DESDE 2023
https://rictrevista.org

RICT Revista de Investigación Científica, Tecnológica e Innovación

Publicación Semestral RICT Vol. 4 No. 8 (2026) P. 88-99

ISSN: 2992-7315

Optimizadores estocásticos adaptativos en Visión Computacional: una revisión sistemática

Stochastic adaptive optimizers in Computer Vision: a systematic review

Marcela de los Ángeles Yanes-Pérez [a], José Adán Hernández-Nolasco [a] and Miguel A. Wister [a]

[a] *División Académica de Ciencias y Tecnologías de la Información, Universidad Juárez Autónoma de Tabasco, 86690, Cunduacán, Tabasco, México*

Resumen

Esta revisión sistemática sintetiza la evidencia sobre el comportamiento de los optimizadores estocásticos adaptativos en Visión Computacional con redes neuronales profundas. Siguiendo PRISMA, se buscó en Web of Science, IEEE Xplore y SpringerLink (701 registros, enero de 2020–junio de 2026) y 79 estudios cumplieron los criterios de elegibilidad. Adam fue el comparador predominante (97.5%) y se reportó como el de mejor desempeño en el 29.9% de sus evaluaciones, proporción cercana a la esperable si los optimizadores fueran equivalentes; AdamW y AdaGrad, en el 64.7% (11 de 17) y el 35.7% (10 de 28). Estas proporciones provienen de estudios con arquitecturas, datos y configuraciones distintos; describen tendencias del corpus y no una comparación experimental común. Desde 2023, AdamW aparece recurrentemente entre los reportados con más frecuencia como de mejor desempeño, incluidos 7 de 14 estudios con Vision Transformer. El ámbito médico representó el 41.8% de los estudios y Vision Transformer solo el 17.7%, lo que sugiere una brecha en la evidencia. Los resultados no muestran que un optimizador sea superior en todos los contextos; su selección debe fundamentarse empíricamente.

Palabras clave: optimizadores adaptativos, aprendizaje profundo, visión computacional, redes neuronales convolucionales, revisión sistemática

Abstract

This systematic review synthesizes the evidence on the performance of adaptive stochastic optimizers in Computer Vision using deep neural networks. Following the PRISMA guidelines, we searched Web of Science, IEEE Xplore, and SpringerLink (701 records, January 2020–June 2026), and 79 studies met the eligibility criteria. Adam was the predominant comparator (97.5%) and was reported as the best-performing optimizer in 29.9% of its evaluations, a proportion close to what would be expected if the optimizers were equivalent; AdamW and AdaGrad were reported as the best-performing optimizers in 64.7% (11 out of 17) and 35.7% (10 out of 28) of their evaluations, respectively. These proportions come from studies with different architectures, data, and configurations; they describe trends in the corpus rather than a common experimental comparison. Since 2023, AdamW has repeatedly appeared among the optimizers most frequently reported as best-performing, including in 7 out of 14 studies using Vision Transformer. The medical field accounted for 41.8% of the studies, while Vision Transformer accounted for only 17.7%, suggesting a gap in the evidence. The results do not show that any single optimizer is superior in all contexts; its selection must be based on empirical evidence.

Keywords: adaptive optimizers, deep learning, computer vision, convolutional neural networks, systematic review

*Autor para la correspondencia: adan.hernandez@ujat.mx
Correo electrónico: marcelayanesperez@gmail.com (Marcela de los Ángeles Yanes-Pérez), adan.hernandez@ujat.mx (José Adán Hernández-Nolasco), miguel.wister@ujat.mx (Miguel A. Wister)

Historial del manuscrito: recibido el 04/10/2026, última versión-revisada recibida el 29/10/2026, aceptado el 12/11/2026, publicado el 24/11/2026. **DOI:** https://doi.org/10.5281/zenodo.24503641

1. Introduction

Computer Vision (CV) has become one of the most active areas of deep learning (DL), largely because it can automate complex visual tasks in scientific, industrial, and social contexts. DL model performance in CV does not depend solely on architecture or dataset: the training process and optimization algorithm, in particular, are methodologically relevant dimensions that influence convergence speed, learning stability, and the model's generalization capacity. A frequent practice in the literature is to adopt Adam, SGD, or RMSProp as default configurations without always justifying that choice against more recent alternatives.

This practice makes it hard to tell whether reported improvements stem from the architecture, preprocessing, data augmentation, the image dataset, or the optimization algorithm. It also limits comparison across studies, especially when results span different domains such as medical imaging, agriculture, remote sensing, autonomous driving, industrial inspection, and visual security.

1.1 Foundations of stochastic optimization

Training a deep neural network involves minimizing a loss function with respect to a set of parameters: the model weights are iteratively adjusted to reduce the error between predictions and expected outputs. The stochastic approximation approach estimates gradients from subsets of the data, which makes training on large datasets feasible without computing the exact gradient over the whole sample at every iteration (Robbins & Monro, 1951).

Stochastic gradient descent (SGD) builds on this principle by updating parameters with the gradient computed on mini-batches, reducing computational cost and serving as a fundamental methodological basis for supervised learning with neural networks (LeCun et al., 1998). However, SGD relies on a single global learning rate, which can hamper training when parameter dimensions differ in scale, curvature, or update frequency.

Momentum accelerates convergence and reduces oscillations by incorporating information from previous updates to smooth the optimization trajectory, which is especially useful on loss surfaces with steep ravines or noisy gradients (Polyak, 1964). Even so, SGD and its momentum variants retain a central limitation: the learning rate is not adapted individually to each parameter.

This limitation motivates adaptive methods, which adjust the magnitude of each update based on each parameter's historical gradient information. This property is particularly relevant in deep architectures, where layers, filters, or attention components can show highly heterogeneous learning dynamics during training.

1.2 The family of adaptive optimizers

AdaGrad was one of the first widely used methods to adapt the learning rate per parameter.

It accumulates the squares of past gradients and uses them to reduce the step size for parameters that have received large or frequent gradients (Duchi et al., 2011). This is advantageous for problems with sparse features, but it can shrink the learning rate excessively as training progresses. RMSProp addresses this drawback by replacing the full accumulation with an exponential moving average of the squared gradient, which keeps the adaptation more stable across iterations (Tieleman & Hinton, 2012).

Adam combines first-order momentum with adaptive normalization based on the gradient's second moment (Kingma & Ba, 2015). Its widespread adoption has made it a standard experimental reference, although it is not always the best alternative for every architecture.

Several variants address limitations of the original Adam formulation. AMSGrad modifies the computation of the second moment to resolve convergence issues observed in specific scenarios (Reddi et al., 2018); AdamW decouples weight decay from the adaptive gradient update to improve regularization and generalization (Loshchilov & Hutter, 2019); and RAdam rectifies the variance of the adaptive learning rate during the first iterations to stabilize the early phase of training (Liu et al., 2020).

Other proposals follow alternative update mechanisms: AdaBelief scales the step according to the discrepancy between the observed gradient and its expected value, so that the update depends on the "belief" in the gradient direction (Zhuang et al., 2020). In contrast, Lion bases its updates on the gradient sign, reducing memory consumption compared with methods that store second-order moments (Chen et al., 2023).

1.3 Adaptive optimization in computer vision

In CV, an optimizer's behavior can depend on the visual task, architecture, and application domain. Convolutional neural networks (CNNs) have been fundamental for learning visual representations and remain widely used in image classification, detection, and segmentation (LeCun et al., 1998).

Large visual datasets have enabled systematic comparisons of deep models in image recognition (Deng et al., 2009), and, more recently, Vision Transformers (ViT) have expanded the architectural repertoire by bringing attention mechanisms to visual processing (Dosovitskiy et al., 2021).

Comparing optimizers in DL is not trivial, because results can change with the learning rate, batch size, number of epochs, regularization, initialization, and evaluation criterion. Choi et al. (2019) noted that empirical comparisons between optimizers require careful control of experimental conditions, and Schmidt et al. (2021) showed that optimizers' relative performance can vary by problem and training setting. Therefore, in CV it is not advisable to assume that one optimizer is universally superior. This heterogeneity is also reflected in the applications where optimizers are evaluated, which range from medical imaging to agriculture, remote sensing, and geosciences.

1.4 Research gap and objective

Although some works address optimization in DL in general terms, to the best of our knowledge, the empirical evidence on adaptive stochastic optimizers applied to CV tasks with deep neural networks has not been the subject of an up-to-date systematic synthesis under the PRISMA statement (Preferred Reporting Items for Systematic Reviews and Meta-Analyses; Page et al., 2021a, 2021b) for the period 2020–June 2026. Therefore, this review synthesizes empirical evidence published in this domain between January 2020 and June 2026, analyzing how adaptive stochastic optimizers have been evaluated and which report the best relative performance across application domains, visual tasks, and network architectures.

2. Materials and Methods

This review followed the PRISMA statement (Page et al., 2021a, 2021b), which provides a reproducible framework for identifying, screening, assessing eligibility, and synthesizing evidence.

2.1 Review design

We aimed to synthesize empirical evidence on adaptive stochastic optimizers in CV tasks with deep neural networks. Given the heterogeneity of architectures, datasets, visual tasks, performance metrics, and training configurations among the included studies, no meta-analysis was performed. Instead, we conducted a descriptive and comparative synthesis based on frequencies, percentages, thematic cross-tabulations, and the relative frequency of best performance per optimizer. Although not an adaptive method, we included SGD as a reference comparator because it is widely used in the reviewed studies.

2.2 Research questions

The systematic review was guided by the following research questions (RQs):

- **RQ1.** Which adaptive stochastic optimizers have been empirically evaluated in CV tasks with deep neural networks between January 2020 and June 2026?
- **RQ2.** Which optimizers report the best relative performance within the included studies?
- **RQ3.** How is the evidence distributed across application domains, visual tasks, and network architectures?
- **RQ4.** What temporal trends are observed in the evaluation of adaptive stochastic optimizers over the period analyzed?
- **RQ5.** Which methodological gaps remain open for future research on adaptive optimization in CV?

2.3 Information sources and search strategy

Three bibliographic databases with international coverage in computer science and engineering were searched: Web of Science (WoS), IEEE Xplore, and SpringerLink. The search strings were built from two conceptual blocks joined by the AND operator: the first groups the main adaptive stochastic optimizers and their variants, and the second delimits CV tasks with deep neural networks. The string used in WoS was:

```
  TS=("adaptive optimizer" OR
"stochastic optimizer" OR "Adam
optimizer" OR "AdamW" OR "AdaGrad" OR
"RMSProp" OR "AMSGrad" OR "RAdam" OR
"AdaBelief" OR "Lion optimizer") AND
TS=("image classification" OR
"computer vision" OR "deep learning"
OR "convolutional neural network" OR
"CNN" OR "vision transformer" OR
"ViT")
```

Equivalent strings were adapted to the syntax of IEEE Xplore (Abstract field) and SpringerLink (Keywords with discipline filters). We conducted the searches in June 2026 and restricted them to January 2020–June 2026 and to original research articles in English. In WoS, category filters were applied (Computer Science, Engineering, and Mathematics); in IEEE Xplore, only the Journal Articles option was selected; and in SpringerLink, the disciplines Computer Science and Engineering were used, with the subdisciplines Artificial Intelligence and Computer Vision.

2.4 Eligibility criteria

The inclusion criteria (IC) and exclusion criteria (EC) were defined a priori and are detailed below. Table 1 summarizes them.

Table 1: Inclusion and exclusion criteria

Code	Criterion description
IC1	Original research articles published in peer-reviewed scientific journals indexed in WoS, IEEE Xplore, or SpringerLink.
IC2	Articles published between January 2020 and June 2026.
IC3	Articles written in English.
IC4	The article empirically evaluates at least one adaptive stochastic optimizer (Adam, AdamW, AdaGrad, RMSProp, AMSGrad, RAdam, AdaBelief, Lion, or a derived variant) in a CV task with deep neural networks (classification, detection, segmentation, recognition, or image analysis using CNNs, ViTs, or hybrid architectures), through direct comparison, a proposed improved variant, or an analysis of the optimizer's effect on model performance.
IC5	The article reports at least one quantitative performance metric (accuracy, F1-score, AUC-ROC, precision, recall, mAP, IoU, Dice, or validation loss) that allows assessing the optimizer's behavior on the visual task studied.
EC1	Literature reviews, surveys, meta-analyses, conference papers, book chapters, editorials, letters to the editor, abstracts, and preprints without formal publication.

EC2	Articles whose main application domain is not CV: natural language processing, time series, non-visual biomedical signals (ECG, EEG), numerical prediction, recommendation, or general regression.
EC3	Articles in which the optimizer appears only as an implementation detail, without explicit comparison with alternatives or analysis of the optimizer's impact on model performance.
EC4	Duplicate records not removed during automatic deduplication, identified during manual screening by title, authors, and year of publication.
EC5	Articles in languages other than English, or whose full text could not be retrieved after two attempts through institutional resources and contact with the authors.

Note. IC = inclusion criterion; EC = exclusion criterion. Reporting the learning rate, batch size, or other implementation decisions does not, by itself, amount to an optimizer comparison (see EC3).

2.5 Synthesis methods

Since the 79 included studies showed substantial heterogeneity in network architectures, datasets, visual tasks, performance metrics, and training configurations, no quantitative meta-analysis was performed. Instead, a narrative and descriptive synthesis was carried out following the SWiM guideline (Synthesis Without Meta-analysis; Campbell et al., 2020).

The synthesis was organized around four dimensions: (1) evaluation frequency per optimizer, calculated as the number of articles in which each optimizer was included as a comparator; (2) best-performance rate, defined as the ratio between the number of best-performance cases and the number of evaluations of each optimizer, expressed as a percentage; (3) thematic distribution by application domain, visual task, and network architecture; and (4) temporal trend of the most frequent best-performing optimizer by publication year.

The cross-tabulations between best-performing optimizer, domain, and architecture were built from the audited extraction matrix, which is provided in full as Supplementary File 1 (the list of the 79 included studies is given in Appendix D). Cells corresponding to ties or mixed results between two or more optimizers, as well as articles in which the best-performing optimizer was not reported clearly enough (NR), were recorded as separate categories and were not counted as a best-performance case for any optimizer. However, the corresponding evaluations were kept in the denominator of each evaluated optimizer.

2.6 Selection process

The records retrieved from the three databases were imported into Zotero (version 7), where duplicates were removed through automatic deduplication followed by manual review. Screening was carried out in three consecutive phases: (1) screening by title and abstract, applying IC1–IC3 and EC1–EC5; (2) fine screening by extended abstract for records that did not allow a clear decision in the previous phase; and (3) eligibility assessment through full-text reading, in which compliance with IC4 and IC5 was verified.

The complete flow of identification, screening, retrieval, eligibility, and inclusion is presented in Figure 1, prepared in accordance with the PRISMA statement.

2.7 Data extraction

A structured extraction matrix was built for each of the 79 included studies with the following fields: internal identifier (Matrix ID), authors and year, title, journal, DOI, application domain, main visual task, network architectures used, dataset, dataset size, optimizers evaluated, type of optimizer evaluation (direct comparison, proposed variant, or hyperparameter analysis), best-performing optimizer according to the main reported metric, performance metrics, key result, relevant experimental configuration (learning rate, batch size, scheduler), validation scheme, limitations reported by the authors, and inclusion note (justification of IC4/IC5).

In this article, "best performance" refers to the optimizer that the authors of each included study reported as obtaining the highest value on their main metric, under their own experimental conditions. It was extracted as reported and was not recalculated or verified through a common experimental comparison; therefore, it should not be interpreted as evidence that an optimizer is superior in general.

2.8 Methodological quality and reporting assessment

In accordance with PRISMA item 11, the methodological quality and reporting quality of the 79 included studies were assessed using a 15-item checklist adapted by the authors to the specific study domain (Supplementary Material, Appendix C). This checklist is not a validated risk-of-bias instrument; its scores should be read as an indicator of methodological and reporting quality, not as a strict measure of risk of bias. Each item was scored dichotomously (1 = met, 0 = not met or not reported), for a total possible score of 0-15, classified into three levels: high quality (12–15), moderate quality (8–11), and low quality (≤7).

3. Results

Of the 79 included studies, Adam was the most frequently evaluated optimizer (97.5%). It achieved the best performance in 29.9% of evaluations, while AdamW had the highest best-performance rate (64.7%), followed by Lion (60.0%, five evaluations) and AdaGrad (35.7%). These proportions come from different sets of studies and should be read as trends of the analyzed corpus, not as a common experimental comparison. All figures reported below were calculated from the extraction matrix (Supplementary File 1).

3.1 General characteristics and study selection

The selection process began with 701 records identified in WoS, IEEE Xplore, and SpringerLink. After removing 95 duplicates, 606 records were screened by title and abstract, of which 208 were excluded for not meeting the eligibility criteria.

Subsequently, 398 articles were sought for full-text retrieval; 261 (65.6%) could not be retrieved in full text after two attempts through institutional resources and contact with the authors (criterion EC5; the list is provided in Supplementary Material, Appendix E). The 261 articles not retrieved were distributed across all publication years (2020–2026) in a pattern similar to that of the 137 retrieved articles.

Reading the 137 retrieved full texts led to the exclusion of 58 additional articles and the retention of 79 articles for the final synthesis (Figure 1). Supplementary Material (Appendix A) presents the 58 articles excluded after full-text review, along with the reason for exclusion. The search strings adapted for WoS, IEEE Xplore, and SpringerLink are documented in Supplementary Material (Appendix B).

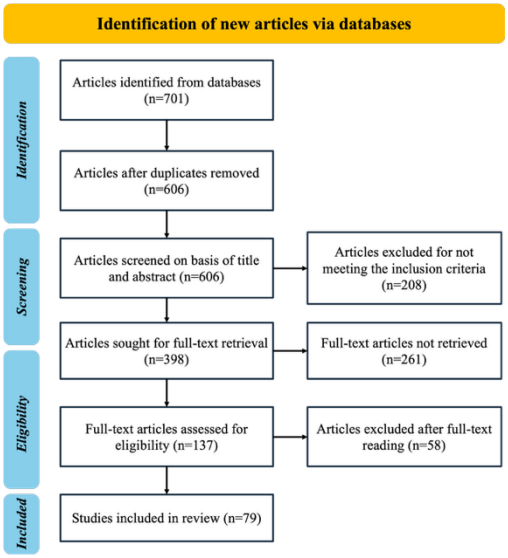

Figure 1: Flow diagram of the article selection process following PRISMA.

The selected articles were distributed across 38 peer-reviewed journals. IEEE Access accounted for the largest share of publications (n = 14; 17.7%), followed by Applied Sciences-Basel (n = 9; 11.4%), Mathematics (n = 7; 8.9%), and Ain Shams Engineering Journal and Electronics (n = 4 each; 5.1%).

IEEE Transactions on Neural Networks and Learning Systems contributed three articles (3.8%). The diversity of publication sources suggests that research on adaptive stochastic optimization applied to CV is cross-disciplinary, spanning journals in AI, machine learning, engineering, applied mathematics, and computer science.

3.2 Temporal distribution of the evidence

Scientific output increased from 2022 onward, with 16 articles per year in 2022 and 2023, followed by a relatively stable trend in the later years. The eight articles counted for 2026 cover only the first half of the year because the search closed in June; annual output could therefore remain stable or increase toward year-end.

Table 2: Temporal distribution of the best-performing optimizer by year (2020–2026)

Year	N	Optimizer(s) with the best-performance cases	2nd	3rd
2020	5	RMSProp (3)	Adam (1)	Nadam (1)
2021	7	AdaGrad (2)	Adam (2)	RMSProp (1)
2022	16	Adam (4)	RMSProp (3)	SGD (3)
2023	16	Adam (6)	AdaGrad (3)	AdamW/ RMSProp (2)
2024	15	AdamW (4)	Adam (3)	SGD (3)
2025	12	Adam (5)	AdamW (3)	AdaGrad/ Lion/SGD (1)
2026*	8	Adam (2)	AdamW (2)	SGD (2)

Note. N = articles published per year; values in parentheses are articles in which the optimizer performed best on the main reported metric. *2026 covers publications up to June.

The temporal pattern does not show a single dominant optimizer. In 2020, RMSProp accounted for most of the reported cases, and in 2021 AdaGrad and Adam tied; from 2022, Adam was the optimizer with the most best-performance cases in three of the four full years (2022, 2023, and 2025), while AdamW led in 2024 and ranked among the top positions in 2023, 2025, and the first half of 2026, sharing the position with Adam or SGD in some years. This pattern is examined in more depth in Section 3.6.

3.3 Application domains and visual tasks

The medical domain was the largest category, accounting for 41.8% of included articles (n = 33). This group included studies on radiological image classification, tumor detection, skin lesion analysis, diabetic retinopathy diagnosis, magnetic resonance image processing, and infectious disease detection, as well as breast cancer, melanoma, pneumonia, and pathological image segmentation.

In these settings, the optimizer was evaluated alongside architectures such as CNN, ResNet, VGG, MobileNet, U-Net, and ViT, and metrics such as accuracy, sensitivity, specificity, AUC, F1-score, and loss were reported. Included studies such as Qureshi et al. (2022), Raiaan et al. (2023), Mehmood et al. (2025), Jozi and Al-Suhail (2025), and Aly et al. (2024) illustrate this concentration of the medical domain within the reviewed evidence. Among the most frequently used specific medical datasets were HAM10000 (n = 4), BraTS (n = 3), BR35H (n = 3), ISIC (n = 3), and EyePACS (n = 2), which suggests a relevant concentration of studies in dermatology, neuroimaging, and medical ophthalmology (Figure 2).

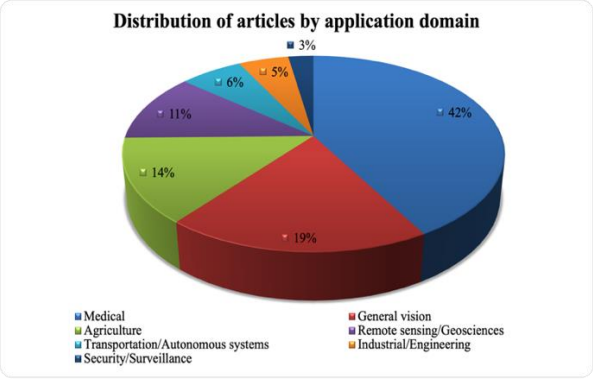

Figure 2: Distribution of articles by application domain.

The medical domain was the most represented category, although it did not account for the majority of the corpus. A further 19.0% of the articles (n = 15) were classified as general computer vision: studies that propose or benchmark optimizers on generic datasets such as CIFAR, MNIST, or ImageNet, without a specific application domain. Studies in agriculture, remote sensing, engineering, transportation, and security show that optimizer analysis extends beyond computer-aided diagnosis to multiple CV contexts.

Table 3: Distribution by application domain and best-performing optimizers

Domain	N	%	1st best	2nd best	3rd best
Medical	33	41.8%	Adam (10)	SGD (8)	AdamW (6)
General vision	15	19.0%	AdaGrad (5)	Adam/AdamW (3)	AdaCB/MRGD Adam/SGD(1)
Agriculture	11	13.9%	Adam (4)	RMSProp (3)	Nadam (2)
Remote sensing/ Geosciences	9	11.4%	RMSProp (3)	Adam (2)	AdaGrad /AdamW /Lion (1)
Transportation/ Autonomous systems	5	6.3%	Adam (2)	AdaGrad/ AdamW/RMSProp (1)	—
Industrial/ Engineering	4	5.1%	Adam (1)	AdaGrad (1)	Lion (1)
Security/ Surveillance	2	2.5%	AdaGrad (1)	Adam (1)	—

Note. N = included articles; values in parentheses are articles where the optimizer performed best in that domain.

Agriculture accounted for 13.9% of the articles (n = 11), with articles on pest detection, plant disease classification, tomato leaf diseases, nematodes, fruit recognition, and pathogen detection in crops using deep models trained with different optimizers. Studies such as Saedi and Khosravi (2020), Wang et al. (2020), Saranya et al. (2023), Shabrina et al. (2023), Athira and Hemalatha (2026), and Jagdev et al. (2025) incorporated optimizer selection as an experimental variable in agricultural visual inspection scenarios.

Remote sensing and geosciences grouped nine publications (11.4%), including thin-section rock classification and satellite image analysis; studies such as Polat et al. (2021), Li et al. (2022), Cenet et al. (2025), Song et al. (2023), Yang et al. (2026), and Jawahar et al. (2023) provide evidence of the use of adaptive optimization in hyperspectral images, thin sections, and geospatial visual analysis.

In these domains, the best-performing optimizers varied (RMSProp led in remote sensing, and Adam in agriculture and medicine), suggesting that an optimizer's relative performance may depend on image type, architecture, and the primary evaluation metric.

For visual tasks, image classification predominated in 78.5% of articles (n = 62), followed by visual detection tasks (63.3%; n = 50), often combined with classification. Visual recognition (30.4%; n = 24) and semantic segmentation (15.2%; n = 12) were identified to a lesser extent. This distribution is consistent with the historical evolution of CV benchmarks, in which image classification has served as a reference task since ImageNet (Deng et al., 2009).

3.4 Datasets and architectures

The most frequently documented datasets or reference sources were ImageNet (n = 40; 50.6%), Kaggle repositories (n = 23; 29.1%), CIFAR-10 (n = 16; 20.3%), and MNIST (n = 14; 17.7%). ImageNet's presence should be interpreted as both a reference dataset and a frequent source of pretraining for deep architectures.

This distribution suggests a methodological preference for standardized benchmarks and widely accessible repositories, which favors reproducibility and comparison across optimization algorithms. In contrast, eight publications (10.1%) used their own image collections or collections strictly tied to a particular domain, which introduces additional challenges for assessing the generalization of the reported results.

Table 4 presents the most frequent deep neural network architectures and the best-performing optimizer within each group. The ResNet family was the most used architecture (n = 62; 78.5%), followed by generic CNN configurations (n = 60; 75.9%) and classic architectures such as VGG (n = 54; 68.4%), MobileNet (n = 35; 44.3%), and AlexNet (n = 33; 41.8%). ViT-based models were identified in 14 articles (17.7%), a low proportion considering the growing influence of these architectures in CV since the work of Dosovitskiy et al. (2021). Likewise, Swin Transformer and ConvNeXt still appeared with limited presence in the corpus, with seven (8.9%) and three (3.8%) studies, respectively.

Table 4: Deep neural network architectures and most frequent best-performing optimizer by architecture

Architecture	N	%	Top optimizer	2nd	3rd
ResNet	62	78.5%	Adam (15)	AdamW (10)	RMSProp (9)
CNN (generic)	60	75.9%	Adam (16)	AdaGrad/ RMSProp (10)	AdamW (7)
VGG	54	68.4%	Adam (16)	SGD (8)	RMSProp (7)
MobileNet	35	44.3%	Adam (11)	AdamW/ SGD (5)	RMSProp (4)
AlexNet	33	41.8%	Adam (12)	AdamW (5)	AdaGrad/ RMSProp /SGD (4)
Inception	32	40.5%	Adam (10)	RMSProp (8)	SGD (6)
DenseNet	27	34.2%	Adam (6)	AdaGrad/ AdamW (4)	RMSProp /SGD (3)
U-Net	19	24.1%	Adam (7)	AdamW (4)	SGD (3)
EfficientNet	18	22.8%	Adam (5)	AdamW (4)	SGD (3)
YOLO	15	19.0%	Adam (4)	AdamW/ SGD (3)	AdaGrad/ Lion/ Nadam/ RMSProp (1)
Xception	15	19.0%	Adam (6)	RMSProp (4)	SGD (2)
ViT	14	17.7%	AdamW (7)	Adam (4)	SGD (2)
Swin Transformer	7	8.9%	AdamW (4)	Adam/ Lion/ SGD (1)	—
ConvNeXt	3	3.8%	AdamW (3)	—	—

Note. N indicates the number of articles in which each architecture appears.

The architecture pattern showed differences. In ViT, AdamW was reported as the best-performing optimizer in seven of 14 articles (50.0%). In Swin Transformer, it was reported as best-performing in four of seven articles (57.1%), and in all three ConvNeXt articles.

Although these subsets are small and do not allow definitive generalizations, the results suggest a possible empirical affinity between AdamW and attention-based or recently designed architectures. In contrast, in classic convolutional architectures such as ResNet, VGG, and AlexNet, Adam was the most frequent best performer (15 of 62, 16 of 54, and 12 of 33 articles, respectively), although in only 24% to 36% of them, with the remaining cases distributed among AdamW, SGD, RMSProp, and AdaGrad.

Overall, ResNet, CNN, VGG, MobileNet, AlexNet, Inception, and DenseNet appeared more frequently than transformer-based models, suggesting that most evaluations focused on convolutional or derived architectures. This distribution matters because optimizers may behave differently across convolutional architectures, attention-based networks, and hybrid models.

3.5 Optimizers evaluated and best-performance rates

Table 5 presents how often each optimizer was evaluated in the included articles, along with its best-performance rate. The latter was defined as the percentage of studies in which an optimizer was reported as the best-performing one, considering only those in which that optimizer was evaluated and taking as reference the main metric declared by the authors.

Table 5: Optimizers evaluated, comparators, and best-performance rates

Optimizer	Times evaluated	% of articles	Best-performance cases	Best-performance rate
Adam	77	97.5%	23	29.9%
SGD	71	89.9%	10	14.1%
RMSProp	60	75.9%	10	16.7%
AdaGrad	28	35.4%	10	35.7%
Nadam	19	24.1%	3	15.8%
AdamW	17	21.5%	11	64.7%
Adamax	14	17.7%	0	0.0%
RAdam	9	11.4%	0	0.0%
AMSGrad	8	10.1%	0	0.0%
Lion	5	6.3%	3	60.0%
AdaBelief	4	5.1%	0	0.0%
AdaCB	1	1.3%	1	100.0%
MRGDAdam	1	1.3%	1	100.0%
HDAdam	1	1.3%	0	0.0%
Tie/mixed	—	—	5	—
Not reported (NR)	—	—	2	—

Note. Rate = (best-performance cases/times evaluated) × 100. Highlighted: optimizers with ≥5 evaluations and a rate above 55%. Ties/mixed results (n = 5) and not-reported cases (NR, n = 2) count as no optimizer's win, but their evaluations remain in the denominator. Single-evaluation rates (AdaCB, MRGDAdam, HDAdam) are not interpretable; AdaCB and MRGDAdam were proposed by the original authors.

Adam was evaluated in 77 of the 79 included articles (97.5%), making it the most frequently referenced comparator in the analyzed set. It achieved the best relative performance in 23 of those evaluations (29.9%), which is close to the expected number if the optimizers compared in each study were equivalent (23.1 expected cases; see the reference below), so its wide adoption did not translate into a consistent advantage in the reported comparisons. AdamW, evaluated in 17 articles (21.5%), had the highest best-performance proportion among the optimizers with more than ten evaluations: 11 of 17 cases (64.7%).

Its best results were distributed across the medical domain (6 cases), general computer vision (3), remote sensing (1), and transportation (1), and were frequent in modern architectures, including those based on attention mechanisms. AdaGrad, with 28 evaluations (35.4%), obtained the best performance in 10 of them (35.7%), with five cases in general computer vision studies and isolated cases in agriculture, industrial imaging, remote sensing, transportation, and security, and none in the medical articles. Lion (Chen et al., 2023) appeared in five articles (6.3%) and was reported as the best-performing optimizer in three of them (60.0%).

Although the size of this subset was small, its relative performance positions Lion as an emerging optimizer of interest, especially in tasks such as cell nucleus segmentation (Chen et al., 2025), defect detection in photovoltaic panels (Elbedwehy et al., 2026), and hyperspectral classification (Song et al., 2023). In contrast, Adamax, RAdam, AMSGrad, and AdaBelief did not obtain the best performance in any of the articles in which they were evaluated, a result interpreted in Section 4.1.

As a descriptive reference, if the optimizers compared in each study were equivalent, the expected number of best-performance cases for each optimizer would be the sum, over the studies in which it was evaluated, of one divided by the number of optimizers compared in that study. Under this reference, the observed cases were close to the expected ones for Adam (23 observed, 23.1 expected) and Nadam (3 and 3.2), below them for SGD (10 and 19.6) and RMSProp (10 and 15.8), and above them for AdamW (11 and 4.1), AdaGrad (10 and 5.5), and Lion (3 and 1.9). This reference is not a statistical test and ignores differences in architectures, datasets, metrics, and configurations across studies; it is offered only to contextualize the proportions in Table 5.

AdaCB, MRGDAdam, and HDAdam appeared with a single evaluation each, so their proportions are not interpretable. Overall, these results show that the most widely used optimizers were not always those that reported the best relative performance. This difference takes on an additional dimension when the temporal and architectural patterns are analyzed, as developed in the following subsection.

3.6 Temporal trend of the best-performing optimizer

Table 2, presented in Section 3.2, does not suggest a single linear transition among the best-performing optimizers. In 2020, RMSProp accounted for most of the reported cases (3 of 5 articles), and in 2021 AdaGrad and Adam tied (two cases each).

From 2022, Adam led in the best-performing cases in three of the four full years (four cases in 2022, six in 2023, and five in 2025), which coincided with the frequent use of classic convolutional architectures.

AdamW led in 2024 (four cases) and also ranked among the top positions in 2023 (two cases), 2025 (three cases), and the first half of 2026 (two cases, tied with Adam and SGD). This pattern may be interpreted as a sign of the growing adoption of variants with explicit decoupling of regularization, especially in modern architectures and in scenarios where training is more sensitive to hyperparameter configuration.

However, the number of cases per year is small. As a methodological complement for interpreting these trends, the validation schemes reported by the included articles were reviewed. The train/test split was the most common scheme (n = 77; 97.5%), followed by an explicit validation set (n = 27; 34.2%) and k-fold cross-validation (n = 19; 24.1%). Only five articles used external validation on independent datasets (6.3%), which should be considered when interpreting the optimizer's reported generalization.

3.7 Synthesis by research question

This subsection integrates the findings of the previous sections with the five research questions that guided the systematic review.

RQ1. Which adaptive stochastic optimizers have been empirically evaluated in CV tasks with deep neural networks between January 2020 and June 2026?

The included articles identified fourteen distinct optimizers (Table 5). Adam (97.5%, n = 77), SGD (89.9%, n = 71), and RMSProp (75.9%, n = 60) appear in most articles and, in practice, serve as near-mandatory reference comparators in the experimental design of the reviewed studies. AdaGrad (35.4%, n = 28), Nadam (24.1%, n = 19), AdamW (21.5%, n = 17), and Adamax (17.7%, n = 14) appear with a notable but much smaller presence. At the opposite end, more recent variants such as Lion (6.3%, n = 5) and AdaBelief (5.1%, n = 4) are scarcely evaluated within the analyzed set. This asymmetric distribution suggests that the research community's experimental attention remains concentrated on classic optimizers. At the same time, more recent proposals, despite their theoretical grounding and growing adoption in other areas of DL, still lack a comparable empirical base in the specific domain of CV.

RQ2. Which optimizers report the best relative performance within the included articles?

Considering the proportion of articles in which each optimizer was reported as the best performer, defined in Section 2.5 as the ratio between the number of best-performance cases and the number of evaluations of each optimizer, AdamW shows the highest proportion (64.7%, 11 cases in 17 evaluations), followed by Lion (60.0%, 3 cases in 5 evaluations) and AdaGrad (35.7%, 10 cases in 28 evaluations) (Table 5).

Adam shows a lower proportion (29.9%) despite its near-universal evaluation, which is close to the proportion expected if all compared optimizers were equivalent, as examined in Section 4. These proportions come from different numbers of evaluations and heterogeneous studies, so they describe trends of the analyzed corpus and do not establish the superiority of any optimizer. Adamax, RAdam, AMSGrad, and AdaBelief did not achieve the best performance in any article in which they were evaluated, a result interpreted with caution in Section 4.1 given the small number of evaluations available for several of these optimizers.

RQ3. How is the evidence distributed across application domains, visual tasks, and network architectures?

The evidence is unevenly distributed across application domains, visual tasks, and network architectures (Tables 3 and 4). The medical field accounts for 41.8% of the articles (n = 33), followed by general vision (19.0%, n = 15), agriculture (13.9%, n = 11), remote sensing and geosciences (11.4%, n = 9), transportation/autonomous systems (6.3%, n = 5), industrial/engineering (5.1%, n = 4), and security/surveillance (2.5%, n = 2). By visual task, image classification predominates (78.5%, n = 62), followed by object detection (63.3%, n = 50), recognition (30.4%, n = 24), and semantic segmentation (15.2%, n = 12). By architecture, ResNet, generic CNNs, and VGG are the most frequent, while ViT appears in only 17.7% of the articles (n = 14) despite its growing influence in recent CV benchmarks and models. This uneven distribution has a direct implication for RQ2: given that AdamW concentrates its highest best-performance proportions in attention-based architectures, and that the medical domain is led by Adam, SGD, and AdamW, the available evidence does not allow determining whether the advantage of AdamW reflects an intrinsic property of the optimizer or the thematic concentration of the analyzed articles in those particular contexts.

RQ4. What temporal trends are observed in the evaluation of adaptive stochastic optimizers over the analyzed period?

The number of articles grew steadily from 2022 onward, reaching 16 publications in both 2022 and 2023 (Table 2). The distribution of the most frequent best-performing optimizer by year does not show a single linear transition: RMSProp led in 2020, AdaGrad and Adam tied in 2021, and Adam had the most best-performance cases in 2022, 2023, and 2025; AdamW led in 2024 and appears recurrently among the most frequent best-performing optimizers from 2023 onward, sharing the position with Adam and SGD in the eight articles corresponding to the first half of 2026.

This temporal progression may relate to the growing adoption of AdamW in training attention-based architectures (Liu et al., 2021; Touvron et al., 2021) and large-scale multimodal models (Radford et al., 2021). However, the reviewed articles do not establish a causal relationship between the two phenomena, only a temporal coincidence, which is discussed in more detail in Section 4.1.

RQ5. Which methodological gaps remain open for future research on adaptive optimization in CV?

From the preceding analysis, four methodological gaps are identified that could guide future research in this field. The first concerns the evaluation of optimizers in ViT and related architectures, which account for only 17.7% of the articles despite their growing role in current classification benchmarks; given that AdamW was most frequently reported as the best-performing optimizer in the few evaluations available for these architectures, extending this line of evidence would help confirm or qualify that association.

The second refers to Lion, which has only five evaluations across the articles, a number insufficient to support generalizable recommendations. However, it was reported as best-performing in three of those five evaluations.

The third is the scarcity of external validation: only 6.3% of the articles used independent datasets, limiting the ability to assess the generalization capacity of the reported results. The fourth gap is cross-cutting: heterogeneity in architectures, datasets, learning rates, learning rate schedulers, and validation schemes hinders direct comparison across articles, so greater standardization in reporting experimental configurations would facilitate future quantitative syntheses in this field.

3.8 Methodological quality and reporting assessment

The assessment showed that 92.4% of the included articles (n = 73) reached the high-quality level, 7.6% reached the moderate level (n = 6), and no article scored ≤ 7. These results reflect the completeness of methodological reporting and should not be interpreted as a strict estimate of risk of bias. The mean score was 13.3/15. The items with the lowest compliance were 12, 4, and 15, related mainly to identifying study limitations, describing the image dataset used in detail, and linking the findings to prior literature in the discussion section, respectively. The full per-article detail is presented in Supplementary Material (Appendix C).

4. Discussion

The central finding is that frequency of use does not coincide with relative performance. Adam is the most evaluated optimizer in the set. Yet, it was reported as the best-performing optimizer in fewer than a third of its evaluations (29.9%), a proportion close to what would be expected if the compared optimizers were equivalent. This does not imply that Adam is ineffective; in the reviewed literature, it appears to serve mainly as a reference comparator.

Moreover, with an average of about four optimizers compared per article, Adam's 23 best-performance cases (29.9%) are close to the number expected if all optimizers were equivalent (23.1), so its rate does not, by itself, indicate inferiority or superiority. In other words, Adam is a reasonable starting point, but not necessarily the most competitive alternative under specific architecture, domain, and metric conditions.

In contrast, AdamW shows the most favorable relative results among the optimizers with a reasonable number of evaluations (64.7%), followed by AdaGrad (35.7%). This difference suggests that optimizer selection should not be treated as a secondary or routine decision in CV publications. Rather, the data suggest that the optimizer may influence final model performance, especially when combined with modern architectures, specialized datasets, or specific regularization and learning-rate scheduling schemes. However, the heterogeneity of the included studies prevents quantifying that influence.

4.1 Interpretation in the context of the literature and experimental practice

The findings align with prior debates about the limitations of adaptive methods. Although Adam has been widely adopted since its original formulation (Kingma & Ba, 2015), previous research has pointed out that adaptive optimizers do not always generalize better than simpler or explicitly regularized alternatives (Wilson et al., 2017). In this sense, the relative advantage observed for AdamW is coherent with the proposal to decouple weight decay from the adaptive gradient update, which allows more controlled regularization during training (Loshchilov & Hutter, 2019).

The prominent presence of AdamW in ViT, Swin Transformer, and ConvNeXt is also relevant. Although the number of articles on these architectures is still small, the pattern suggests that AdamW may be especially competitive in modern models sensitive to regularization and hyperparameter fine-tuning. However, this association must be interpreted with caution: the available evidence does not allow claiming causality, since the articles differ in datasets, sample sizes, schedulers, number of epochs, learning rates, and evaluation criteria.

AdaGrad also deserves attention. Although its mechanism of historical gradient accumulation can progressively reduce the effective learning rate (Duchi et al., 2011), it obtained the best result in 10 of 28 evaluations (35.7%), above the proportion expected under equivalence (19.5%), but with cases dispersed across general computer vision, agriculture, industrial, remote sensing, transportation, and security studies, and none in the medical articles.

This dispersion, together with the heterogeneity of the studies, does not allow identifying the conditions under which AdaGrad is competitive, and any interpretation requires controlled experimental validation. Adamax, RAdam, AMSGrad, and AdaBelief were never reported as the best performer in any article where they were evaluated (14, 9, 8, and 4 evaluations, respectively). This should be read with caution: they are usually included as secondary comparators, with few evaluations each, and their performance depends on each study's hyperparameter configuration.

Therefore, we cannot conclude that they are inferior to Adam or AdamW in CV. From a practical perspective, the results suggest no universally superior optimizer for CV. In medical, agricultural, industrial, or remote sensing applications, researchers should justify the optimizer choice empirically rather than defaulting to Adam.

For researchers and developers, this means reporting explicit comparisons, hyperparameters, schedulers, validation schemes, and metrics for each optimizer so results are reproducible and comparable.

4.2 Limitations of the evidence and potential biases

The synthesized evidence has strengths but also important limitations. The first is the methodological heterogeneity of the included articles. They differ in architecture, domain, dataset size, train/test split, main metric, and optimizer configuration.

This variability prevents direct quantitative comparisons across articles and limits the ability to formulate universal recommendations. A second limitation is the scarcity of external validation. Although most articles used train/test partitions, only a small proportion used independent datasets. This limits the ability to assess whether the reported advantages hold outside the original experimental set. In highly sensitive domains such as medical imaging, this limitation is especially relevant because internal performance may not adequately reflect clinical or institutional generalization.

Possible publication bias must also be considered. Articles in which a new or less conventional optimizer outperforms Adam are likely to be reported more often than those in which no relevant differences are observed. Therefore, Adam ranking as the best performer in a minority of comparisons (29.9%) should not be interpreted in isolation as definitive evidence of inferiority, but as an indication that its widespread adoption does not guarantee empirical superiority in all scenarios.

This systematic review was limited to WoS, IEEE Xplore, and SpringerLink, and to English-language articles published between January 2020 and June 2026. Excluding other databases such as Scopus, ACM Digital Library, or ScienceDirect may have omitted relevant publications. In addition, the definition of "best performance" depended on the main metric reported in each article, introducing variability across domains and tasks.

Moreover, 261 of the 398 articles sought (65.6%) could not be retrieved in full text and therefore could not be assessed for eligibility. If these results differ systematically from those of the 137 retrieved, the findings may be affected by selection bias, and the proportions reported here should be interpreted as trends in the retrieved corpus.

4.3 Implications and future directions

The results have theoretical and practical implications. Theoretically, they reinforce the need to study interactions among the optimizer, architecture, and application domain, rather than evaluating optimizers in isolation. The evidence suggests that AdamW could be particularly relevant in modern architectures and that Adam remains a competitive baseline in classic convolutional architectures.

Still, both observations require more controlled comparative studies. In practical terms, future research should avoid reporting the optimizer as an implementation detail.

The training configuration should be presented in sufficient detail, including learning rate, batch size, number of epochs, scheduler, regularization, best-model selection criterion, and validation scheme.

This would allow results to be compared more rigorously and would move the field toward more solid recommendations for specific CV tasks. Four priority research lines are identified. First, expand the evaluation of optimizers on ViT, Swin Transformer, ConvNeXt, and hybrid architectures, where evidence remains limited.

Second, studying emerging optimizers such as Lion in greater depth, whose relative performance was high but is based on few evaluations. Third, incorporating external validation and multicenter testing, especially in medical imaging. Fourth, promoting standardized experimental protocols that allow optimizers to be compared under controlled conditions.

5. Conclusions

This systematic review synthesized the empirical evidence on adaptive stochastic optimizers in CV with deep neural networks from 79 articles published between 2020 and June 2026, identified following the PRISMA statement in WoS, IEEE Xplore, and SpringerLink. The main findings are as follows. Adam is the most evaluated optimizer (97.5% of the articles) and obtains the best performance in 29.9% of its evaluations, a proportion close to the one expected if the optimizers compared in each study were equivalent (about 30%, with an average of four optimizers per article), so its widespread adoption does not appear to reflect a consistent empirical advantage according to the reviewed evidence. AdamW stands out with the highest best-performance proportion (64.7%), followed by Lion (60.0%, few evaluations) and AdaGrad (35.7%); these proportions come from different sets of studies and should be read as trends of the analyzed corpus.

AdamW was most frequently reported as best-performing in attention-based architectures (ViT and Swin Transformer) and in all three ConvNeXt evaluations, a modernized CNN with a transformer-inspired design; since 2023, it has appeared recurrently among the best-performing optimizers (leading in 2024), though it shares positions with Adam or SGD depending on the year. Lion was reported as best-performing in three of its five evaluations (60.0%); given how few evaluations exist, this is a gap in the available evidence.

The medical domain is the largest category, with 41.8% of included articles; within it, Adam, SGD, and AdamW are the optimizers most frequently reported as best, whereas in agriculture Adam and RMSProp compete more evenly. ViT appears in only 17.7% of articles despite being the reference architecture in current benchmarks, so evaluating optimizers on ViT and its variants is a further gap in the available evidence. From a practical perspective, the results suggest considering domain-specific empirical evidence, the architecture, and dataset characteristics before adopting Adam as the default configuration.

Future research should include AdamW as an additional baseline alongside Adam, systematically expand the evaluation of Lion across multiple domains and architectures, incorporate external validation schemes, and standardize the reporting of experimental configurations to facilitate quantitative synthesis.

The main conclusion is not that a universally superior optimizer exists, but that performance depends on interactions among architecture, domain, data, and experimental configuration.

Acknowledgements

The authors gratefully acknowledge the Secretariat of Science, Humanities, Technology and Innovation (SECIHTI) for the financial support provided through the doctoral scholarship awarded to the first author (CVU: 1131812) and for the resources contributed to the development of this research.

Data availability

Appendices A (excluded studies with reasons for exclusion), B (search strings by database), C (methodological quality assessment per article), D (list of the 79 included studies), and E (articles not retrieved in full text), together with the complete audited extraction matrix, are available as supplementary material.

Conflict of interest

The authors declare no conflict of interest related to this article.

Author contributions

Marcela de los Ángeles Yanes-Pérez: conceptualization, methodology, investigation, data curation, formal analysis, visualization, writing – original draft, and writing – review and editing. José Adán Hernández-Nolasco: conceptualization, methodology, supervision, and writing – review and editing. Miguel A. Wister: conceptualization, methodology, supervision, and writing – review and editing. All authors have read and agreed to the final version of the manuscript.

Ethics approval

Not applicable. This investigation analyzed published literature and involved no human participants or sensitive data.

Declaration on the use of AI tools

The authors did not use generative artificial intelligence tools to produce the content of this article.

6. References

Aly, M., Ghallab, A., & Fathi, I. S. (2024). Tumor ViT-GRU-XAI: Advanced brain tumor diagnosis framework: Vision transformer and GRU integration for improved MRI analysis: A case study of Egypt. IEEE Access, 12, 184726–184754. https://doi.org/10.1109/access.2024.3513235

Athira, P. S., & Hemalatha, S. (2026). Cosine annealing scheduler added with momentum factor: Classification of tomato plant diseases. Ain Shams Engineering Journal, 17(7), Article 104218. https://doi.org/10.1016/j.asej.2026.104218

Campbell, M., McKenzie, J. E., Sowden, A., Katikireddi, S. V., Brennan, S. E., Ellis, S., Hartmann-Boyce, J., Ryan, R., Shepperd, S., Thomas, J., Welch, V., & Thomson, H. (2020). Synthesis without meta-analysis (SWiM) in systematic reviews: Reporting guideline. BMJ, 368, Article l6890. https://doi.org/10.1136/bmj.l6890

Cenet, R., Ünsal, E., & Canbaz, O. (2025). Classification of hydrothermal alteration types from thin-section images using deep convolutional neural networks. Applied Sciences, *15*(22), Article 12274. https://doi.org/10.3390/app152212274

Chen, X., Liang, C., Huang, D., Real, E., Wang, K., Pham, H., Dong, X., Luong, T., Hsieh, C.-J., Lu, Y., & Le, Q. V. (2023). Symbolic discovery of optimization algorithms. Advances in Neural Information Processing Systems, 36, 49205–49233. https://doi.org/10.52202/075280-2140

Chen, Z. P., Peng, S. L., Yang, R., Zhao, M., & Zhang, C. L. (2025). AL-Net: Adaptive learning for enhanced cell nucleus segmentation in pathological images. Electronics, 14(17), Article 3507. https://doi.org/10.3390/electronics14173507

Choi, D., Shallue, C. J., Nado, Z., Lee, J., Maddison, C. J., & Dahl, G. E. (2019). On empirical comparisons of optimizers for deep learning. arXiv. https://arxiv.org/abs/1910.05446

Deng, J., Dong, W., Socher, R., Li, L.-J., Li, K., & Fei-Fei, L. (2009). ImageNet: A large-scale hierarchical image database. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR) (pp. 248–255). https://doi.org/10.1109/CVPR.2009.5206848

Dosovitskiy, A., Beyer, L., Kolesnikov, A., Weissenborn, D., Zhai, X., Unterthiner, T., Dehghani, M., Minderer, M., Heigold, G., Gelly, S., Uszkoreit, J., & Houlsby, N. (2021). An image is worth 16×16 words: Transformers for image recognition at scale. In Proceedings of the International Conference on Learning Representations (ICLR 2021).

Duchi, J., Hazan, E., & Singer, Y. (2011). Adaptive subgradient methods for online learning and stochastic optimization. Journal of Machine Learning Research, 12, 2121–2159.

Elbedwehy, S., Turki-Hadj Alouane, M., Hassan, E., & Tarek, Z. (2026). LiCPV: Hybrid deep learning for PV fault detection. Ain Shams Engineering Journal, 17(3), Article 104041. https://doi.org/10.1016/j.asej.2026.104041

Jagdev, S., Sundararaman, B., Khatri, N., Gaur, P., & Mewada, H. (2025). SqueezeNet-based deep learning framework for accurate tomato (Solanum lycopersicum) leaf disease diagnosis and classification. International Journal of Computational Intelligence Systems, 18(1), Article 229. https://doi.org/10.1007/s44196-025-00978-2

Jawahar, M., Anbarasi, L. J., Jasmine, S. G., Daya, J. L. F., Ravi, V., & Chakrabarti, P. (2023). TRS-Net: Tropical revolving storm disasters analysis and classification based on multispectral images using 2-D deep convolutional neural network. Multimedia Tools and Applications, 82(30), 46651–46671. https://doi.org/10.1007/s11042-023-15450-6

Jozi, N. S., & Al-Suhail, G. A. (2025). LCxNet: An explainable CNN framework for lung cancer detection in CT images using multi-optimizer and visual interpretability. Applied System Innovation, 8(5), Article 153. https://doi.org/10.3390/asi8050153

Kingma, D. P., & Ba, J. (2015). Adam: A method for stochastic optimization. In Proceedings of the 3rd International Conference on Learning Representations (ICLR 2015). https://arxiv.org/abs/1412.6980

LeCun, Y., Bottou, L., Bengio, Y., & Haffner, P. (1998). Gradient-based learning applied to document recognition. Proceedings of the IEEE, 86(11), 2278–2324. https://doi.org/10.1109/5.726791

Li, D. Y., Zhao, J. J., & Ma, J. Y. (2022). Experimental studies on rock thin-section image classification by deep learning-based approaches. Mathematics, 10(13), Article 2317. https://doi.org/10.3390/math10132317

Liu, L., Jiang, H., He, P., Chen, W., Liu, X., Gao, J., & Han, J. (2020). On the variance of the adaptive learning rate and beyond. In Proceedings of the 8th International Conference on Learning Representations (ICLR 2020).

Liu, Z., Lin, Y., Cao, Y., Hu, H., Wei, Y., Zhang, Z., Lin, S., & Guo, B. (2021). Swin Transformer: Hierarchical vision transformer using shifted windows. In Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV) (pp. 9992–10002). https://doi.org/10.1109/ICCV48922.2021.00986

Loshchilov, I., & Hutter, F. (2019). Decoupled weight decay regularization. In Proceedings of the 7th International Conference on Learning Representations (ICLR 2019).

Mehmood, F., Mehmood, A., & Whangbo, T. K. (2025). Alzheimer's disease detection in various brain anatomies based on optimized vision transformer. Mathematics, 13(12), Article 1927. https://doi.org/10.3390/math13121927

Page, M. J., McKenzie, J. E., Bossuyt, P. M., Boutron, I., Hoffmann, T. C., Mulrow, C. D., Shamseer, L., Tetzlaff, J. M., Akl, E. A., Brennan, S. E., Chou, R., Glanville, J., Grimshaw, J. M., Hróbjartsson, A., Lalu, M. M., Li, T., Loder, E. W., Mayo-Wilson, E., McDonald, S., … Moher, D. (2021a). The PRISMA 2020 statement: An updated guideline for reporting systematic reviews. BMJ, 372, Article n71. https://doi.org/10.1136/bmj.n71

Page, M. J., Moher, D., Bossuyt, P. M., Boutron, I., Hoffmann, T. C., Mulrow, C. D., Shamseer, L., Tetzlaff, J. M., Akl, E. A., Brennan, S. E., Chou, R., Glanville, J., Grimshaw, J. M., Hróbjartsson, A., Lalu, M. M., Li, T., Loder, E. W., Mayo-Wilson, E., McDonald, S., … McKenzie, J. E. (2021b). PRISMA 2020 explanation and elaboration: Updated guidance and exemplars for reporting systematic reviews. BMJ, 372, Article n160. https://doi.org/10.1136/bmj.n160

Polat, O., Polat, A., & Ekici, T. (2021). Automatic classification of volcanic rocks from thin section images using transfer learning networks. Neural Computing and Applications, 33(18), 11531–11540. https://doi.org/10.1007/s00521-021-05849-3

Polyak, B. T. (1964). Some methods of speeding up the convergence of iteration methods. USSR Computational Mathematics and Mathematical Physics, 4(5), 1–17.

Qureshi, M. N., Umar, M. S., & Shahab, S. (2022). A transfer-learning-based novel convolution neural network for melanoma classification. Computers, 11(5), Article 64. https://doi.org/10.3390/computers11050064

Radford, A., Kim, J. W., Hallacy, C., Ramesh, A., Goh, G., Agarwal, S., Sastry, G., Askell, A., Mishkin, P., Clark, J., Krueger, G., & Sutskever, I. (2021). Learning transferable visual models from natural language supervision. In Proceedings of the 38th International Conference on Machine Learning (ICML), PMLR 139, pp. 8748–8763.

Raiaan, M. A. K., Fatema, K., Khan, I. U., Azam, S., Rashid, M. R. U., Mukta, M. S. H., Jonkman, M., & De Boer, F. (2023). A lightweight robust deep learning model gained high accuracy in classifying a wide range of diabetic retinopathy images. IEEE Access, 11, 42361–42388. https://doi.org/10.1109/access.2023.3272228

Reddi, S. J., Kale, S., & Kumar, S. (2018). On the convergence of Adam and beyond. In Proceedings of the 6th International Conference on Learning Representations (ICLR 2018).

Robbins, H., & Monro, S. (1951). A stochastic approximation method. The Annals of Mathematical Statistics, 22(3), 400–407.

Saedi, S. I., & Khosravi, H. (2020). A deep neural network approach towards real-time on-branch fruit recognition for precision horticulture. Expert Systems with Applications, 159, Article 113594. https://doi.org/10.1016/j.eswa.2020.113594

Saranya, T., Deisy, C., Sridevi, S., & Anbananthen, K. S. M. (2023). A comparative study of deep learning and Internet of Things for precision agriculture. Engineering Applications of Artificial Intelligence, 122, Article 106034. https://doi.org/10.1016/j.engappai.2023.106034

Schmidt, R. M., Schneider, F., & Hennig, P. (2021). Descending through a crowded valley — Benchmarking deep learning optimizers. In Proceedings of the 38th International Conference on Machine Learning (ICML 2021), PMLR 139, pp. 9367–9376.

Shabrina, N. H., Lika, R. A., & Indarti, S. (2023). Deep learning models for automatic identification of plant-parasitic nematode. Artificial Intelligence in Agriculture, 7, 1–12. https://doi.org/10.1016/j.aiia.2022.12.002

Song, Y. L., Wang, A. L., Zhao, Y., Wu, H. B., & Iwahori, Y. J. (2023). Multi-scale spatial-spectral attention-based neural architecture search for hyperspectral image classification. Electronics, 12(17), Article 3641. https://doi.org/10.3390/electronics12173641

Tieleman, T., & Hinton, G. (2012). Lecture 6.5 — RMSProp: Divide the gradient by a running average of its recent magnitude. COURSERA: Neural Networks for Machine Learning, 4(2), 26–31.

Touvron, H., Cord, M., Douze, M., Massa, F., Sablayrolles, A., & Jégou, H. (2021). Training data-efficient image transformers & distillation through attention. In Proceedings of the 38th International Conference on Machine Learning (ICML), PMLR 139, pp. 10347–10357.

Wang, J., Li, Y., Feng, H., Ren, L., Du, X., & Wu, J. (2020). Common pests image recognition based on deep convolutional neural network. Computers and Electronics in Agriculture, 179, Article 105834. https://doi.org/10.1016/j.compag.2020.105834

Wilson, A. C., Roelofs, R., Stern, M., Srebro, N., & Recht, B. (2017). The marginal value of adaptive gradient methods in machine learning. Advances in Neural Information Processing Systems, 30, 4148–4158.

Yang, W., Li, A., Zhang, L., & Qin, X. (2026). Deep learning-based intelligent analysis of rock thin sections: From cross-scale lithology classification to grain segmentation for quantitative fabric characterization. Electronics, 15(7), Article 1509. https://doi.org/10.3390/electronics15071509

Zhuang, J., Tang, T., Ding, Y., Tatikonda, S. C., Dvornek, N., Papademetris, X., & Duncan, J. (2020). AdaBelief optimizer: Adapting stepsizes by the belief in observed gradients. Advances in Neural Information Processing Systems, 33, 18795–18806.