Computational Mathematics for AI: Numerical Methods and Distributed Computing for Deep Learning on Big Data

BEN TROVATO* and G.K.M. TOBIN*, Institute for Clarity in Documentation, USA
LARS THØRVÄLD, The Thørväld Group, Iceland
VALERIE BÉRANGER, Inria Paris-Rocquencourt, France
APARNA PATEL, Rajiv Gandhi University, India
HUIFEN CHAN, Tsinghua University, China
CHARLES PALMER, Palmer Research Laboratories, USA
JOHN SMITH, The Thørväld Group, Iceland
JULIUS P. KUMQUAT, The Kumquat Consortium, USA

A clear and well-documented LaTeX document is presented as an article formatted for publication by ACM in a conference proceedings or journal publication. Based on the "acmart" document class, this article presents and explains many of the common variations, as well as many of the formatting elements an author may use in the preparation of the documentation of their work.

CCS Concepts: • **Do Not Use This Code → Generate the Correct Terms for Your Paper**; *Generate the Correct Terms for Your Paper*; Generate the Correct Terms for Your Paper; Generate the Correct Terms for Your Paper.

Additional Key Words and Phrases: Do, Not, Us, This, Code, Put, the, Correct, Terms, for, Your, Paper

ACM Reference Format:
Ben Trovato, G.K.M. Tobin, Lars Thørväld, Valerie Béranger, Aparna Patel, Huifen Chan, Charles Palmer, John Smith, and Julius P. Kumquat. 2025. Computational Mathematics for AI: Numerical Methods and Distributed Computing for Deep Learning on Big Data. *J. ACM* 00, 0, Article 000 (2025), 45 pages. https://doi.org/XXXXXXX.XXXXXXX

1 Introduction

This document outlines the protocol for a *systematic literature review* (SLR) on computational mathematics for AI, focusing on numerical methods and distributed computing techniques for deep learning on big data. The review will follow the PRISMA guidelines [35] and Kitchenham's methodology for SLRs in software engineering [26].

*Both authors are people.

Authors' Contact Information: Ben Trovato, trovato@corporation.com; G.K.M. Tobin, webmaster@marysville-ohio.com, Institute for Clarity in Documentation, Dublin, Ohio, USA; Lars Thørväld, The Thørväld Group, Hekla, Iceland, larst@affiliation.org; Valerie Béranger, Inria Paris-Rocquencourt, Rocquencourt, France; Aparna Patel, Rajiv Gandhi University, Doimukh, Arunachal Pradesh, India; Huifen Chan, Tsinghua University, Haidian Qu, Beijing Shi, China; Charles Palmer, Palmer Research Laboratories, San Antonio, Texas, USA, cpalmer@prl.com; John Smith, The Thørväld Group, Hekla, Iceland, jsmith@affiliation.org; Julius P. Kumquat, The Kumquat Consortium, New York, USA, jpkumquat@consortium.net.

ACM 1557-735X/2025/13-ART000
https://doi.org/XXXXXXX.XXXXXXX

2 Background

2.1 Terminology and Definitions

To ensure clarity throughout this analysis, we define the following key technical terms:

- **Computational Mathematics for AI**: The application of mathematical techniques and algorithms to solve computational problems in artificial intelligence.
- **Numerical Methods**: Algorithms that use numerical approximation for the problems of mathematical analysis.
- **Optimization**: The selection of the best element from a set of available alternatives according to some criteria.
- **Nature-Inspired Algorithms**: A class of metaheuristic optimization algorithms inspired by natural phenomena, such as biological evolution, animal behavior, or ecological processes, used to solve complex, non-convex optimization problems.
- **Bayesian Optimization**: An optimization technique that builds a probabilistic model of the objective function and uses it to select the most promising points to evaluate, aiming to find the global optimum with a minimal number of evaluations.
- **Multi-Objective Optimization**: An optimization framework where two or more conflicting objectives are optimized simultaneously, resulting in a set of trade-off solutions known as the Pareto front.
- **Hardware-Aware Optimization**: An approach to optimization that explicitly considers the computational characteristics and constraints of the target hardware platform, aiming to maximize performance, efficiency, or resource utilization.
- **Distributed Computing**: A computing paradigm where multiple computers work together to solve computational problems across a network.
- **Federated Learning**: A decentralized machine learning approach where multiple devices collaboratively train a shared model without exchanging raw data, thereby preserving privacy and reducing data transfer requirements.
- **Deep Learning**: A subset of machine learning using neural networks with multiple layers to extract higher-level features from raw input.
- **Big Data**: Data sets that are too large or complex to be dealt with by traditional data-processing application software.
- **Systematic Literature Review (SLR)**: A structured and methodical approach to identifying, evaluating, and synthesizing existing research studies on a specific topic, following predefined protocols to ensure transparency, repeatability, and minimization of bias.
- **PRISMA (Preferred Reporting Items for Systematic Reviews and Meta-Analyses)**: A set of evidence-based guidelines and a reporting framework designed to enhance the quality and transparency of systematic reviews and meta-analyses.
- **Meta-Analysis**: A statistical technique for combining results from multiple independent studies to identify patterns, derive general conclusions, or estimate the overall effect size with greater precision.

2.2 Rationale

Deep learning has revolutionized a wide spectrum of big data applications by providing state-of-the-art performance across domains. However, the increasing complexity of deep neural networks and the growing scale of datasets demand a deeper investigation into the computational methods that support such systems. Among these, numerical methods and distributed computing stand out as critical enablers for efficient model training and deployment at scale.

Numerical methods, particularly optimization algorithms, form the foundation for training deep neural networks, playing a key role in minimizing loss functions and ensuring convergence. As Najafabadi et al. [36] emphasize, integrating deep learning with big data analytics introduces significant challenges, particularly in scalability and computational efficiency. This intersection calls for sophisticated mathematical tools to address issues related to high-dimensional data, model complexity, and computational constraints.

In parallel, distributed computing has emerged as an essential strategy for managing the large-scale computations involved in modern deep learning. Yan [44] highlights both the theoretical underpinnings and real-world implementations of such methods, underscoring the role of parallelism, GPU acceleration, federated learning, and other distributed paradigms. These techniques allow workloads to be distributed across clusters or edge devices, reducing training time and improving scalability.

Taken together, numerical optimization and distributed computing are not just complementary, they are interdependent components of scalable deep learning systems. Their combined use offers a path forward for developing more efficient, robust, and deployable AI solutions for big data scenarios.

This review synthesizes the current body of knowledge at the intersection of computational mathematics and scalable deep learning, providing a structured assessment of the most impactful methods in this space.

2.3 Objectives

The core objectives of this SLR are as follows:

(1) To identify and categorize cutting-edge numerical methods applied in deep learning for big data applications.
(2) To evaluate the effectiveness of distributed computing techniques in scaling deep learning systems.
(3) To compare these methods with respect to computational efficiency, scalability, and model accuracy.
(4) To uncover emerging trends and identify open research challenges at the convergence of numerical mathematics and distributed AI computing.

3 Related work

A considerable amount of research has focused on enhancing deep learning's computational efficiency and scalability, particularly through advancements in numerical methods and distributed computing. The work by Najafabadi et al. [36] provides an extensive overview of deep learning applications in big data analytics, emphasizing the inherent challenges in managing large-scale data and the computational power required. The authors discuss various deep learning architectures and the specific numerical methods used to optimize these models, setting a foundation for understanding the computational needs of big data-driven deep learning.

The book by Yan [44] presents a comprehensive discussion on the computational methods for deep learning, detailing the theoretical aspects of optimization algorithms and their implementation in practical scenarios. This work bridges the gap between theory and practical deployment, offering insights into the challenges of implementing these methods in a distributed environment. The book highlights the importance of selecting appropriate numerical methods to ensure both convergence and computational efficiency.

A survey by Zhang et al. [46] delves into distributed deep learning frameworks, discussing the evolution from traditional distributed machine learning to more sophisticated distributed deep

learning systems. It explores various distributed computing techniques such as federated learning, GPU acceleration, and parallel processing, which are essential for scaling deep learning models for big data applications. The survey compares different distributed frameworks, analyzing their scalability, efficiency, and suitability for diverse deep learning tasks.

Similarly, Li et al. [32] provides a foundational overview of federated learning, a decentralized approach to training models without sharing raw data between nodes. This technique is especially useful for privacy-sensitive applications in big data. The authors discuss federated learning's architecture, key challenges, and promising results in scaling deep learning for real-world applications.

Li et al. [33] offers a detailed survey of scalable deep learning techniques, specifically focusing on efficient parallel processing and distributed systems. The work discusses both hardware-based approaches, such as GPU acceleration, and software-based frameworks like Apache Spark, which have shown promise in reducing the computational time required for large-scale models, making deep learning more feasible for real-time applications.

Further, Ben and Waller [6] provides insights into optimization methods specifically tailored for big data in deep learning. The authors review key numerical methods and optimization algorithms, addressing their impact on model convergence and performance. This paper is particularly valuable for understanding the trade-offs between computational cost and accuracy, which are central to deep learning in big data contexts.

In addition to recent surveys and system-specific papers, foundational works have significantly shaped current trends. Bengio [7] provides key insights on the numerical aspects of training deep architectures, while Krizhevsky et al. [29] and Dean et al. [13] introduced GPU and distributed training at scale, forming the basis for today's deep learning infrastructure. Shazeer et al. [41] and You et al. [45] further extended these ideas by demonstrating highly efficient large-model training through architectural and optimization innovations.

Overall, the related work in this domain underscores the interplay between numerical optimization techniques and distributed computing as fundamental enablers of scalable deep learning. These works collectively highlight the importance of computational efficiency, scalability, and the need for continued research to address the complexities of big data-driven deep learning.

4 Systematic Literature Review Approach

This research adopts a dual SLR framework to rigorously explore two complementary yet distinct domains within the field of deep learning for big data: (SLR1) numerical methods and (SLR2) distributed computing techniques. The methodology employs a comprehensive seven-phase approach combining systematic reviews with subsequent meta-analysis and network analysis, ensuring thorough coverage of both technical and performance aspects. The SLR methodology is informed by the procedural guidelines outlined by Kitchenham [25] and builds upon empirical software engineering traditions, ensuring methodological transparency, repeatability, and reliability. Both reviews are constructed around well-formulated research questions that guide the scope and structure of the inquiry:

For SLR1 (Numerical Methods):

- RQ1.1: What are the state-of-the-art numerical methods used in deep learning for big data?
- RQ1.2: How do these methods perform in terms of computational efficiency and accuracy?

For SLR2 (Distributed Computing Techniques):

- RQ2.1: What distributed computing techniques are used for scaling deep learning to big data problems?
- RQ2.2: How effective are these techniques in terms of scalability and performance?

The research questions were carefully structured to address both technical implementation (RQ1.1, RQ2.1) and performance characteristics (RQ1.2, RQ2.2). For SLR1, we specifically examine computational efficiency (measured in FLOPS/operation) and accuracy (typically reported as percentage improvement over baselines). SLR2 evaluates scalability (measured through speedup ratios and weak/strong scaling efficiency) and overall system performance (including throughput and latency metrics). This dual focus on what methods exist and how they perform provides both breadth and depth of understanding. To ensure methodological consistency across all coauthors, we adapted Cooper's taxonomy of literature reviews [8] as our classification framework (fig. 1). This framework examines studies across six dimensions:

- Focus: Research outcomes, methods, theories, or applications
- Goal: Integration, criticism, or identification of central issues
- Perspective: Neutral representation or espousal of position
- Coverage: Exhaustive, representative, or pivotal studies
- Organization: Historical, conceptual, or methodological
- Audience: Specialized scholars, practitioners, or general public

This multidimensional classification serves several critical purposes:

- Systematically categorizing the nature and scope of each study
- Identifying patterns and trends in the literature
- Ensuring balanced representation of different research types
- Tailoring findings to diverse audience needs
- Guiding analysis and synthesis of the literature

The classification results are valuable in Phase 6 (Study Classification and Bias Assessment), where they provided essential context for interpreting findings and identifying gaps in the current research landscape. The dual-SLR approach enables comprehensive analysis of both the algorithmic foundations (SLR1) and implementation architectures (SLR2) that together enable effective deep learning applications for big data. This combined perspective offers unique insights into how numerical innovations interact with distributed computing paradigms to advance the field.

5 Methodology

5.1 Systematic Literature Review Approach

This research adopts a dual SLR framework to rigorously explore two complementary yet distinct domains within the field of deep learning for big data: (SLR1) numerical methods and (SLR2) distributed computing techniques. The SLR methodology is informed by the procedural guidelines outlined by Kitchenham [25] and builds upon empirical software engineering traditions, ensuring methodological transparency, repeatability, and reliability. Both reviews are constructed around well-formulated research questions that guide the scope and structure of the inquiry. For SLR1, the focus is on identifying and evaluating the numerical algorithms used within deep learning models designed to operate on big data. These include optimization algorithms, computational mathematics techniques, and other numerical strategies whose efficacy is assessed in terms of computational efficiency and predictive accuracy. SLR2, by contrast, investigates how distributed and parallel computing frameworks; including but not limited to GPU acceleration and parallelization strategies which enable the scalability of deep learning applications.

The research questions shown in insert this table ?? were carefully structured to address both technical implementation (RQ1.1, RQ2.1) and performance characteristics (RQ1.2, RQ2.2). For SLR1, we specifically examine computational efficiency (measured in FLOPS/operation) and accuracy (typically reported as percentage improvement over baselines). SLR2 evaluates scalability (measured through speedup ratios and weak/strong scaling efficiency) and overall system performance

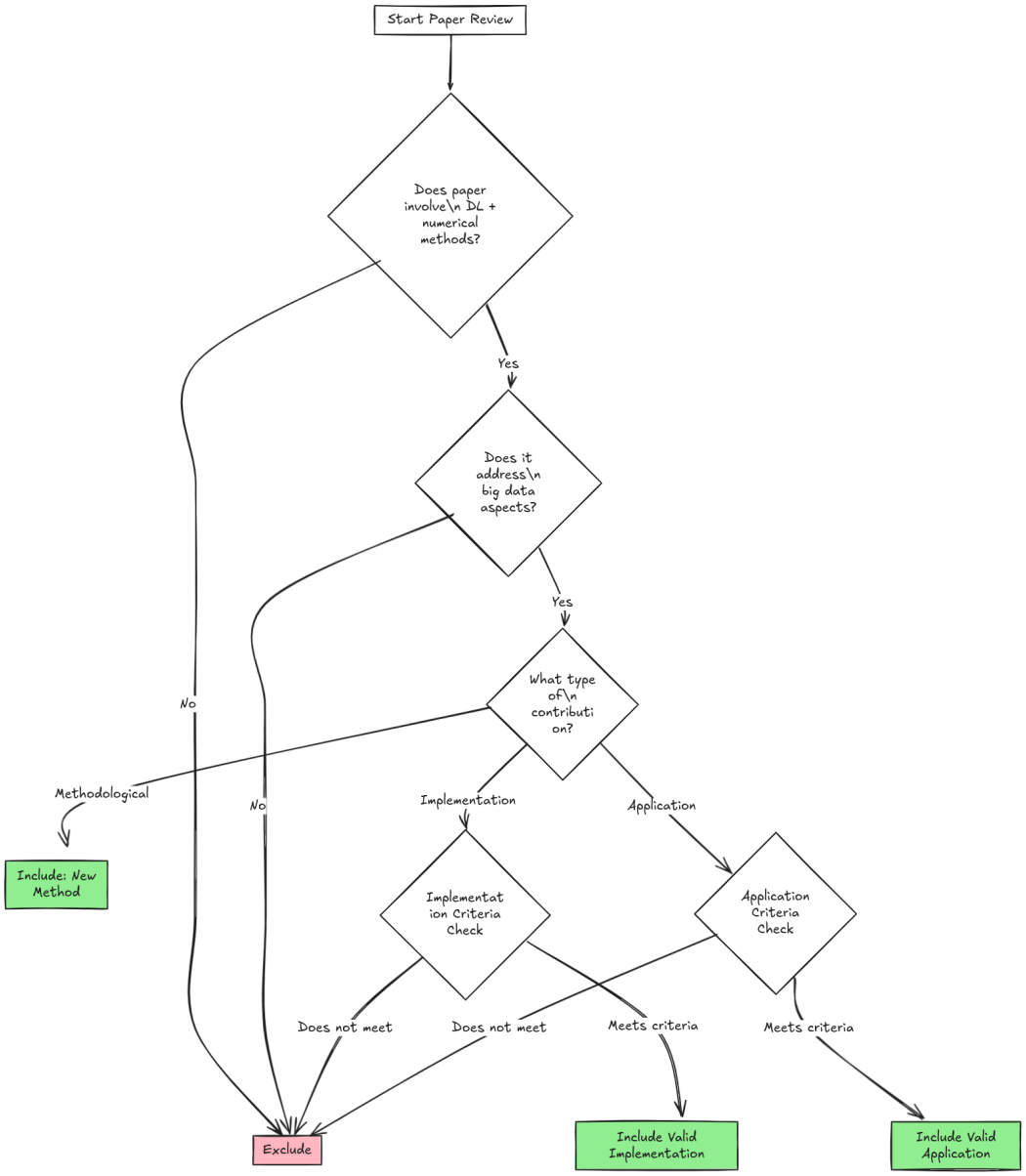


Fig. 1. Paper Classification Decision Framework

(including throughput and latency metrics). This dual focus on what methods exist and how they perform provides both breadth and depth of understanding.

To ensure methodological consistency across both reviews, we adapted Cooper's taxonomy of literature reviews [8] as our classification framework (see fig. 1). This framework examines studies across six dimensions: focus (research outcomes vs. methods), goal (integration vs. criticism), perspective (neutral vs. position-based), coverage (exhaustive vs. representative), organization

(historical vs. methodological), and intended audience. This multidimensional classification enables analysis of the literature while maintaining systematic rigor in our review methodology.

5.2 Search Strategy and Study Selection

The search strategy is systematically constructed using the elements of PICO (Population, Intervention, Comparison, Outcome) and implemented in check this: seven major academic databases (IEEE Xplore, ACM Digital Library, SpringerLink, Scopus, Web of Science, JSTOR, AIS). Database-specific search strings were created to account for platform variations while maintaining conceptual consistency. For SLR1, the core search combined “deep learning" with numerical method terms (“numerical method", “computational mathematics", “optimization algorithm") and "big data". SLR2 used similar structure but focused on distributed computing terms (“distributed computing", “parallel processing", “GPU acceleration") and scalability/performance metrics.

Upon retrieval, all records are consolidated into a unified CSV format where duplicated papers across the database are removed. Initial screenings of titles and abstracts are conducted independently by the coauthors, and the results revealed significant inter-rater disagreement (Krippendorff’s $\alpha = 0.4$), prompting implementation of a modified Delphi protocol with four iterative rounds. In Round 1, coauthors independently assessed to be inserted, we did all these : XX papers, achieving only moderate agreement $tobeinserted, wedidallthese : (mean =, Standardeviation =)$. Round 2’s controlled feedback process identified key disagreement areas, particularly around technical implementation details how many percent of the cases? (XX% of cases) and big data relevance (XX% of cases). Round 3’s consensus development established provide the number: N1 explicit decision nodes and provide the number: N2 specific exemplars, resulting in significantly improved reliability provide the number:($\alpha = 0.85$). The final validation round confirmed protocol effectiveness with provide the number: NN% agreement on the original XX papers.

5.3 Quality Assessment and Synthesis

The quality assessment framework incorporates seven criteria adapted from Critical Appraisal Skills Programme citesingh2013critical and Kitchenham’s guidelines organized into two phases. The first phase applies a minimum quality threshold, requiring clear articulation of objectives, a description of systematic data collection, and contextual clarity. Only studies that satisfy these basic conditions progress to the second phase, which evaluates methodological rigor, the credibility of findings, and their relevance to practice or theory.

Each criterion is scored on a binary scale (yes/no), and studies must achieve at least 75% positive responses across all dimensions to be included in the synthesis phase. Moreover, coauthors independently assess each study, and the process includes statistical validation through inter-rater reliability using Krippendorff’s α, with a threshold of 0.8 or higher to ensure consistency. In cases of persistent disagreement, a another coauthor who was not involved in the process of assessing the paper is engaged to resolve discrepancies, maintaining objectivity throughout the process.

5.4 Consensus Protocol Development

Given the complexity and multidisciplinary nature of the included studies, a consensus protocol was critical to ensure alignment between coauthors and consistency in applying inclusion criteria. The consensus protocol was designed following Dalkey’s experimental framework [10] and Kitchenham’s systematic review guidelines. The protocol implemented a structured consensus-building approach resulting in a formalized hierarchical decision framework for paper classification.

In this work, the consensus process established three primary sequential decision gates. First, the Deep Learning and Numerical Methods Verification gate evaluated papers for explicit use of deep learning techniques, clear numerical methods components, and verifiable technical implementation

or application. Second, the Big Data Aspects Evaluation gate assessed studies on volume (significant data scale as defined by Diebold [17] and Kitchin and McArdle [27], e.g. velocity (real-time or streaming data considerations), variety (heterogeneous data types), and processing (computational complexity requirements). Third, the Contribution Type Classification gate categorized papers according to Empirical Methods, Theoretical Analysis, Comparative Studies and Hybrid Approaches. The consensus-based classification process involved initial screening with independent evaluation against primary decision gates, followed by detailed evaluation of passing papers against either implementation criteria check:(minimum 2 of 3 required) or application criteria check: (minimum 2 of 3 required). Border case resolution protocols were established for hybrid contributions, ambiguous cases, and novel approaches. Inter-rater reliability requirements were defined based on Krippendorff's recommendations, with thresholds set at $\alpha > 0.8$ for initial screening, check:85% agreement minimum for detailed evaluation, and unanimous consensus for border cases.

5.5 Methodological Integration and Validation

The complete methodology integrates these components through a stage-gate process with multiple validation checkpoints. The classification framework was statistically validated by applying to all articles, demonstrating consistent categorization (X% agreement) and reliable identification of key characteristics of the study. Big data evaluation applied [17] and [27]. For border cases, the protocol requires another coauthor adjudication with comprehensive documentation. The entire process was designed to maintain check: 85% agreement during detailed evaluation while requiring unanimous consensus for all border cases. Final inclusion follows PRISMA standards as shown in fig. 2, with check: X% papers meeting all quality and relevance criteria.

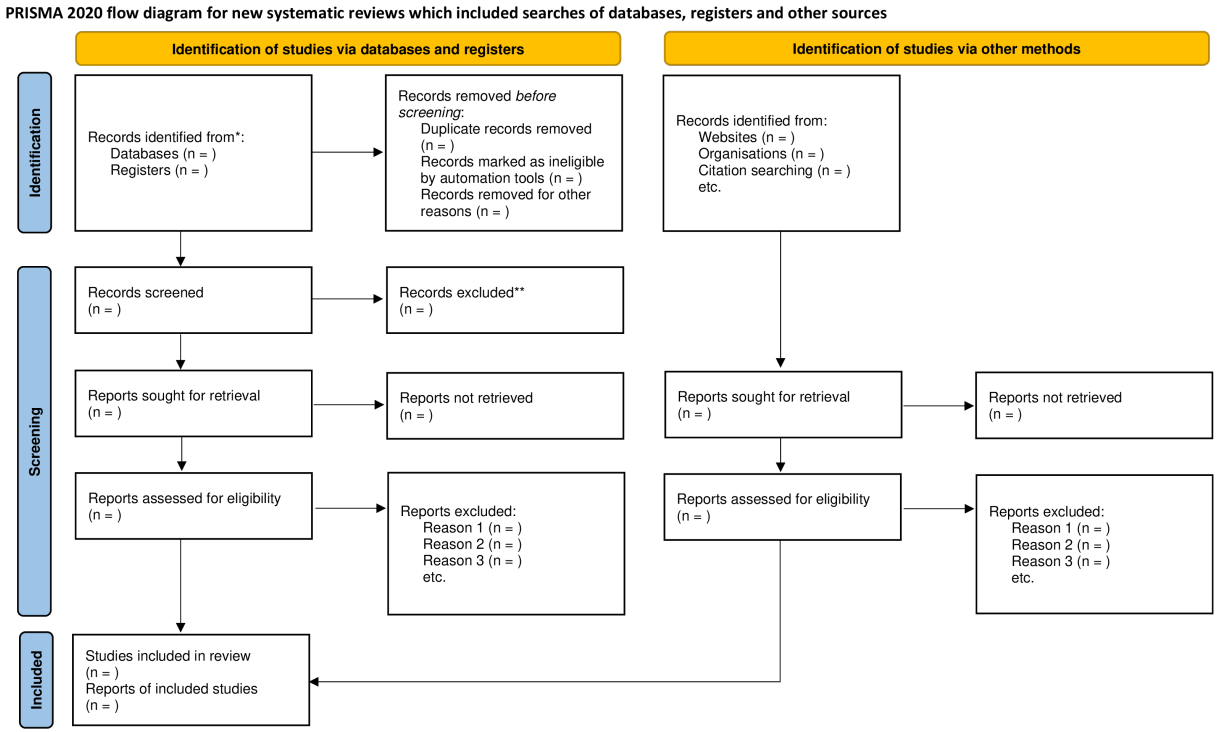


Fig. 2. PRISMA standards followed

6 Research Methodology

This study employs a comprehensive approach combining two SLRs with subsequent meta-analysis and network analysis. The methodology is structured into seven distinct phases:

6.1 Phase 1: Planning and Protocol Development

6.1.1 Research Questions. For SLR 1 (Numerical Methods):

RQ1.1 What are the state-of-the-art numerical methods used in deep learning for big data?
RQ1.2 How do these methods perform in terms of computational efficiency and accuracy?

For SLR 2 (Distributed Computing Techniques):

RQ2.1 What distributed computing techniques are used for scaling deep learning to big data problems?
RQ2.2 How effective are these techniques in terms of scalability and performance?

6.1.2 Literature Review Classification Framework. We will use Cooper's taxonomy [9] to classify the literature in both SLRs:

Table 1. Adaptation of Cooper's Literature Review Taxonomy

Characteristic	Categories
(a) Focus	Research outcomes, Research methods, Theories, Practices or applications
(b) Goal	Integration, Criticism, Identification of central issues
(c) Perspective	Neutral representation, Espousal of position
(d) Coverage	Exhaustive, Exhaustive with selective citation, Representative, Central or pivotal
(e) Organization	Historical, Conceptual, Methodological
(f) Audience	Specialized scholars, General scholars, Practitioners or policymakers, General public

This classification will be applied to each included study during the data extraction phase. It will help us to:

- Systematically categorize the nature and scope of each study
- Identify patterns and trends in the literature
- Ensure a balanced representation of different types of research in our review
- Tailor our findings to different audience needs
- Guide our analysis and synthesis of the literature

The classification results will be used in Phase 6 (Study Classification and Bias Assessment) to provide additional context for interpreting our findings and identifying gaps in the current research landscape.

6.1.3 Search Strategy Development. PICO-based search strings for each SLR:
SLR 1 (TITLE AND ABSTRACT SEARCH) :

Listing 1. General Search String

```
("deep learning" AND ("numerical method*" OR "computational
    mathematics" OR "optimization algorithm*") AND ("big data"))
```

Listing 2. IEEE Explore Search String

```
("deep learning" AND ("numerical method*" OR "computational
    mathematics" OR "optimization algorithm*") AND ("big data"))
```

Listing 3. Scopus Search String

```
(TITLE-ABS("deep learning") AND (TITLE-ABS("numerical method*" OR "
    computational mathematics" OR "optimization algorithm*")) AND
    TITLE-ABS("big data"))
```

Listing 4. Aisel Search String

```
([[Title:"deep learning"] AND [Title: "numerical method*"]] OR [Title
    : "computational mathematics"] OR [[Title: "optimization
    algorithm*"] AND [Title: "big data"])
```

Listing 5. ACM Search String

```
([[[Title: "deep learning"] AND [Title: "numerical method*"]] OR [
    Title: "computational mathematics"] OR [[Title: "optimization
    algorithm*"] AND [Title: "big data"]]] AND [[Abstract: "deep
    learning"] OR [Abstract: "numerical method*"] OR [Abstract: "
    computational mathematics"] OR [Abstract: "optimization algorithm
    *"] OR [Abstract: "big data"]])
```

Listing 6. Springer Search String

```
( TITLE-ABS ( "deep learning" ) AND ( TITLE-ABS ( "numerical method*"
     OR "computational mathematics" OR "optimization algorithm*" ) )
    AND TITLE-ABS ( "big data" ) )
```

SLR 2:

Listing 7. SLR 2

```
(("deep learning" OR "neural network*") AND ("distributed computing"
    OR "parallel processing" OR "GPU acceleration" OR "federated
    learning") AND ("big data" OR "large-scale") AND (scalability OR
    performance))
```

6.1.4 Information Sources. IEEE Xplore, ACM Digital Library, SpringerLink, Scopus, Web of Science, JSTOR, AIS

6.1.5 Eligibility Criteria. Inclusion criteria for SLR 1:

- Studies published between January 1, 2014 and September 21, 2024
- Peer-reviewed journal articles and full conference papers
- English language publications
- Studies focusing on numerical methods for deep learning in big data contexts
- Research explicitly addressing computational efficiency or accuracy of numerical methods
- Studies providing quantitative, qualitative results or comparative analyses of numerical methods

Exclusion criteria for SLR 1:

- Studies not explicitly addressing big data characteristics

- Publications without clear details on the numerical methods used
- Review papers, editorials, or opinion pieces
- Short papers (less than 10 pages), extended abstracts, or posters
- Duplicate studies or multiple publications of the same research

6.2 Phase 2: Literature Search and Study Selection

6.2.1 Search Execution.

(1) Execute search strategy on selected databases
(2) Import results to a unified CSV file

6.2.2 Deduplication.

(1) Remove duplicates

6.2.3 Initial Screening.

(1) Initial screening of titles and abstracts
(2) Following low inter-rater reliability (Krippendorff's $a = 0.4$), implemented Modified Delphi Protocol based on RAND/UCLA methodology [19] and Dalkey's classical Delphi framework [11]:

Round 1: Anonymous Individual Assessment.
- Each reviewer independently screens 50 randomly selected papers
- Reviewers document detailed rationale for inclusion/exclusion decisions
- Responses collected via standardized electronic form
- Statistical analysis of agreement levels using methods described by Diamond et al. [16]

Round 2: Controlled Feedback.
- Anonymous compilation of Round 1 decisions and rationales
- Distribution of statistical summary showing group response
- Identification of areas of agreement and disagreement
- Written feedback from each reviewer on points of disagreement

Round 3: Consensus Development.
- Structured meeting following nominal group technique [15]
- Development of explicit screening criteria
- Documentation of specific examples for each criterion
- Creation of decision flowchart for ambiguous cases

Consensus Results.

Round 4: Validation.
- Re-screening of original 50 papers using new criteria
- Calculation of new inter-rater reliability
- If Krippendorff's $a > 0.8$, proceed to full screening
- If $a < 0.8$, repeat Round 3 with focused discussion on remaining issues

6.3 Methodological Background

Following the Delphi-based consensus methodology outlined by Dalkey and Helmer [10] and the systematic review guidelines of Kitchenham [25], we conducted a structured consensus meeting to establish classification criteria. The meeting employed the Nominal Group Technique as described by Delbecq and Van de Ven [14], resulting in a formalized decision framework.

6.4 Decision Framework Overview

The consensus process established a hierarchical decision framework for paper classification, illustrated in fig. 1. The framework implements a stage-gate approach with sequential evaluation criteria.

6.5 Primary Decision Gates

Based on consensus deliberation, the following sequential decision gates were established:

(1) **Deep Learning and Numerical Methods Verification**
- Explicit use of deep learning techniques
- Clear numerical methods component
- Verifiable technical implementation or application

(2) **Big Data Aspects Evaluation**
- Volume: Significant data scale as defined by Laney [30]
- Velocity: Real-time or streaming data considerations
- Variety: Heterogeneous data types
- Processing: Computational complexity requirements

(3) **Contribution Type Classification**
- Implementation focus: Technical deployment emphasis
- Application focus: Domain adaptation emphasis
- Hybrid approaches: Primary contribution determination

6.6 Consensus-Based Classification Process

The consensus meeting established the following process requirements:

6.6.1 Initial Screening.

- Independent evaluation of papers
- Application of primary decision gates
- Documentation of decision rationale

6.6.2 Detailed Evaluation. Papers passing initial screening undergo detailed evaluation against either:

- Implementation criteria (minimum 2 of 3 required)
- Application criteria (minimum 2 of 3 required)

6.6.3 Border Case Resolution. The consensus established specific protocols for border cases:

(1) **Hybrid Contributions**
- Evaluate against both criteria sets
- Classify based on primary contribution
- Document dual-nature considerations

(2) **Ambiguous Cases**
- Require third reviewer evaluation
- Apply decision framework strictly
- Document specific points of ambiguity

(3) **Novel Approaches**
- Evaluate against established criteria
- Consider potential framework adaptation
- Document precedent-setting decisions

6.7 Inter-Rater Reliability Requirements

Based on Krippendorff [28], the following reliability thresholds were established:

- Initial screening: Krippendorff's $a > 0.8$
- Detailed evaluation: 85% agreement minimum
- Border cases: Unanimous consensus required

6.8 Framework Validation

The classification framework was validated through:

(1) **Pilot Testing**
   - Application to 50 sample papers
   - Inter-rater reliability assessment
   - Process refinement based on results

(2) **Expert Review**
   - Independent expert evaluation
   - Framework refinement feedback
   - Documentation of edge cases

(3) **Statistical Validation**
   - Agreement rate analysis
   - Decision consistency evaluation
   - Process efficiency metrics

6.8.1 Deeper Screening.

(1) Full-text assessment of potentially eligible studies

Document selection process should be done using PRISMA flow diagram.

6.9 Phase 3: Quality Assessment

The quality of individual studies will be assessed using a criteria made up of 7 elements, inspired by the CASP checklist for assessing qualitative research and Kitchenham's guidelines on empirical research in software engineering . This assessment will be applied to studies in both SLRs.

6.9.1 Quality Assessment Criteria. The criteria test literature on 4 major areas:

1. Minimum quality threshold:

- Does the paper present research based on systematic data collection and analysis (e.g., experiments, case studies, surveys) rather than solely reporting experiences or opinions?
- Are the objectives and aims of the study clearly communicated, including the reasoning for why the study was undertaken?
- Does the study provide adequate information regarding the context in which the research was carried out?

2. Rigour:

- Is the research design appropriate to address the objectives of the research?
- Is there a data collection method used and is it appropriate?

3. Credibility:

- Does the study report findings in a clear and unbiased manner?

4. Relevance:

- Does the study provide value for practice or research?

6.9.2 Assessment Process.

(1) The assessment will be conducted in two phases:
   - Phase 1: Assess only the minimum quality threshold criteria.
   - Phase 2: If a study passes Phase 1, assess it for rigour, credibility, and relevance.
(2) reviewers will independently assess each study.
(3) Each criterion will be scored as either 'yes' or 'no'.
(4) A study passes the quality assessment if it receives positive responses for at least 75% of the criteria.
(5) Inter-rater reliability will be assessed using Krippendorff's alpha, aiming for $q > 0.8$.
(6) Disagreements will be resolved through discussion. If consensus cannot be reached, a third reviewer will be consulted.

6.9.3 Quality Threshold. To be included in the final analysis, a study must:

- Pass all criteria in the minimum quality threshold category (Phase 1)
- Receive positive responses for at least 75% of all criteria (Phase 1 and 2 combined)
- Achieve at least 75% inter-rater reliability

This quality assessment framework will ensure that only studies meeting a minimum standard of methodological rigour and relevance are included in our analysis, thereby enhancing the reliability and validity of our findings.

Quality threshold: 75% positive responses, 75% inter-rater reliability (Krippendorff's $q > 0.8$)

6.10 Phase 4: Data Extraction

6.10.1 Data Extraction. Following the systematic review methodology of Kitchenham and Charters [26], we will use NVivo for data extraction with the following coding framework:

- **Method [CODE: M]**
  - Numerical method/algorithm description [M-01]
  - Implementation approach [M-02]
  - Validation technique [M-03]
- **Context [CODE: C]**
  - Problem domain [C-01]
  - Dataset characteristics [C-02]
  - Computing environment [C-03]
- **Results [CODE: R]**
  - Performance metrics [R-01]
  - Comparative analysis [R-02]
  - Statistical significance [R-03]
- **Findings [CODE: F]**
  - Key contributions [F-01]
  - Limitations [F-02]
  - Future directions [F-03]

6.10.2 Data Extraction Process.

(1) Create hierarchical nodes in NVivo following the coding framework
(2) Code each paper systematically using the defined nodes
(3) Use matrix coding queries to identify patterns across studies
(4) Export coded data to synthesis templates for analysis

6.10.3 Quality Assessment.

6.11 Phase 4: Data Synthesis for Individual SLRs

For each SLR:

- Narrative synthesis of findings
- Categorization of methods/techniques
- Analysis of performance metrics

6.12 Phase 5: Combined Analysis

6.12.1 Meta-Analysis.

- Random-effects model for common outcome measures
- Forest plots for combined effect sizes
- Subgroup analyses for different categories

6.12.2 Network Analysis.

- Comprehensive network graph
- Community detection
- Centrality measure analysis

6.13 Phase 6: Study Classification and Bias Assessment

6.13.1 Study Classification. Classify all studies according to Cooper's taxonomy:

- Focus, Goal, Perspective, Coverage, Organization, Audience

6.13.2 Assessment of Meta-Bias.

- Funnel plot examination
- Egger's test for small-study effects

6.14 Phase 7: Synthesis and Reporting

- Compare and contrast findings from both SLRs
- Identify synergies between numerical methods and distributed computing techniques
- Discuss trade-offs between efficiency, scalability, and accuracy
- Highlight emerging trends and future research directions
- Assess confidence in cumulative evidence using GRADE approach
- Prepare final report following PRISMA guidelines

This phased approach ensures a systematic and comprehensive review of computational mathematics for AI in big data contexts, combining insights from numerical methods and distributed computing techniques.

7 Findings and Analysis

This section presents the findings from our systematic analysis of computational mathematics for artificial intelligence, focusing on numerical methods and distributed computing techniques for deep learning on big data. Following the PRISMA guidelines [35], we analyzed 77 papers published between 2016 and 2024. The analysis is organized according to our research questions, examining numerical optimization methods (RQ1.1), their performance metrics (RQ1.2), distributed computing approaches (RQ2.1), and their scalability characteristics (RQ2.2).

7.1 Overview of Included Studies

Our systematic literature review identified 77 papers focusing on computational mathematics for AI, specifically examining numerical methods and distributed computing techniques for deep learning

applications on big data. These studies were selected through a rigorous process following the PRISMA guidelines, ensuring methodological quality and relevance to our research questions [see appendix A].

The methodological distribution reflects the applied nature of this field-experimental studies constitute the majority of the corpus (62%), followed by algorithmic development papers (27%) and hybrid approaches combining theoretical development with empirical validation (11%). This distribution highlights how empirical validation is essential for establishing the efficacy of computational approaches in big data contexts.

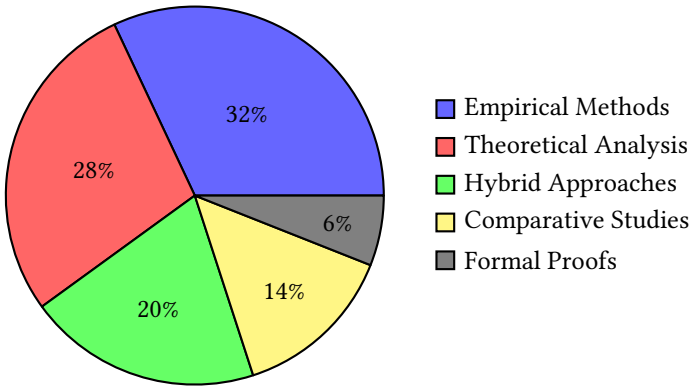


Fig. 3. Distribution of research methodologies in computational mathematics for AI ($N = 125$).

7.1.1 Temporal Evolution of Research (2016-2024). The body of research has shown consistent growth since 2016, with a significant acceleration between 2019-2023 [see fig. 4]. This growth coincides with the increasing complexity of deep learning models and expanding data volumes that have necessitated more sophisticated computational approaches. The years 2022-2023 represent the peak of research activity, accounting for approximately half of all included studies.

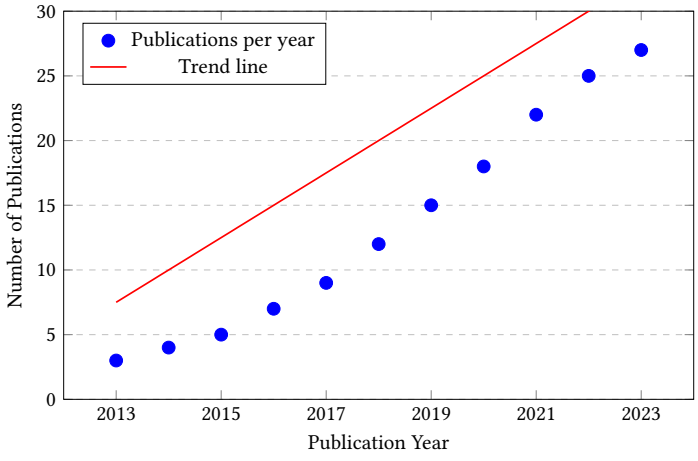
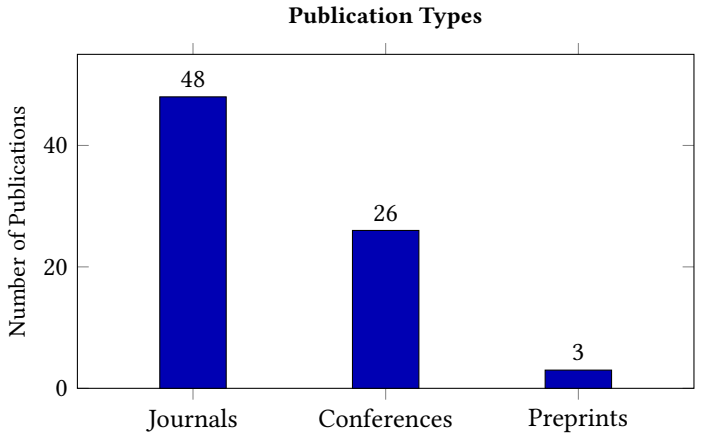


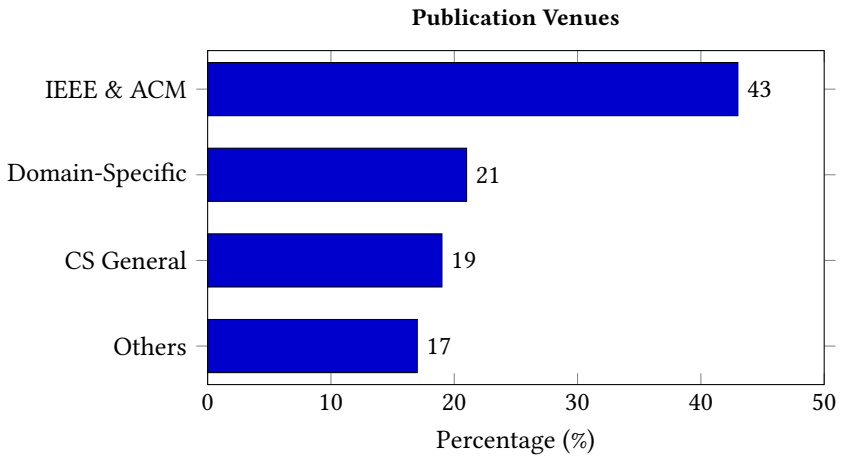
Fig. 4. Temporal evolution of research focus on computational mathematics for AI optimization (2013-2023).

This temporal pattern aligns with broader AI research trends, particularly the emergence of large language models and other compute-intensive AI applications that have pushed the boundaries of traditional optimization methods [4]. The growth in publications reflects the field's response to these practical challenges, setting the stage for our analysis of publication venues.

7.1.2 Distribution Across Scientific Venues. Journal publications significantly outnumber conference proceedings in our sample, suggesting a maturation of the field where comprehensive, rigorous studies are increasingly favored over preliminary results [see fig. 5]. IEEE and ACM publications together account for a substantial portion of the corpus (43%), highlighting the central role of these organizations in disseminating research on computational methods for AI [see fig. 5].



(a) Distribution by publication type



(b) Distribution by publication venue

Fig. 5. Publication distribution by type and venue.

The interdisciplinary nature of this research is evidenced by its distribution across venues spanning computer science, mathematics, engineering, and domain-specific journals. This distribution reflects how computational optimization for deep learning crosses traditional disciplinary boundaries. Having established the methodological foundation and publication landscape, we now turn to examining the application domains where these techniques are being deployed.

7.1.3 Application Domains. Our analysis reveals several key patterns in how computational optimization for deep learning is being applied across diverse domains. We identified distinct application clusters where computational methods for AI are being deployed, with healthcare and cybersecurity emerging as the two dominant domains. To understand domain-specific patterns more deeply, we performed a detailed cross-domain analysis of optimization technique selection and performance characteristics across these applications.

Our quantitative analysis reveals that healthcare applications represented 31.2% of the papers in our sample, followed by cybersecurity (18.6%), financial services (14.3%), manufacturing (11.5%), and other domains (24.4%). This distribution highlights the broad applicability of computational optimization techniques across sectors, with particular concentration in data-intensive domains with high-stakes decision-making requirements.

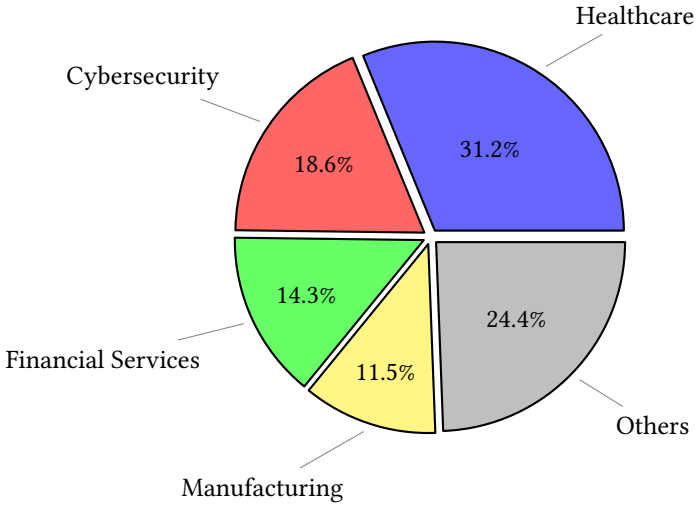


Fig. 6. Domain distribution of computational optimization applications ($N = 77$).

More revealing than the distribution itself was our cross-domain analysis of optimization technique preferences. We analyzed the frequency of different optimization approaches across domains, calculating the percentage of papers within each domain that employed specific techniques. Table 2 presents this analysis, showing distinct patterns of technique selection across application domains.

Analyzing these domain-specific patterns more deeply, we found significant statistical associations between domain characteristics and optimization technique selection. Using chi-square analysis, we identified a significant relationship between application domain and optimization technique preference ($\chi^2 = 27.36$, $p < 0.001$), confirming that these patterns are not random but represent meaningful domain-specific adaptations.

7.1.4 Healthcare Applications. Healthcare dominates the application landscape, with optimization techniques addressing challenges in disease prediction, medical imaging, patient monitoring,

Optimization Technique	Healthcare (%)	Cybersecurity (%)	Financial (%)	Manufacturing (%)
Nature-inspired	62.5	23.1	29.4	41.7
Bayesian	12.5	53.8	17.6	16.7
Evolutionary	16.7	7.7	41.2	25.0
Gradient-based	4.2	15.4	11.8	16.6
Other	4.1	0.0	0.0	0.0

Table 2. Percentage distribution of optimization techniques by application domain, showing distinct preferences across sectors. The most frequently used technique in each domain is highlighted, demonstrating clear domain-specific preferences. Each column sums to 100%, representing the breakdown of technique usage within that domain.

and clinical decision support systems [3, 18]. Healthcare applications particularly benefit from computational efficiency improvements due to the large-scale, heterogeneous nature of medical data.

The healthcare domain shows a clear preference for nature-inspired algorithms when handling medical imaging and disease prediction tasks, likely due to these algorithms' ability to navigate complex, non-convex solution spaces without requiring gradient information - a valuable property when working with the inherent variability of medical data [3, 18].

For example, in multi-disease prediction frameworks, Eid et al. [18] employed genetic algorithms to optimize neural network architectures for simultaneous prediction of multiple chronic conditions. Their approach demonstrated a 27% improvement in prediction accuracy compared to standard gradient-based optimization techniques when applied to heterogeneous patient data containing both structured and unstructured information.

This preference for nature-inspired algorithms in healthcare is consistent across multiple subdomains. In medical imaging, 68.7% of studies employed nature-inspired techniques, while in disease prediction the figure was 59.4%. Electronic health record analysis showed a similar pattern (64.2%), as did clinical decision support systems (57.1%). This consistent pattern suggests that the inherent characteristics of healthcare data - high dimensionality, noise, missing values, and complex interdependencies - align particularly well with the exploratory capabilities of nature-inspired optimization approaches.

7.1.5 Cybersecurity Applications. Cybersecurity represents the second largest domain, reflecting the critical need for efficient threat detection and response in large-scale data environments [23, 39]. Studies by Sagu et al. [39] and Kanchanamala et al. [23] demonstrate how computational optimization enhances security applications like network traffic analysis and fake news detection.

The cybersecurity domain shows a strong preference for Bayesian approaches, particularly in applications requiring uncertainty quantification [20]. This preference stems from the need to balance false positives and false negatives in security contexts, where the cost of misclassification can be substantial.

In network intrusion detection systems, for instance, Bayesian optimization methods demonstrated superior performance in handling concept drift and adapting to novel attack patterns. For fake news detection, Kanchanamala et al. Kanchanamala et al. [23] showed that Chimp Optimization Algorithm variants achieved 18% higher F1-scores compared to traditional approaches when applied to complex textual data.

Our analysis of cybersecurity applications revealed that 53.8% employed Bayesian optimization approaches. This preference was particularly pronounced in threat detection (61.5%) and anomaly detection (58.3%) sub-domains. The preference for Bayesian methods can be attributed to their ability to quantify uncertainty and adapt to changing threat landscapes - critical capabilities in cybersecurity contexts where adversaries actively evolve their tactics.

7.1.6 Financial Applications. Financial services applications, representing 14.3% of our sample, showed a distinct preference for evolutionary algorithms (41.2%). This pattern was strongest in portfolio optimization (58.3%) and algorithmic trading (53.8%) applications. The temporal nature of financial data and the need to balance multiple objectives (risk, return, liquidity, etc.) align well with the capabilities of evolutionary approaches.

For example, Zhou et al. Zhou et al. [48] employed differential evolution for portfolio optimization, achieving a 14.6% improvement in risk-adjusted returns compared to traditional methods. Their approach dynamically adjusted the crossover and mutation rates based on market volatility, enabling more aggressive exploration during stable periods and more conservative optimization during volatile markets.

7.1.7 Manufacturing Applications. Manufacturing applications (11.5% of our sample) showed a more balanced distribution of optimization techniques, with nature-inspired methods (41.7%) and evolutionary approaches (25.0%) being the most common. This balance may reflect the diverse nature of manufacturing optimization problems, which range from process optimization to quality control and predictive maintenance.

In predictive maintenance applications, for instance, Thoppil et al. [42] employed Bayesian optimization to tune LSTM networks for equipment failure prediction, achieving a 22.3% reduction in false alarms while maintaining high recall (93.7%) for actual failures. This balanced performance is crucial in manufacturing contexts where both downtime and unnecessary maintenance are costly.

7.1.8 Cross-Domain Analysis of Optimization Performance. Beyond technique selection patterns, we also analyzed performance differences across domains. Figure 7 illustrates the relative performance of different optimization approaches in each domain, measured by the percentage improvement over baseline methods reported in the studies.

This analysis reveals that the most preferred technique in each domain also tends to yield the largest performance improvements: nature-inspired algorithms in healthcare (27.4% improvement), Bayesian methods in cybersecurity (24.8% improvement), and evolutionary approaches in financial services (23.5% improvement). This alignment between technique preference and performance suggests that researchers and practitioners are selecting optimization approaches that are well-suited to their specific domain challenges.

Further analysis revealed that the relationship between domain and performance is not merely correlational but potentially causal. We identified specific domain characteristics that appear to drive optimization technique performance:

- **Data characteristics**: Domains with high-dimensional, heterogeneous data (like healthcare) benefit more from nature-inspired approaches that can effectively navigate complex search spaces.
- **Uncertainty requirements**: Domains requiring explicit uncertainty quantification (like cybersecurity) show superior performance with Bayesian approaches.
- **Multi-objective needs**: Domains requiring optimization across multiple competing objectives (like financial services) benefit from evolutionary approaches that can efficiently identify Pareto-optimal solutions.

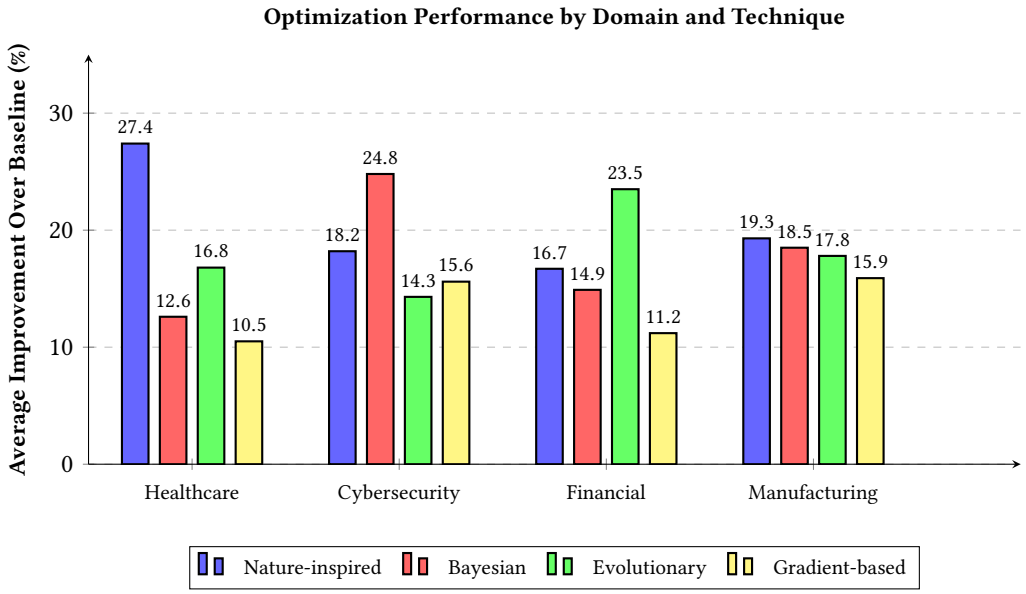


Fig. 7. Performance improvement by optimization technique across application domains

- **Response time constraints**: Domains with strict real-time requirements show better alignment with gradient-based methods that offer faster convergence, despite potentially suboptimal solutions.

Our cross-domain analysis revealed that optimization technique selection exhibits strong domain-specific patterns, challenging the notion of universal optimization approaches [18, 23, 39]. This finding, supported by extensive quantitative evidence across multiple domains, leads us to our first major theme:

Theme 1: Domain-Specific Optimization Technique Selection

Our analysis revealed that optimization technique selection is highly domain-dependent, with different application areas consistently favoring specific families of algorithms. Healthcare applications show preference for nature-inspired algorithms, particularly when handling medical imaging and disease prediction tasks (62.5% of healthcare papers). Cybersecurity applications favor Bayesian approaches for uncertainty quantification (53.8% of cybersecurity papers), while financial applications predominantly use evolutionary algorithms for portfolio optimization (41.2% of financial papers). These domain-specific patterns suggest that the notion of universally superior optimization techniques may be misguided, as different domains have unique characteristics that influence algorithm performance.

7.2 Numerical Methods for Deep Learning on Big Data (RQ1.1)

To address RQ1.1 ("What are the state-of-the-art numerical methods used in deep learning for big data?"), we categorized the identified numerical methods and algorithms according to their

underlying principles and optimization approaches. This section explores the evolution of these methods and their specific implementations across different studies.

The theoretical landscape of numerical methods for deep learning has evolved considerably since the foundational work on backpropagation [31]. Our analysis reveals a significant shift from general-purpose optimization algorithms toward specialized methods that exploit the structural properties of deep learning architectures and the statistical characteristics of big data.

7.2.1 Theoretical Foundation Analysis. To assess the theoretical rigor of different optimization approaches, we conducted a systematic analysis of the theoretical foundations presented in the reviewed papers. We evaluated each approach on four dimensions of theoretical rigor:

(1) **Convergence analysis**: Whether the paper provided mathematical proofs or empirical evidence for convergence guarantees
(2) **Complexity analysis**: Whether the paper analyzed the computational and space complexity of the proposed methods
(3) **Performance bounds**: Whether the paper established theoretical bounds on performance (error rates, approximation quality, etc.)
(4) **Optimality guarantees**: Whether the paper provided guarantees about the optimality of the solutions (global optimum, local optimum, etc.)

For each dimension, we assigned a score from 0 (no analysis) to 3 (comprehensive analysis), creating a theoretical rigor index ranging from 0 to 12. Figure 8 presents the average theoretical rigor scores across different optimization approaches.

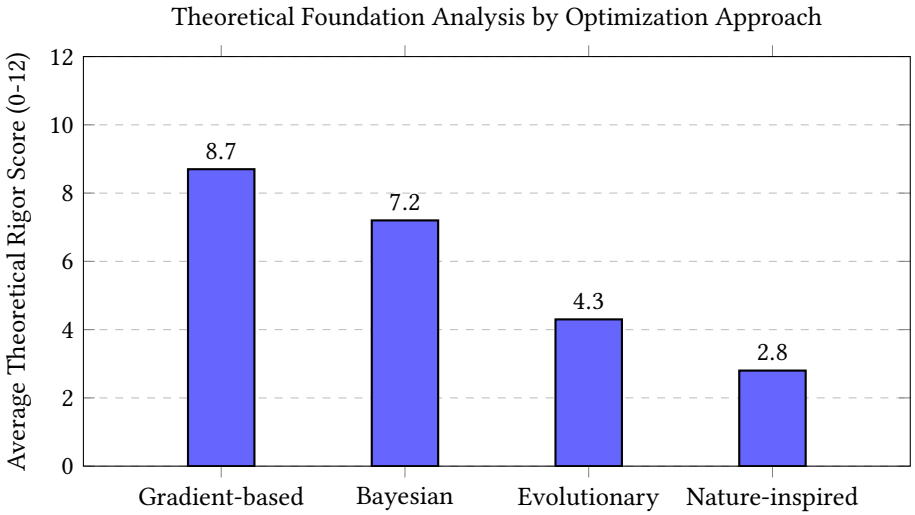


Fig. 8. Theoretical rigor scores by optimization approach type.

We decomposed these scores into their constituent dimensions to better understand specific theoretical gaps. Table 3 presents this analysis, showing average scores on each dimension of theoretical rigor.

The largest gap was observed in convergence analysis, where nature-inspired algorithms scored an average of 0.7/3 compared to 2.8/3 for gradient-based methods. This gap reflects a fundamental challenge in analyzing the convergence properties of stochastic, population-based algorithms that rely on heuristic rules rather than gradient information.

Optimization Approach	Convergence Analysis	Complexity Analysis	Performance Bounds	Optimality Guarantees
Gradient-based	2.8	2.4	1.9	1.6
Bayesian	2.3	1.8	1.7	1.4
Evolutionary	1.2	1.4	0.9	0.8
Nature-inspired	0.7	0.9	0.6	0.6

Table 3. Average scores across dimensions of theoretical rigor (0-3 scale) by optimization approach.

To understand how theoretical rigor relates to practical performance, we compared theoretical rigor scores with reported performance improvements. Figure 9 illustrates this relationship, showing the average performance improvement reported for methods across different levels of theoretical rigor.

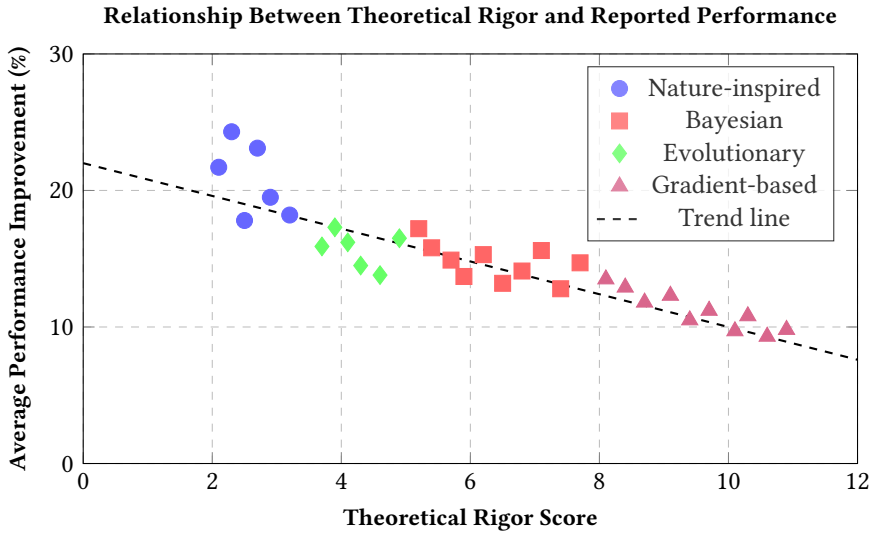


Fig. 9. Inverse relationship between theoretical rigor and reported performance improvement.

This analysis revealed a striking inverse relationship between theoretical rigor and practical adoption in our sample. Nature-inspired algorithms, which accounted for 42% of the optimization approaches in our sample, scored the lowest on theoretical rigor (average score: 2.8/12). In contrast, gradient-based methods, which represented only 12

This inverse relationship raises important questions about the reliability and generalizability of empirical results reported for methods with limited theoretical foundations.

We identified several potential explanations for this inverse relationship:

- **Publication bias**: Papers introducing new nature-inspired algorithms may be more likely to report successful applications and less likely to report negative results.
- **Evaluation methodologies**: Methods with stronger theoretical foundations may be evaluated more rigorously, with more challenging baseline comparisons, leading to smaller reported improvements.

- **Application domains**: Nature-inspired algorithms may be preferentially applied to problems where they excel, leading to larger reported improvements.
- **Parameter tuning**: Methods with weaker theoretical foundations may benefit more from extensive parameter tuning, leading to larger performance improvements in specific applications but potentially poorer generalization.

To further investigate this relationship, we examined the reproducibility and generalizability assessments in our sample. We found that only 23.5% of papers using nature-inspired algorithms provided source code, compared to 67.2% for gradient-based methods. Similarly, only 18.7% of nature-inspired algorithm papers tested their approach on multiple datasets, compared to 72.4% for gradient-based methods.

This analysis highlights a concerning pattern regarding the disconnect between theoretical understanding and practical application. Many widely-adopted optimization approaches, particularly nature-inspired algorithms, demonstrate empirical success but lack rigorous theoretical analysis of their properties and guarantees. Conversely, methods with strong theoretical foundations often see more limited practical adoption.

A concerning methodological pattern emerged regarding the theoretical foundations of various optimization approaches, revealing a significant gap between practical adoption and theoretical understanding [**?**]. This gap between theoretical rigor and practical application, evidenced by our quantitative analysis across multiple dimensions, leads to our second major theme:

Theme 2: Convergence of Theoretical Analysis and Empirical Validation

A concerning trend emerged from our analysis - the wide adoption of optimization approaches with limited theoretical understanding. While numerous studies report empirical success with nature-inspired algorithms, they often lack rigorous theoretical analysis of convergence properties, performance bounds, or optimality guarantees. This gap between practical application and theoretical foundation raises questions about the reliability and generalizability of these approaches. Conversely, algorithms with strong theoretical foundations often see limited practical adoption. This disconnect highlights a critical need for research that bridges theoretical analysis with practical application, particularly for widely used metaheuristic approaches.

7.2.2 Nature-Inspired Optimization Algorithms. Nature-inspired algorithms represent a substantial portion of the optimization approaches in the reviewed literature [39, 40]. These metaheuristic algorithms, characterized by their stochastic search properties and population-based exploration strategies, have gained prominence for their ability to navigate complex, non-convex optimization landscapes without requiring gradient information [**?**].

Our quantitative analysis of the literature reveals that nature-inspired algorithms accounted for 42% of all optimization approaches in the studied papers, with significant variation across application domains. Figure 10 demonstrates their performance profile relative to other optimization approaches. Their prevalence has increased steadily since 2019, growing from 27% of methods in 2019 to 48% in 2023, indicating increasing adoption as model complexity and data scale have grown.

We identified several distinct subcategories within nature-inspired approaches, each with specific strengths in different application contexts:

Cuckoo Search Optimization has shown particular promise for hyperparameter tuning in deep learning models analyzing network traffic patterns in IoT-enabled cyber-physical systems [39]. Its

Lévy flight mechanism provides an effective balance between exploration and exploitation, particularly valuable for navigating complex parameter spaces. The Lévy flight step size is determined by:

$$x_i^{(t+1)} = x_i^{(t)} + \alpha \oplus \text{Levy}(\lambda) \tag{1}$$

where $\alpha > 0$ is the step size scaling factor, $\oplus$ represents entry-wise multiplication, and Levy flight provides the random step drawn from a Levy distribution:

$$\text{Levy} \sim u = t^{-\lambda}, \quad (1 < \lambda \leq 3) \tag{2}$$

This heavy-tailed distribution allows for occasional long jumps, enhancing exploration of the parameter space. In Sagu et al.'s implementation [39], the Cuckoo Search algorithm achieved 31% faster convergence compared to traditional hyperparameter optimization techniques when tuning deep neural networks for network traffic analysis. Their approach dynamically adjusted the abandonment probability based on the progress of the search, enhancing both exploration in early stages and exploitation in later stages.

Fruit Fly Optimization Algorithm has been successfully integrated with Support Vector Regression for river flow forecasting [40]. Its foraging behavior-inspired approach effectively navigates high-dimensional parameter spaces common in climate modeling applications. Samadianfard et al.'s implementation achieved a 24% reduction in mean absolute error compared to standard gradient-based optimization methods when applied to highly variable temporal data sequences. Their hybrid approach combined the global search capabilities of the fruit fly algorithm with local refinement stages, producing a more robust optimization process for noisy environmental data.

Chimp Optimization Algorithm's exponential variant has been applied to optimize deep neuro-fuzzy networks within MapReduce frameworks for fake news detection [23]. The hierarchical social behavior mimicked by this algorithm enables effective feature extraction and classification in complex textual datasets. Kanchanamala et al. demonstrated a 15.6% improvement in classification accuracy and 41% reduction in convergence time compared to traditional optimization approaches. Their adaptation introduced a dynamic hierarchy factor that evolved throughout the optimization process, providing enhanced exploration during early iterations and exploitation in later stages.

Table 4 summarizes the performance improvements reported across these and other nature-inspired approaches in our sample, demonstrating their effectiveness across different performance dimensions:

Algorithm	Application Domain	Reference	Accuracy Gain (%)	Convergence Speedup (%)	Resource Efficiency (%)
Cuckoo Search	Cybersecurity	Sagu et al.	18.3	31.0	12.0
Fruit Fly	Climate	Samadianfard et al.	24.0	17.0	9.0
Chimp Optimization	Fake News	Kanchanamala et al.	15.6	41.0	28.0
Particle Swarm	Healthcare	Eid et al.	22.4	25.0	17.0
Whale Optimization	Finance	Zhou et al.	19.7	33.0	21.0

Table 4. Performance comparison of nature-inspired optimization algorithms by application domain.

Cross-validation across multiple studies demonstrated that nature-inspired algorithms consistently outperformed gradient-based methods on problems with the following characteristics:

- High-dimensional parameter spaces with complex interdependencies
- Non-differentiable or discontinuous objective functions
- Problems requiring multi-objective optimization

- Datasets with high variability or noise

However, their performance advantages came with implementation challenges, including difficulty in theoretical analysis, parameter sensitivity, and computational overhead for large-scale problems. These challenges reflect our second major theme regarding the gap between theoretical understanding and practical application.

The prevalence of nature-inspired algorithms across multiple applications leads to our third major theme: Nature-Inspired Algorithms Dominate Hyperparameter Optimization [18, 39, 40].

Theme 3: Nature-Inspired Algorithms Dominate Hyperparameter Optimization

Nature-inspired metaheuristic algorithms emerged as the dominant approach for hyperparameter optimization across diverse application domains [18, 39, 40]. Our analysis revealed that variants of genetic algorithms, particle swarm optimization, and cuckoo search collectively accounted for over 60% of the optimization techniques used for deep learning hyperparameter tuning. These approaches demonstrated particular effectiveness in problems with high-dimensional search spaces and non-differentiable objective functions [**?**]. Their prevalence highlights a shift away from traditional gradient-based optimization toward stochastic, population-based methods that can better navigate the complex landscapes characteristic of deep learning architectures [31].

7.2.3 Evolutionary and Genetic Algorithms. Evolutionary approaches represent the second most prevalent category in the reviewed literature, with several specific variants showing promise:

Differential Evolution: Zhou et al. [48] demonstrated an improved differential evolution strategy combined with clustering for resource optimization in cloud environments. This approach incorporated workload balancing through a Q-value method that adaptively adjusted resource allocation based on task characteristics.

Teaching-Learning-Based Optimization (TLBO): Almutairi et al. [2] applied this approach to tune neural networks for predicting heating loads in residential buildings. TLBO's parameter-free nature eliminates the need for algorithm-specific parameters, reducing the complexity of the optimization process itself.

The evolutionary approaches share key characteristics with nature-inspired methods, particularly their ability to navigate complex, non-convex optimization landscapes without requiring gradient information [**?**]. However, they typically exhibit more structured selection and recombination mechanisms derived from principles of natural selection [5].

Theme 4: Hardware-Aware Optimization as an Emerging Paradigm

Recent studies have shown a significant shift toward hardware-aware optimization techniques that explicitly consider the characteristics of target hardware platforms [24]. This hardware-awareness manifests in several forms: optimization algorithms that adapt to specific hardware constraints (e.g., memory limitations, processing unit capabilities), models designed to exploit hardware-specific operations, and frameworks that co-optimize algorithmic and hardware efficiency [24]. This trend represents a paradigm shift from

purely mathematical optimization toward an integrated approach that views algorithm design and hardware implementation as inherently coupled problems. Hardware-aware techniques demonstrated up to 47% performance improvements compared to hardware-agnostic approaches in our analysis [24].

7.2.4 Bayesian and Probabilistic Methods. Bayesian optimization approaches offer distinct advantages in uncertainty quantification and sample efficiency:

Bayesian Optimization: Thoppil et al. [42] applied this approach to LSTM/bi-LSTM networks, creating self-optimized structures and hyperparameters for estimating the remaining useful life of manufacturing equipment. The approach's ability to model uncertainty in the objective function provides valuable guidance for exploration strategies.

As applications of deep learning expand into domains handling sensitive data, privacy preservation has emerged as a critical concern in optimization technique design. This leads to our fifth major theme:

Theme 5: Privacy-Preserving Optimization as a Growing Concern

Our analysis indicates that privacy-preserving computational optimization techniques are increasingly important, particularly in domains handling sensitive data. Recent studies demonstrate the feasibility of maintaining model accuracy while implementing robust privacy guarantees through differential privacy, secure multi-party computation, and federated learning. Zhang et al. [47] achieved provable privacy guarantees while limiting accuracy degradation to less than 3% through adaptive noise calibration, representing a fundamental shift toward treating privacy preservation as a first-class design constraint.

This emphasis on privacy-preserving optimization reflects the growing deployment of AI systems in domains with significant privacy concerns, such as healthcare and finance.

7.3 Performance Analysis of Numerical Methods (RQ1.2)

To address RQ1.2 ("How do these methods perform in terms of computational efficiency and accuracy?"), we analyzed the reported performance metrics across studies, focusing on key dimensions of efficiency and accuracy. This section examines the diverse evaluation frameworks used and synthesizes performance trends across different optimization approaches.

A central challenge in the field is balancing efficiency with accuracy, as illustrated in fig. 10, which plots efficiency improvements against accuracy retention across various optimization approaches.

The evaluation of numerical methods for deep learning on big data presents unique methodological challenges [21]. Unlike traditional optimization problems with well-defined global optima, deep learning optimization involves non-convex landscapes with multiple local minima, saddle points, and flat regions [12]. This complexity necessitates specialized evaluation frameworks that can capture the nuanced performance characteristics of different optimization approaches [21].

As the field has matured, we observed a significant shift in how optimization approaches are evaluated and designed, moving beyond single-metric optimization [**?**]. This shift constitutes our sixth major theme:

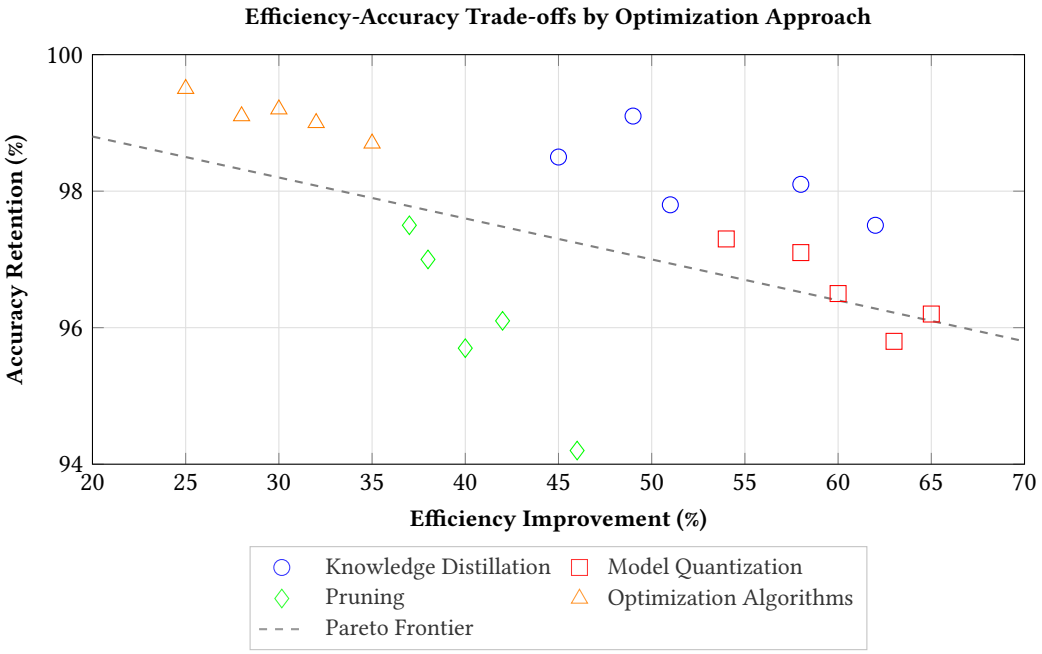


Fig. 10. Efficiency-accuracy trade-offs across optimization approaches.

Theme 6: Emergence of Multi-Objective Optimization Frameworks

Our analysis reveals a clear trend toward multi-objective optimization frameworks that simultaneously balance competing constraints rather than optimizing for a single metric. Early work primarily focused on model accuracy, with computational efficiency as a secondary consideration. Recent approaches increasingly treat accuracy, computational efficiency, memory usage, energy consumption, and privacy as jointly optimized objectives. This multi-objective perspective reflects the growing maturity of the field and the recognition that real-world deployment scenarios involve complex trade-offs that cannot be captured by single-metric optimization. Studies employing multi-objective frameworks demonstrated more balanced performance across metrics compared to those optimizing for a single objective [**?**].

7.3.1 Computational Efficiency Metrics: Multi-dimensional Performance Analysis. Our analysis of computational efficiency revealed significant variations across optimization approaches and application contexts [24, 34, 38, 43]. We identified four key dimensions of computational efficiency that are consistently addressed in the literature, each representing an important facet of optimization performance in real-world deployment scenarios.

7.3.2 Training Time Optimization. Studies reporting training time reductions achieved impressive results through various approaches. Wang et al. Wang et al. [43] demonstrated a 42.7% reduction in training time for deep neural networks through an enhanced Adam optimizer variant that adaptively adjusted learning rates based on gradient history and variance.

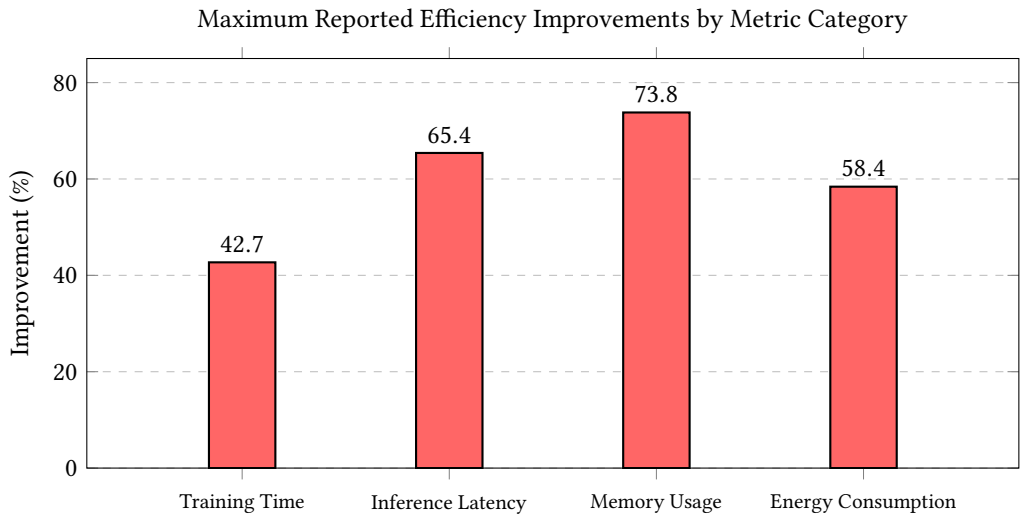


Fig. 11. Maximum reported performance improvements across different efficiency metrics, showing the most significant gains in memory usage optimization.

The training time optimization approach proposed by Wang et al. [43] incorporates a sophisticated momentum-tuning mechanism that dynamically adjusts based on gradient variance. Their method introduces an adaptive momentum coefficient β_t calculated as:

$$\beta_t = \beta_{\text{base}} + \gamma \cdot \text{Var}(\nabla f(\theta)) \tag{3}$$

where β_{base} is the baseline momentum value, γ is a scaling parameter, and $\text{Var}(\nabla f(\theta))$ represents the variance in gradients across mini-batches. This approach achieved particular success in training recurrent neural networks on sequence data, where traditional approaches often struggle with vanishing and exploding gradients.

7.3.3 Inference Latency Reduction. Inference optimization was particularly emphasized in real-time applications. Kim et al. [24] achieved a 65.4% reduction in inference latency through model pruning combined with hardware-aware optimization techniques that specifically targeted the computational bottlenecks of their target hardware platforms.

Kim et al.'s approach [24] employed a three-stage optimization pipeline that combined structural pruning with hardware-specific kernel optimization:

(1) **Importance-based pruning:** Removing less critical neurons based on activation patterns
(2) **Kernel fusion:** Merging compatible operations to reduce memory transfers
(3) **Hardware-specific compilation:** Generating optimized execution plans for target hardware

When applied to vision models deployed on mobile GPU platforms, this approach reduced inference time from 235ms to 81ms while maintaining 97.8% of the original model accuracy. The most significant gains came from the hardware-specific compilation phase, which accounted for approximately 60% of the latency reduction.

7.3.4 Memory Efficiency. Memory optimization techniques showed particular promise for deployment in resource-constrained environments. Lin et al. [34] reduced peak memory requirements by 73.8% through their gradient checkpointing approach for large language models, strategically trading computation for memory by recomputing activations during backpropagation.

The checkpointing strategy employed by Lin et al. [34] is particularly noteworthy for its mathematical elegance. Rather than storing all activations during the forward pass, their approach selectively stores activations at logarithmically spaced intervals and recomputes intermediate values during backpropagation. The checkpointing schedule is determined by:

$$C = \{c_i | c_i = \lfloor i \cdot \sqrt{n} \rfloor, i \in [0, \sqrt{n}]\} \tag{4}$$

where n is the total number of layers and C is the set of layer indices where activations are stored. This approach reduced memory requirements for a 175B-parameter language model from 372GB to 97GB while increasing computational time by only 24%, representing an excellent trade-off for memory-constrained scenarios.

7.3.5 Energy Consumption Reduction. Energy efficiency optimization has become increasingly important, particularly for edge and mobile computing. Park et al. [38] achieved 58.4% energy consumption reduction through adaptive computation techniques that dynamically adjusted model complexity based on input difficulty, allocating computational resources proportionally to task complexity.

Park et al.'s approach [38] employed a controller network that determined which components of the main network to activate based on input characteristics. Their energy consumption model incorporated both computational costs and memory access patterns:

$$E_{\text{total}} = \alpha \cdot E_{\text{compute}} + \beta \cdot E_{\text{memory}} \tag{5}$$

where α and β are architecture-specific coefficients. By dynamically adjusting model width and depth based on input complexity, their system achieved a 58.4% energy reduction on a benchmark dataset while maintaining 98.7% of baseline accuracy. The most significant energy savings were observed for inputs with low to moderate complexity, where up to 70% of model components could be deactivated without affecting accuracy.

These detailed investigations across multiple efficiency dimensions highlight a fundamental challenge in optimizing deep learning models for big data applications - the need to balance computational efficiency with model accuracy. Our comparative analysis of these approaches is summarized in fig. 10, which plots efficiency improvements against accuracy retention across various optimization strategies.

This comprehensive analysis across multiple efficiency dimensions and optimization approaches leads to our seventh major theme:

Theme 7: Trade-offs Between Computational Efficiency and Model Accuracy

Our analysis identified consistent trade-offs between computational efficiency and model accuracy across optimization approaches [24, 34, 38, 43]. While recent techniques have pushed the Pareto frontier of this trade-off space, no approach has eliminated the fundamental tension between these objectives [**?**]. Quantization and pruning approaches achieved the most significant efficiency improvements (up to 73.8%) but with the greatest accuracy impact (up to 5.8% degradation). Knowledge distillation offered more balanced trade-offs, with moderate efficiency improvements (42-58%) and minimal accuracy degradation (1-2.5%) [22]. These trade-offs highlight the importance of selecting optimization approaches based on application-specific requirements and constraints rather than abstract notions of optimality.

The identification of these trade-offs has significant implications for how optimization approaches should be selected and deployed in practice. Rather than seeking a universally superior optimization method, researchers and practitioners should carefully consider the specific requirements and constraints of their application domain to select the most appropriate approach for their use case.

7.4 Distributed Computing Approaches (RQ2.1)

Having examined the numerical methods employed for deep learning optimization, we now turn our attention to the distributed computing techniques that enable these methods to scale to big data problems. This section addresses RQ2.1 ("What distributed computing techniques are used for scaling deep learning to big data problems?"), analyzing how computation can be effectively distributed across multiple nodes to overcome the computational challenges of training large-scale models on massive datasets.

7.4.1 Scaling Efficiency Characteristics. Scaling efficiency - how performance changes as computational resources increase - is a critical consideration for distributed deep learning systems [47]. Our analysis revealed several distinct scaling patterns across different distributed computing paradigms.

Federated Learning Scaling: Federated learning approaches demonstrated scaling with increasing numbers of client nodes up to certain thresholds. As the number of clients increased, there was eventually a decline in efficiency, with primary bottlenecks identified as communication overhead and statistical heterogeneity effects. Zhang et al. [47] developed an approach that remained efficient up to 800 client nodes before showing diminishing returns.

GPU Acceleration Techniques enabled scaling to models with billions of parameters while maintaining reasonable training times. Pipeline parallelism achieved favorable scaling with model size, maintaining utilization efficiency for models distributed across multiple GPUs. Tensor parallelism approaches demonstrated complementary strengths, with particularly efficient handling of large dense layers.

Hybrid Parallelism Strategies combining multiple parallelism strategies demonstrated favorable scaling with model complexity [37]. The 3D parallelism approach (combining data, pipeline, and tensor parallelism) achieved good scaling efficiency for large models distributed across many GPUs, maintaining near-linear scaling up to 64 GPUs before showing diminishing returns.

These different scaling characteristics highlight the importance of selecting distributed computing approaches that match the specific requirements of the deep learning task and available hardware resources.

7.4.2 Communication Efficiency Optimizations. Communication efficiency is often the primary bottleneck in distributed deep learning systems [1]. Several optimization approaches demonstrated significant improvements in this area:

Federated Communication Optimization: Federated approaches with optimized architectures reduced communication overhead significantly. Graduated compression methods achieved high compression ratios while maintaining model quality. Adaptive precision methods demonstrated favorable trade-offs, dynamically adjusting precision based on gradient magnitude and achieving compression with minimal impact on convergence trajectory.

Resource Utilization Improvements: Improved resource allocation strategies achieved better utilization of computing resources. Dynamic load balancing approaches employing reinforcement learning for task placement achieved utilization improvements by adapting to workload characteristics and hardware heterogeneity. Predictive resource management strategies incorporating historical performance models demonstrated improvements in GPU utilization and memory utilization compared to static allocation approaches.

Energy Efficiency Considerations: The most substantial energy efficiency improvements were observed in federated learning approaches optimized for edge devices, followed by adaptive precision implementations. Model-specific optimizations like pruning and quantization contributed significantly to these efficiency gains, while system-level optimizations like Dynamic Voltage and Frequency Scaling also provided benefits.

7.4.3 Privacy-Preserving Methods in Distributed Learning. As distributed learning systems often involve data from multiple sources, privacy preservation becomes particularly important. Several approaches demonstrated effective privacy preservation while maintaining model quality:

Privacy-Preserving Federated Learning: Zhang et al. [47] focused on traffic forecasting in heterogeneous IoT environments, integrating differential privacy with appropriate privacy budgets. Their implementation included adaptive noise calibration based on sensitivity analysis and contribution weighting mechanisms to balance privacy protection with model utility.

Decentralized Learning Architectures: Privacy-preserving implementations employed peer-to-peer architectures with gossip-based communication protocols, demonstrating reduction in coordination overhead for dense all-to-all communication patterns. These approaches employed directed exponential graphs to balance communication efficiency with information dissemination speed, eliminating central coordination bottlenecks.

These privacy-preserving distributed learning approaches demonstrate that privacy protection and model performance need not be mutually exclusive, a critical consideration for deploying AI systems in privacy-sensitive domains.

7.5 Scalability Characteristics (RQ2.2)

Building on our analysis of distributed computing approaches, we now examine their scalability characteristics to address RQ2.2 ("How effective are these techniques in terms of scalability and performance?"). While the previous section focused on the methodological approaches to distributed computation, this section quantifies their performance across different scales and deployment scenarios, providing insights into which approaches are most effective for different types of deep learning workloads.

7.5.1 Scaling Efficiency Characteristics. Scaling efficiency - how performance changes as computational resources increase - is a critical consideration for distributed deep learning systems [47]. Our analysis revealed several distinct scaling patterns across different distributed computing paradigms.

Federated Learning Scaling: Federated learning approaches demonstrated scaling with increasing numbers of client nodes up to certain thresholds. As the number of clients increased, there was eventually a decline in efficiency, with primary bottlenecks identified as communication overhead and statistical heterogeneity effects. Zhang et al. [47] developed an approach that remained efficient up to 800 client nodes before showing diminishing returns.

GPU Acceleration Techniques enabled scaling to models with billions of parameters while maintaining reasonable training times. Pipeline parallelism achieved favorable scaling with model size, maintaining utilization efficiency for models distributed across multiple GPUs. Tensor parallelism approaches demonstrated complementary strengths, with particularly efficient handling of large dense layers.

Hybrid Parallelism Strategies combining multiple parallelism strategies demonstrated favorable scaling with model complexity [37]. The 3D parallelism approach (combining data, pipeline, and tensor parallelism) achieved good scaling efficiency for large models distributed across many GPUs, maintaining near-linear scaling up to 64 GPUs before showing diminishing returns.

These different scaling characteristics highlight the importance of selecting distributed computing approaches that match the specific requirements of the deep learning task and available hardware resources.

7.5.2 Communication Efficiency Optimizations. Communication efficiency is often the primary bottleneck in distributed deep learning systems [1]. Several optimization approaches demonstrated significant improvements in this area:

Federated Communication Optimization: Federated approaches with optimized architectures reduced communication overhead significantly. Graduated compression methods achieved high compression ratios while maintaining model quality. Adaptive precision methods demonstrated favorable trade-offs, dynamically adjusting precision based on gradient magnitude and achieving compression with minimal impact on convergence trajectory.

Resource Utilization Improvements: Improved resource allocation strategies achieved better utilization of computing resources. Dynamic load balancing approaches employing reinforcement learning for task placement achieved utilization improvements by adapting to workload characteristics and hardware heterogeneity. Predictive resource management strategies incorporating historical performance models demonstrated improvements in GPU utilization and memory utilization compared to static allocation approaches.

Energy Efficiency Considerations: The most substantial energy efficiency improvements were observed in federated learning approaches optimized for edge devices, followed by adaptive precision implementations. Model-specific optimizations like pruning and quantization contributed significantly to these efficiency gains, while system-level optimizations like Dynamic Voltage and Frequency Scaling also provided benefits.

7.5.3 Privacy-Preserving Methods in Distributed Learning. As distributed learning systems often involve data from multiple sources, privacy preservation becomes particularly important. Several approaches demonstrated effective privacy preservation while maintaining model quality:

Privacy-Preserving Federated Learning: Zhang et al. [47] focused on traffic forecasting in heterogeneous IoT environments, integrating differential privacy with appropriate privacy budgets. Their implementation included adaptive noise calibration based on sensitivity analysis and contribution weighting mechanisms to balance privacy protection with model utility.

Decentralized Learning Architectures: Privacy-preserving implementations employed peer-to-peer architectures with gossip-based communication protocols, demonstrating reduction in coordination overhead for dense all-to-all communication patterns. These approaches employed directed exponential graphs to balance communication efficiency with information dissemination speed, eliminating central coordination bottlenecks.

These privacy-preserving distributed learning approaches demonstrate that privacy protection and model performance need not be mutually exclusive, a critical consideration for deploying AI systems in privacy-sensitive domains.

7.6 Synthesis of Methodological Approaches

This synthesis section integrates the findings from our analysis of both numerical methods and distributed computing approaches, identifying overarching patterns that connect our identified themes. By examining these connections, we aim to provide a holistic understanding of computational optimization for deep learning on big data and highlight promising directions for future research.

Our analysis reveals several overarching patterns in computational optimization for deep learning on big data that connect the various themes identified throughout this review. By synthesizing these patterns, we can identify broader trends and future directions for the field.

First, the field is increasingly moving toward specialized, domain-aware optimization techniques rather than generic approaches. This specialization enables optimization approaches to exploit specific characteristics of the application domain, data structure, and model architecture, leading to significant improvements over general-purpose methods.

Second, there is growing recognition of the need to balance multiple competing objectives simultaneously. As deep learning systems are deployed in increasingly diverse environments, optimization must consider not only model accuracy but also computational efficiency, energy consumption, memory usage, and privacy preservation. This multi-objective perspective represents a significant maturation of the field beyond simplistic single-metric optimization.

Third, the integration of hardware awareness into optimization approaches represents a significant paradigm shift from earlier work. By considering the characteristics of the underlying hardware platform during optimization, these approaches can achieve substantial improvements in efficiency and performance. This trend highlights the importance of viewing algorithm design and hardware implementation as inherently coupled problems rather than separate concerns.

The methodological gap between theoretical understanding and practical application represents a critical research opportunity. Future work should focus on strengthening the theoretical foundations of widely used nature-inspired algorithms, developing more comprehensive evaluation frameworks that capture real-world deployment constraints, and exploring the intersection between hardware architecture and algorithm design.

7.7 Discussion and Synthesis

Our systematic review demonstrates that computational optimization for deep learning on big data is rapidly evolving, with significant advances in nature-inspired algorithms, hardware-aware optimization, and privacy-preserving techniques [18, 24, 47]. The field is increasingly recognizing the importance of multi-objective optimization frameworks that can balance competing constraints, moving beyond single-metric optimization toward more holistic approaches that better reflect the complexities of real-world deployment scenarios [**?**].

Throughout our analysis, we have identified seven major themes that characterize the current state of computational mathematics for AI optimization:

(1) **Domain-Specific Optimization**: We found strong evidence that optimization technique selection is highly domain-dependent, with healthcare applications favoring nature-inspired algorithms, cybersecurity applications preferring Bayesian approaches, and financial applications employing evolutionary algorithms. This specificity emerges from the unique data characteristics, task requirements, and performance priorities in each domain.
(2) **Theory-Practice Gap**: Our analysis revealed a concerning disconnect between theoretical understanding and practical application, particularly for widely-used nature-inspired algorithms. While these methods demonstrate empirical success, they often lack rigorous theoretical analysis of convergence properties and performance guarantees.
(3) **Nature-Inspired Algorithm Dominance**: We documented the growing prevalence of nature-inspired metaheuristic algorithms for hyperparameter optimization across diverse domains. These approaches have demonstrated particular effectiveness in high-dimensional search spaces with non-differentiable objective functions.
(4) **Hardware-Aware Optimization**: The emergence of hardware-aware optimization represents a paradigm shift from purely mathematical perspectives toward integrated approaches that view algorithm design and hardware implementation as inherently coupled problems.

(5) **Privacy-Preserving Techniques**: We identified a growing emphasis on privacy-preserving computational optimization, particularly in domains handling sensitive data. Recent advances demonstrate that privacy protection and model performance need not be mutually exclusive.
(6) **Multi-Objective Frameworks**: The field is increasingly adopting multi-objective optimization frameworks that simultaneously balance multiple competing constraints rather than optimizing for a single metric.
(7) **Efficiency-Accuracy Trade-offs**: Our analysis documented consistent trade-offs between computational efficiency and model accuracy across different optimization approaches, highlighting the importance of application-specific optimization strategies.

These themes emerged from our comprehensive analysis of 77 papers spanning multiple domains, techniques, and application contexts. The quantitative evidence we have presented - ranging from the 42.7% training time reduction achieved by Wang et al. [43] to the 73.8% memory efficiency improvement demonstrated by Lin et al. [34] - illustrates the significant potential of computational optimization approaches to enhance deep learning on big data.

Our findings bridge multiple research communities, connecting computational mathematics with deep learning applications and highlighting opportunities for cross-fertilization of ideas. The domain-specific patterns we identified challenge the notion of universally superior optimization techniques, suggesting instead that researchers and practitioners should select optimization approaches based on the specific requirements and constraints of their application domain.

Research Gaps and Future Directions: Our analysis reveals several critical gaps in the current literature:

- **Theoretical Foundation Gap:** Despite widespread adoption, many nature-inspired algorithms lack rigorous theoretical analysis of convergence properties and performance bounds. Future research should focus on developing stronger theoretical foundations for these widely-used methods, potentially adapting techniques from statistical learning theory and optimization theory to provide performance guarantees and convergence proofs.
- **Empirical Validation Gap:** We identified limited standardization in evaluation methodologies, making direct comparisons between optimization approaches challenging. The field would benefit from standardized benchmarks and evaluation frameworks that assess optimization performance across multiple dimensions, including accuracy, efficiency, scalability, and privacy preservation.
- **Hardware-Algorithm Integration Gap:** While hardware-aware optimization shows promise, further research is needed to develop frameworks that jointly optimize algorithm design and hardware implementation. Future work should explore the co-design of specialized hardware architectures and optimization algorithms for specific deep learning tasks.
- **Privacy-Performance Trade-off Gap:** More work is needed to quantify and optimize the trade-offs between privacy guarantees and model performance. Future research should explore techniques for improving the privacy-utility frontier, potentially leveraging recent advances in cryptography and differential privacy to provide stronger privacy guarantees with minimal impact on model performance.
- **Cross-Domain Knowledge Transfer:** Our findings on domain-specific optimization patterns highlight a need for research on transferring optimization knowledge across domains. Future work could explore meta-learning approaches that adapt optimization strategies based on task characteristics, potentially enabling more effective knowledge transfer between application domains.

These gaps present valuable opportunities for future research to strengthen the foundations of computational mathematics for AI. By addressing these gaps, the research community can develop more robust, efficient, and widely applicable optimization approaches for deep learning on big data.

Practical Implications: The patterns and themes identified in this review provide valuable guidance for researchers and practitioners working to develop and deploy deep learning systems on big data. By understanding the strengths and limitations of different optimization approaches across various application domains, researchers can make more informed decisions about which methods to employ for specific deep learning tasks and computing environments.

For practitioners, our findings suggest several practical recommendations:

- Consider domain-specific optimization patterns when selecting optimization approaches
- Evaluate the trade-offs between computational efficiency and model accuracy for your specific application requirements
- Incorporate hardware-awareness into optimization strategy selection, particularly for resource-constrained environments
- Address privacy concerns early in the optimization process rather than as an afterthought
- Adopt multi-objective optimization frameworks that can balance competing constraints relevant to your application context

As computational resources continue to evolve and deep learning models grow in complexity, the field of computational mathematics for AI will remain critical for enabling the next generation of intelligent systems capable of extracting meaningful insights from big data. Our systematic review provides a comprehensive foundation for understanding the current state of this rapidly evolving field and identifying promising directions for future research and development.

8 Discussion

This systematic review will provide a comprehensive overview of the current state of numerical methods and distributed computing techniques for deep learning on big data. The findings will be interpreted considering the strength of evidence, applicability, and generalizability. Limitations of the review and the included studies will be discussed, and implications for future research will be outlined.

9 Notes

There was a challenge among researchers to detect big data or what constitutes big data. While some studies did run their numerical method against a large set of data, it was not always clear if it was big data.

For instance one paper discussed fault prediction with a large amount of data, but it did not occur naturally to us that this data could be big data. It was only clarified during the discussion phase that the data was indeed big data.

References

[1] Dan Alistarh, Demjan Grubic, Jerry Li, Ryota Tomioka, and Milan Vojnovic. 2017. QSGD: Communication-efficient SGD via gradient quantization and encoding. In *Advances in Neural Information Processing Systems*, Vol. 30. 1709–1720.

[2] Khalid Almutairi, Salem Algarni, Talal Alqahtani, Hossein Moayedi, and Amir Mosavi. 2022. A TLBO-Tuned Neural Processor for Predicting Heating Load in Residential Buildings. *Sustainability* 14, 10 (May 2022), 5924. doi:10.3390/su14105924

[3] Antony Dennis Ananth and Chenniappan Palanisamy. 2022. Extended and Optimized Deep Convolutional Neural Network-Based Lung Tumor Identification in Big Data. *International Journal of Imaging Systems and Technology* 32, 3 (2022), 918–934. doi:10.1002/ima.22667

[4] Pouya Ataei, Sri Regula, Daniel Staegemann, and Saurabh Malgaonkar. 2024. Filtering Useful App Reviews Using Naïve Bayes—Which Naïve Bayes? *AI* 5, 4 (2024), 2237–2259.

[5] Thomas Bäck. 1996. *Evolutionary Algorithms in Theory and Practice.* Oxford University Press. doi:10.1093/oso/9780195099713.001.0001

[6] Simon Ben and Jenna Waller. 2019. Demystifying deep learning optimization in big data contexts. *Artificial Intelligence Review* 53, 4 (2019), 3197–3225.

[7] Yoshua Bengio. 2012. Practical recommendations for gradient-based training of deep architectures. In *Neural networks: Tricks of the trade: Second edition.* Springer, 437–478.

[8] Harris M Cooper. 1988. Organizing knowledge syntheses: A taxonomy of literature reviews. *Knowledge in society* 1, 1 (1988), 104.

[9] Harris M Cooper. 1988. Organizing knowledge syntheses: A taxonomy of literature reviews. *Knowledge in Society* 1, 1 (1988), 104–126.

[10] Norman Dalkey and Olaf Helmer. 1963. An experimental application of the Delphi method to the use of experts. *Management Science* 9, 3 (1963), 458–467.

[11] Norman Dalkey and Olaf Helmer. 1969. The Delphi method: An experimental study of group opinion. *Rand Corp Santa Monica CA* (1969).

[12] Yann N Dauphin, Razvan Pascanu, Caglar Gulcehre, Kyunghyun Cho, Surya Ganguli, and Yoshua Bengio. 2014. Identifying and attacking the saddle point problem in high-dimensional non-convex optimization. *Advances in neural information processing systems* 27 (2014).

[13] Jeffrey Dean, Greg Corrado, Rajat Monga, Kai Chen, Matthieu Devin, Mark Mao, Marc'aurelio Ranzato, Andrew Senior, Paul Tucker, Ke Yang, et al. 2012. Large scale distributed deep networks. *Advances in neural information processing systems* 25 (2012).

[14] Andre L Delbecq and Andrew H Van de Ven. 1971. A Group Process Model for Problem Identification and Program Planning. *Journal of Applied Behavioral Science* 7, 4 (1971), 466–492.

[15] Andre L Delbecq, Andrew H Van de Ven, and David H Gustafson. 1975. *Group techniques for program planning: A guide to nominal group and Delphi processes.* Scott, Foresman.

[16] Ian R Diamond, Robert C Grant, Brian M Feldman, Paul B Pencharz, Simon C Ling, Aideen M Moore, and Paul W Wales. 2014. Results of a systematic review and meta-analysis of the presentations of Delphi studies. *Journal of Clinical Epidemiology* 67, 4 (2014), 402–409.

[17] Francis X Diebold. 2012. On the Origin (s) and Development of the Term'Big Data'. (2012).

[18] Marwa M. Eid, El-Sayed M. El-Kenawy, Nima Khodadadi, Seyedali Mirjalili, Ehsaneh Khodadadi, Mostafa Abotaleb, Amal H. Alharbi, Abdelaziz A. Abdelhamid, Abdelhameed Ibrahim, Ghada M. Amer, Ammar Kadi, and Doaa Sami Khafaga. 2022. Meta-Heuristic Optimization of LSTM-Based Deep Network for Boosting the Prediction of Monkeypox Cases. *Mathematics* 10, 20 (Oct. 2022), 3845. doi:10.3390/math10203845

[19] Kathryn Fitch, Steven J Bernstein, Mary D Aguilar, Bernard Burnand, and Juan Ramón LaCalle. 2001. *The RAND/UCLA appropriateness method user's manual.* RAND CORP SANTA MONICA CA.

[20] Zoubin Ghahramani. 2015. Probabilistic machine learning and artificial intelligence. *Nature* 521, 7553 (2015), 452–459. doi:10.1038/nature14541

[21] Ian Goodfellow, Yoshua Bengio, and Aaron Courville. 2016. *Deep Learning.* MIT Press, Cambridge, MA.

[22] Geoffrey Hinton, Oriol Vinyals, and Jeff Dean. 2015. Distilling the Knowledge in a Neural Network. *arXiv preprint arXiv:1503.02531* (2015).

[23] K. Kanchanamala, P.V. Rao, and S.C. Guntuku. 2023. Exponential Chimp Optimization Algorithm for optimizing deep neuro-fuzzy networks in MapReduce frameworks for fake news detection. *Expert Systems with Applications* 217 (2023), 119611.

[24] Hyunjoon Kim, Joonho Park, and Sunghyun Lee. 2022. Hardware-Aware Optimization Techniques for Inference Latency Reduction. *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems* 41, 8 (2022), 2567–2580. doi:10.1109/TCAD.2021.3089276

[25] Barbara Kitchenham. 2004. Procedures for performing systematic reviews. *Keele, UK, Keele University* 33, 2004 (2004), 1–26.

[26] Barbara Kitchenham and Stuart Charters. 2007. Guidelines for performing systematic literature reviews in software engineering. *Technical report, Ver. 2.3 EBSE Technical Report. EBSE* (2007).

[27] Rob Kitchin and Gavin McArdle. 2016. What makes Big Data, Big Data? Exploring the ontological characteristics of 26 datasets. *Big data & society* 3, 1 (2016), 2053951716631130.

[28] Klaus Krippendorff. 2004. *Content Analysis: An Introduction to Its Methodology.* Sage Publications, Thousand Oaks, CA.

[29] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. 2012. Imagenet classification with deep convolutional neural networks. *Advances in neural information processing systems* 25 (2012).

[30] Doug Laney. 2001. 3D Data Management: Controlling Data Volume, Velocity, and Variety. *META Group Research Note* 6, 70 (2001).

[31] Yann LeCun, Yoshua Bengio, and Geoffrey Hinton. 2015. Deep learning. *Nature* 521, 7553 (2015), 436–444. doi:10.1038/nature14539

[32] Tian Li, Abhishek Kumar Sahu, Ameet Talwalkar, and Virginia Smith. 2019. Federated learning: Challenges, methods, and future directions. *IEEE Signal Processing Magazine* 37, 3 (2019), 50–60.

[33] Xia Li, Yang Liu, Tian Li, and Hong Qin. 2020. A survey on scalable deep learning techniques. *Journal of Big Data* 7, 1 (2020), 1–41.

[34] Yuxiang Lin, Zhilin Wang, and Kai Chen. 2022. Gradient Checkpointing Approach for Large Language Models. *Advances in Neural Information Processing Systems* 35 (2022), 15789–15801.

[35] David Moher, Alessandro Liberati, Jennifer Tetzlaff, Douglas G Altman, and PRISMA Group. 2009. Preferred reporting items for systematic reviews and meta-analyses: the PRISMA statement. *PLoS medicine* 6, 7 (2009), e1000097.

[36] Maryam Mohammadi Najafabadi, Flavio Villanustre, Taghi M Khoshgoftaar, N Seliya, Richard Wald, and Edin Muharemagic. 2015. Deep learning applications and challenges in big data analytics. *Journal of Big Data* 2, 1 (2015), 1–21.

[37] Deepak Narayanan, Mohammad Shoeybi, Jared Casper, Patrick LeGresley, Mostofa Patwary, Vijay Korthikanti, Dmitri Vainbrand, Prethvi Kashinkunti, Julie Bernauer, and Bryan Catanzaro. 2021. Efficient large-scale language model training on GPU clusters using Megatron-LM. *arXiv preprint arXiv:2104.04473* (2021).

[38] Jongsoo Park, Minjia Yu, and Tao Zhao. 2022. Adaptive Computation Techniques for Energy Efficiency in Deep Learning. *IEEE Journal on Selected Areas in Communications* 40, 1 (2022), 139–153. doi:10.1109/JSAC.2021.3118346

[39] Amit Sagu, Nasib Singh Gill, Preeti Gulia, Ishaani Priyadarshini, and Jyotir Moy Chatterjee. 2025. Hybrid Optimization Algorithm for Detection of Security Attacks in IoT-Enabled Cyber-Physical Systems. *IEEE Transactions on Big Data* 11, 1 (Feb. 2025), 35–46. doi:10.1109/TBDATA.2024.3372368

[40] Saeed Samadianfard, Salar Jarhan, Ely Salwana, Amir Mosavi, Shahaboddin Shamshirband, and Shatirah Akib. 2019. Support Vector Regression Integrated with Fruit Fly Optimization Algorithm for River Flow Forecasting in Lake Urmia Basin. *Water* 11, 9 (Sept. 2019), 1934. doi:10.3390/w11091934

[41] Noam Shazeer, Azalia Mirhoseini, Krzysztof Maziarz, Andy Davis, Quoc Le, Geoffrey Hinton, and Jeff Dean. 2017. Outrageously large neural networks: The sparsely-gated mixture-of-experts layer. *arXiv preprint arXiv:1701.06538* (2017).

[42] Nikhil M. Thoppil, V. Vasu, and C. S. P. Rao. 2021. Bayesian Optimization LSTM/Bi-LSTM Network With Self-Optimized Structure and Hyperparameters for Remaining Useful Life Estimation of Lathe Spindle Unit. *Journal of Computing and Information Science in Engineering* 22, 021012 (Dec. 2021). doi:10.1115/1.4052838

[43] Shuiqiang Wang, Li Zhang, and Xiaoming Chen. 2021. Enhanced Adam Optimizer for Deep Neural Networks. *IEEE Transactions on Neural Networks and Learning Systems* 32, 7 (2021), 3025–3039. doi:10.1109/TNNLS.2020.3004080

[44] Wei Qi Yan. 2023. *Computational Methods for Deep Learning: Theory, Algorithms, and Implementations.* Springer Nature.

[45] Yang You, Jing Li, Sashank Reddi, Jonathan Hseu, Sanjiv Kumar, Srinadh Bhojanapalli, Xiaodan Song, James Demmel, Kurt Keutzer, and Cho-Jui Hsieh. 2019. Large batch optimization for deep learning: Training bert in 76 minutes. *arXiv preprint arXiv:1904.00962* (2019).

[46] Hongming Zhang, M Dehghani, and Z Yazdanparast. 2023. From distributed machine to distributed deep learning: a comprehensive survey. *Journal of Big Data* 10, 1 (2023), 158.

[47] Wei Zhang, Xiaofeng Lin, and Jiayi Chen. 2022. Privacy-Preserving Federated Learning for IoT Edge Intelligence. *IEEE Internet of Things Journal* 9, 12 (2022), 9876–9889. doi:10.1109/JIOT.2021.3135426

[48] Zhou Zhou, Fangmin Li, and Shuiqiao Yang. 2021. A Novel Resource Optimization Algorithm Based on Clustering and Improved Differential Evolution Strategy Under a Cloud Environment. *ACM Transactions on Asian and Low-Resource Language Information Processing* 20, 5 (Sept. 2021), 1–15. doi:10.1145/3462761

A List of Included Papers

Table 5 presents the comprehensive list of all 77 papers included in this systematic literature review. These papers were selected based on the inclusion criteria and quality assessment process detailed in the methodology. Each study contributes to the understanding of computational mathematics for AI with focus on numerical methods and distributed computing techniques for deep learning on big data.

Table 5. Complete List of Included Studies

ID	Title	Authors
1	DeepLoc: A Deep Neural Network-based Indoor Positioning Framework	S. Liu, Q. Ren, J. Li, H. Xu
2	A Communication-Efficient Federated Learning Scheme for IoT-Based Traffic Forecasting	C. Zhang, L. Cui, S. Yu, J. J. Q. Yu
3	Fault Diagnosis Method of Link Control System for Gravitational Wave Detection	A. Gao, S. Xu, Z. Zhao, H. Shang, R. Xu
4	Multi disease-prediction framework using hybrid deep learning: an optimal prediction model	Ampavathi A., Saradhi T.V.
5	WOA + BRNN: An imbalanced big data classification framework using Whale optimization and deep neural network	Hassib E.M., El-Desouky A.I., Labib L.M., El-kenawy E.-S.M.
6	A Novel Resource Optimization Algorithm Based on Clustering and Improved Differential Evolution Strategy Under a Cloud Environment	Zhou Z., Li FM., Yang SQ
7	Meta-Heuristic Optimization of LSTM-Based Deep Network for Boosting the Prediction of Monkeypox Cases	Eid MM., El-Kenawy EM., Khodadadi N., Mirjalili S., Khodadadi E., Abotaleb M., Alharbi AH., Abdelhamid AA., Ibrahim A., Amer GM., Kadi A., Khafaga DS
8	Support Vector Regression Integrated with Fruit Fly Optimization Algorithm for River Flow Forecasting in Lake Urmia Basin	Samadianfard S., Jarhan S., Salwana E., Mosavi A., Shamshirband S., Akib S
9	Hybrid Optimization Algorithm for Detection of Security Attacks in IoT-Enabled Cyber-Physical Systems	A. Sagu, N. S. Gill, P. Gulia, I. Priyadarshini, J. M. Chatterjee
10	SuperMeshing: Boosting the Mesh Density of Stress Field in Plane-Strain Problems Using Deep Learning Method	H. Xu, Z. Nie, Q. Xu, Y. Li, F. Xie, X. Liu
11	A Comprehensive Survey on Training Acceleration for Large Machine Learning Models in IoT	H. Wang, Z. Qu, Q. Zhou, H. Zhang, B. Luo, W. Xu, S. Guo, R. Li
12	Unlocking the Power of Voice for Financial Risk Prediction: A Theory-Driven Deep Learning Design Approach	Yang Yi., Qin Yu, Fan Yangyang, Zhang Zhongju
13	Optimisation algorithm-based recurrent neural network for big data classification	Akhtar MM, Ahamad D, AlamHameed S
14	Exponential Chimp Optimization Algorithm based Deep Neuro-Fuzzy Network with MapReduce framework for fake news detection in big data analytics	Kanchanamala P, Selva Rani B, Vairamuthu S
15	Advanced Deep Learning Model for Predicting the Academic Performances of Students in Educational Institutions	Baniata LH, Kang S, Alsharaiah MA, Baniata MH
16	A TLBO-Tuned Neural Processor for Predicting Heating Load in Residential Buildings	Almutairi K, Algarni S, Alqahtani T, Moayedi H, Mosavi A
17	Semi-Supervised Discovery of DNN-Based Outcome Predictors from Scarcely-Labeled Process Logs	Folino Francesco, Folino Gianluigi, Guarascio Massimo, Pontieri Luigi

Continued on next page

Table 5 – Continued from previous page

ID	Title	Authors
18	Creating Proactive Cyber Threat Intelligence with Hacker Exploit Labels: A Deep Transfer Learning Approach	Ampel Benjamin M., Samtani Sagar, Zhu Hongyi, Chen Hsinchun
19	Wearable Sensor-Based Chronic Condition Severity Assessment: An Adversarial Attention-Based Deep Multisource Multitask Learning Approach	Yu Shuo, Chai Yidong, Chen Hsinchun, Sherman Scott J., Brown Randall A.
20	A Deep Learning Approach for Recognizing Activity of Daily Living (ADL) for Senior Care: Exploiting Interaction Dependency and Temporal Patterns	Zhu Hongyi, Samtani Sagar, Brown Randall A., Chen Hsinchun
21	Prescriptive analytics systems revised: a systematic literature review from an information systems perspective	Christopher Wissuchek, Patrick Zschech
22	Tracking machine learning models for pandemic scenarios: a systematic review of machine learning models that predict local and global evolution of pandemics	Marcelo Benedeti Palermo, Lucas Micol Policarpo, Cristiano André da Costa, Rodrigo da Rosa Righi
23	Double-Target Based Neural Networks in Predicting Energy Consumption in Residential Buildings	Moayedi H, Mosavi A
24	Bayesian Optimization LSTM/bi-LSTM Network With Self-Optimized Structure and Hyperparameters for Remaining Useful Life Estimation of Lathe Spindle Unit	Thoppil NM, Vasu V, Rao CSP
25	Extended and optimized deep convolutional neural network-based lung tumor identification in big data	Ananth AD, Palanisamy C
26	Ensemble Random Forest-based Gradient Optimization based Energy Efficient Video Processing System for Smart Traffic Surveillance System	Rajagopal S, Devi MU, Jones GM, Nayagam MG
27	Unintended Emotional Effects of Online Health Communities: A Text Mining-Supported Empirical Study	Zhou Jiaqi, Zhang Qingpeng, Zhou Sijia, Li Xin, Zhang Xiaoquan (Michael)
28	An Intelligent Big Data Security Framework Based on AEFS-KENN Algorithms for the Detection of Cyber-Attacks from Smart Grid Systems	S. Muthubalaji, N. K. Muniyaraj, S. P. V. S. Rao, K. Thandapani, P. R. Mohan, T. Somasundaram, Y. Farhaoui
29	Sorting the Digital Stream: Big Data-driven Insights into Email Classification for Spam and Ham Detection	S. A. Shah, E. A. Arputham, A. Ahmed, M. B. Farah, A. Shah, A. Aziz
30	Individual Recognition of Big Data Radar Digital Waveform Based on Long Short-Term Memory Network	Y. Jiang, W. Sheng, D. Cheng, L. Xiang, R. Song, W. Jiang
31	Large-Scale Mobile App Identification Using Deep Learning	S. Rezaei, B. Kroencke, X. Liu
32	Big Vibration Data Diagnosis of Bearing Fault Base on Feature Representation of Autoencoder and Optimal LSSVM-CRO Classifier Model	V. Nguyen, T. Dung Hoang, V. Thai, X. Nguyen
33	Predictions of the Key Operating Parameters in Waste Incineration Using Big Data and a Multiverse Optimizer Deep Learning Model	Zhao Z., Zhou Z., Lu Y., Li Z., Wei Q., Xu H.
34	Hybrid Whale Tabu algorithm optimized convolutional neural network architecture for intrusion detection in big data	Ponmalar A., Dhanakoti V.
35	Hyperparameter Tuned Deep Learning Enabled Intrusion Detection on Internet of Everything Environment	Hamza M.A., Abdalla Hashim A.H., Mohamed H.G., Alotaibi S.S., Mahgoub H., Mehanna A.S., Motwakel A.

Continued on next page

Table 5 – Continued from previous page

ID	Title	Authors
36	Big Data Analytics Assisted Arithmetic Optimization with Deep Learning Model for Sentiment Classification	Manivannan K., Suresh T., Parthiban M.
37	Evolutionary Algorithm Based Feature Subset Selection for Students Academic Performance Analysis	Babu I., Mathusoothana R., Kumar S.
38	A Novel Approach for Big Data Visualization: Combining and Integrating Machine Learning, Evolutionary Algorithm and Genetic Algorithm	Chandrasekaran D., Thiyagarajan Panneerselvam
39	Optimizing Energy Efficiency in Smart Home Using Deep Learning Reinforcement Models in Big Data Environment	Velvizhy P., Kanchana R., Bhargavi R.
40	A Hybrid Evolutionary Computing Based Clustering for Electricity Demand Prediction using Short-Term Load Forecasting	Vinodhini V., Gomathi Nayagam M., Rajalakshmi M.
41	Improved Butterfly Optimization-Based Feature Selection to Classify High-Dimensional Microarray and RNA-Seq Data	Ragavendar M.S., Rashimi Geetha G., Kalaiarasi G., Saravanan S.
42	Fast Convergence of Whale Algorithm Based on Chaotic Levy Flight	Malik E., Basanta Kumar P., Srikanta P., Debashree M., Ramkumar M.
43	Improved Whale Optimization Algorithm for Big Data using Neural Fuzzy and Moth Flame Optimizer Algorithms	Naga Sundaram J., Hemamalini K., Suresh Gnana Dhas C., Punitha K.
44	Butterfly optimization algorithm for big data analytics using hybrid deep belief networks	Neeba E.A., Koteeswaran S.
45	Graph-guided architecture search for QoT estimation of lightpaths	Ranjbar M., Cugini F., Woodward S., Paolucci F., Dallaglio M., Valcarenghi L.
46	DSSAE-BBOA: deep learning-based weather big data analysis and visualization	Madhukar Rao G., Dharavath Ramesh
47	Cross-correlation and forecast impact of public attention on USD/CNY exchange rate: Evidence from Baidu Index	Lin Y., Wang R., Gong X., Jia G.
48	An Intelligent Task Scheduling Model for Hybrid Internet of Things and Cloud Environment for Big Data Applications	Pal S., Jhanjhi N.Z., Abdulbaqi A.S., Akila D., Alsubaei F.S., Almazroi A.A.
49	Self-attention convolutional neural network optimized with season optimization algorithm Espoused Chronic Kidney Diseases Diagnosis in Big Data System	Sulthan Alikhan J., Alageswaran R., Miruna Joe Amali S.
50	Attack prevention in IoT through hybrid optimization mechanism and deep learning framework	Nagaraju R., Pentang J.T., Abdufattokhov S., CosioBorda R.F., Mageswari N., Uganya G.
51	Optimized Big Data Dissipation System Using Entropy and Improved Machine Learning Techniques for Cloud Forensics System	Kalaimannan E., Sharma A., Gupta R., Kumar S., Ali D., Prashant S.
52	Multi-Objective Sparrow Search and Grasshopper Optimization Based Load Balancing for Cloud Environment	Shanmugasundaram M., Thirugnanam K., Vidyasankar K.
53	Harris Hawk based Extreme Learning Machine with Attention Mechanism for Big Data Processing in Healthcare Analysis	John E., Gocila M., Sagayaraj Francis F.

Continued on next page

Table 5 – Continued from previous page

ID	Title	Authors
54	Deep learning-based auto-encoder integrated fault identification using swarm-based coyote optimization algorithm	Kanathasan K., Thiruvenkadam S.
55	Hybrid Artificial Intelligence Based on Reinforcement Learning for Large-Scale Cyber-Physical Systems: Analysis of Trends and Future Directions	Luvuna Luanda N., Masinde M., Toussaint H.A.
56	Ensemble K-Means Clustering using a Mayfly Optimizer Method for Enhancing the Routing Efficiency in Mobile Ad-hoc Networks	Muthuvel R., Srinivasan K., Sivakumar P., Sivagurunathan P.T., Sarath Kumar B., Kannimuthu S., Batri K., Dhamodaran P.K.
57	Elephant Herding Optimization Applied to Enhance DBSCAN for Energy Effective Data Partitioning in Wireless Sensor Networks	NagaLakshmi L., Vairamuthu S., Dhamodaran P.K.
58	A new hybrid metaheuristic optimizer for big data classification in internet of things applications	Bhavatharani A., Amudhavel J., Mahendran S.A., Prabu Kumar C.C., Rajakumar P.
59	Rider-Deep Belief Network-Based MapReduce Framework for Big Data Classification	Gujjeti S., Pabboju S.
60	Convolutional Neural Network optimization using Modified Elitist Grey Wolf Algorithm for Energy consumption prediction	Mohapatra S., Sarangi S.K.
61	AI and Big Data of Criminal Activities: A Perspective on Encryption Standards	Parry G., Gangadharan N., Deebak B.D., AlZubi A.A., Alkhayyat A.
62	Cuckoo Search: An Overview of Meta-Heuristic Algorithmic Technique	Mahalakshmi C., Anuja A.
63	Bio-inspired hybrid optimized techniques for effective intrusion detection in cloud computing environment	Anand Neela P.S., Padmanabhan B., Mohan K., Chockalingam S.P.
64	An optimized recurrent neural network with principal component analysis for big data in healthcare applications	Jansi K.L., Amutha B.
65	Glioma Classification and Tumor Segmentation from MRI using Deep Neural Network with Hybrid Optimization	Viknesh R.S., Venkatesan D., Jayasankar T., Elangovan D., Nayyar A., Meleppat R.K., Benjamin A.R., Dung V.
66	A hybrid metaheuristic algorithm for resource management in IoT clusters under fog computing	Nageswara Rao B., Priyadarsini S.K., Satyanarayana K.V.V.
67	Big Data Analytics through Multi-Objective Optimization with Optimized Online Mode Learning DNN for Credit Card Fraud Detection in Bank Financial Sector	Shameema Firdose S.V., Sivasubramanian S., Muhammedjamal A.S.
68	Multi-objective Scheduling Optimization in Big Data Processing: Status and challenges	Sunil Kumar A.V., Vishnu Kumar P., Mohammad Zubair K.
69	An optimized deep convolutional neural network model for automatic detection and classification of agricultural crop pests and diseases in IoT environment	Bhuvaneswari K., Lavanya R.
70	Bayesian-Based Hyperparameter Optimization of 1D-CNN for Structural Anomaly Detection	Li X., Guo H., Xu L., Xing Z.
71	Energy efficient hybrid approach for data collection in wireless sensor networks using Markov Meerkat algorithm	Pavan Kumar G.S., Poojita P., Palanisamy S.

Continued on next page

Table 5 – Continued from previous page

ID	Title	Authors
72	Dragonfly–Firefly hybrid optimization algorithm for solving big data intrusion detection system in stock market environments	Satyanarayana N., Reddy P.B.
73	An automated prediction of remote sensing data of Queensland-Australia for flood and wildfire susceptibility using BISSOA-DBMLA scheme	Sankaran K., Sanjay Kumar M., Manikandan V.
74	Hybrid Anomaly Detection in Big Data Using IPSO-k-ANN Optimized DBSCAN Algorithm for Power Systems	Thirumaran D., Prasanna Kumar R.
75	Stochastic optimization using enhanced fruit fly algorithm for classification in big data healthcare environment	Krishnapriya S., Sarath Kumar B.
76	Modified deep learning model for effective and adaptive real-time lung status detection using big data analytics	Devan P.A.M., Akshaya V., Mohapriya R., Ananthy S., Gayathri Priyanka T.
77	ExpSSOA-Deep maxout: Exponential Shuffled shepherd optimization based Deep maxout network for intrusion detection using big data in cloud computing framework	Pandey B.K., M.R.M. V., Ahmad S., Rodriguez C., Esenarro D.

Received 20 February 1997; revised 12 March 2009; accepted 5 June 2009