rafael; University of Freiburg
Glatzer, Wolfgang; Goethe-University
Gradmann, Stefan; Universitat Hamburg
Groll, Andre; University of Siegen
Klamma, Ralf; RWTH Aachen University
Wurtz, Rolf P.; Ruhr-Universitat Bochum

Hungary
Mester, Gyula; Óbuda University, Budapest

India
Pareek, Deepak; Technology4Development
Scaria, Vinod; Institute of Integrative Biology
Shah, Mugdha; Mansukhlal Svayam

Ireland
Eisenberg, Jacob; University College Dublin

Israel
Feintuch, Uri; Hadassah-Hebrew University

Italy
Badia, Leonardo; IMT Institute for Advanced
Studies Berrittella, Maria; University of Palermo
Carpaneto, Enrico; Politecnico di Torino

Japan
Hattori, Yasunao; Shimane University
Livingston, Paisley; Linghan University
Srinivas, Hari; Global Development Research Center
Obayashi, Shigeru; Institute of Fluid Science,
Tohoku University

Netherlands
Mills, Melinda C.; University of Groningen
Pires, Luís Ferreira; University of Twente

New Zealand
Anderson, Tim; Van Der Veer Institute

Portugal
Cardoso, Jorge; University of Madeira
Natividade, Eduardo; Polytechnic Institute of
Coimbra Oliveira, Eugenio; University of Porto

Singapore
Tan, Fock-Lai; Nanyang Technological University

South Korea
Kwon, Wook Hyun; Seoul National University

Spain
Barrera, Juan Pablo Soto; University of Castilla
Gonzalez, Evelio J.; University of La Laguna
Perez, Juan Mendez; Universidad de La
Laguna Royuela, Vicente; Universidad de
Barcelona Vizcaino, Aurora; University of
Castilla-La Mancha Vilarrasa, Clelia Colombo;
Open University of Catalonia

Sweden
Johansson, Mats; Royal Institute of Technology

Switzerland
Niinimaki, Marko; Helsinki Institute of Physics
Pletka, Roman; AdNovum Informatik AG
Rizzotti, Sven; University of Basel
Specht, Matthias; University of Zurich

Taiwan
Lin, Hsiung Cheng; Chienkuo Technology University
Shyu, Yuh-Huei; Tamkang University
Sue, Chuan-Ching; National Cheng Kung
University

United Kingdom
Ariwa, Ezendu; London Metropolitan University
Biggam, John; Glasgow Caledonian University
Coleman, Shirley; University of Newcastle
Conole, Grainne; University of Southampton
Dorfler, Viktor; Strathclyde University
Engelmann, Dirk; University of London
Eze, Emmanuel; University of Hull
Forrester, John; Stockholm Environment Institute
Jensen, Jens; STFC Rutherford Appleton
Laboratory Kolovos, Dimitrios S.; The University
of York McBurney, Peter; University of Liverpool
Vetta, Atam; Oxford Brookes University
WHYTE, William Stewart; University of Leeds
Xie, Changwen; Wicks and Wilson Limited

USA
Bach, Eric; University of Wisconsin Bolzendahl,
Catherine; University of California Bussler,
Christoph; Cisco Systems, Inc. Charpentier,
Michel; University of New Hampshire Chong,
Stephen; Cornell University
Collison, George; The Concord Consortium
DeWeaver, Eric; University of Wisconsin -
Madison Gans, Eric; University of California
Gill, Sam; San Francisco State University
Hunter, Lynette; University of California Davis
Iceland, John; University of Maryland
Kaplan, Samantha W.; University of Wisconsin
Langou, Julien; The University of Tennessee
Liu, Yuliang; Southern Illinois University
Edwardsville Lok, Benjamin; University of Florida
Minh, Chi Cao; Stanford University
Morrissey, Robert; The University of Chicago
Mui, Lik; Google, Inc
Rizzo, Albert ; University of Southern
California Rosenberg, Jonathan M. ; University
of Maryland Shaffer, Cliff ; Virginia Tech
Sherman, Elaine; Hofstra University
Snyder, David F.; Texas State University
Song, Zhe; University of Iowa
Wei, Chen; Intelligent Automation, Inc.
Yu, Zhiyi; University of California

Welcome to IPSI Conferences and Journals!

http://tir.ipsitransactions.org

http://www.ipsitransactions.org

CIP – Katalogizacija u publikaciji
Narodna biblioteka Srbije, Beograd
ISSN 1820 – 4503
The IPSI Transactions
on Internet Research
COBISS.SR - ID 119127052

