ADVANCED AND PIONEER RESEARCH IN COMPUTER SCIENCES AND ENGINEERING



All Sciences Academy

ADVANCED AND PIONEER RESEARCH IN COMPUTER SCIENCES AND ENGINEERING

Editor
Assoc. Dr. Mücella Özbay Karakuş



Advanced and Pioneer Research in Computer Sciences and Engineering
Editor: *Assoc. Dr. Mücella Özbay Karakuş*

Design: All Sciences Academy Design
Published Date: October 2026
Publisher's Certification Number: 72273
ISBN: 978-625-5472-33-5

© All Sciences Academy

www.allsciencesacademy.com
allsciencesacademy@gmail.com

CONTENT

1. Chapter **6**

From Pulse Curves to Network Accuracy: A Practical Guide to Simulating On-Chip Learning with Memristive Synapses

Baki GÖKGÖZ

2. Chapter **23**

Evidential Concept Bottlenecks for Colposcopic Swede Scoring: A Data-Audited Case Study on the BUET Multi-Center Colposcopy Dataset

Halil İbrahim COŞAR, Esra GÜNGÖR ULUTAŞ

3. Chapter **47**

Comparative Evaluation of Nine Deep Transfer Learning Architectures for Multi-Class Brain Tumor MRI Classification: A Stratified 5-Fold Cross-Validation Study with Calibration and Grad-CAM Analysis

Halil İbrahim COŞAR, Esra GÜNGÖR ULUTAŞ, Recep Batuhan GÜNAY

4. Chapter **78**

Runtime-Reflexive Architectures: Continuous Evaluation Mechanisms as the Backbone of Adaptive AI-Powered Software Systems

Halit ÇETİNER, Sedat METLEK

5. Chapter **90**

Explainable Deep Classification of Urinary Pathogens on Chromogenic Media

Serpil USTEBAY

6. Chapter **108**

Modern Cryptographic Communication Algorithms and Secure Strategies

Veli AKAY, Muhammet ATEŞ

7. Chapter **131**

Breaking Boundaries in Natural Language Processing: The Road from RNNs to Transformers

Abdo IBRAHIM, Ammar Mohammed Ali AL-QADASI, Bilal YOUSEFI

8. Chapter **145**

A Machine Learning-Based Proactive Framework for Wildfire Risk Prediction and Early Warning Using Historical Fire Records

Mohanad Ehab Al-Hajj, AHMED NAJIB AHMED ABDO, ABDULRAHMAN MOHAMMED AHMED ALHALQI, Ammar Mohammed Ali Al-Qadasi

From Pulse Curves to Network Accuracy: A Practical Guide to Simulating On-Chip Learning with Memristive Synapses

Baki GÖKGÖZ[1]

1. Asst. Prof. Dr.; Gümüşhane University, Torul Vocational School, Department of Computer Technologies, Gümüşhane, Türkiye. bakigokgoz@gumushane.edu.tr ORCID No: 0000-0001-8166-6282

ABSTRACT

This chapter is about the distance between a measured memristive synapse and the accuracy figure reported for it. Between the two lie a pulse protocol, a fitted model, a simulator and a set of training runs, and each adds error that papers rarely quantify. I follow these steps with a synthetic device, Device X, whose true parameters are known, so that the error can be measured rather than guessed. Fitting the depression nonlinearity (true value 2.0) from a single measured cycle gave values between 1.62 and 2.42 over repeated campaigns; with three cycles the range was 1.77 to 2.25, and with ten cycles 1.89 to 2.12. Passed to a 400-100-10 perceptron trained on MNIST, the single-cycle range alone moved test accuracy by 0.7 points. Two configurations whose accuracies differed by 0.14 points were put in the right order by one random seed 64 % of the time and by five seeds only 78 % of the time. Along the way the chapter compares the open-source frameworks NeuroSim, IBM AIHWKit and CrossSim, and sets out the baselines, controls, hyperparameter policies and metrics a study needs. All numbers in the chapter come from simulations written for it. It closes with a reporting checklist and a list of common pitfalls.

Keywords – memristor, on-chip learning, device modelling, simulation methodology, reproducibility, NeuroSim, AIHWKit

INTRODUCTION

Most memristive synapses are measured on a probe station and then judged in software. What happens in between is a long sequence of choices. Someone picks a pulse protocol, normalises the raw curves, fits a model, hands the parameters to a simulator and decides how many training runs to do. Papers usually report the end of that sequence, an accuracy figure, and say little about the steps that produced it. Yet each of those steps can move the figure, sometimes by more than the effect the paper is about.

I wrote this chapter as a practical guide to that sequence. It is meant for readers who have measured, or are about to measure, potentiation and depression curves and want to know what the curves mean for on-chip learning. Device physics and the circuit-level non-idealities of analog in-memory computing are left out on purpose; Sebastian et al. (2020) and Yu (2018) review them well. The chapter is built around a single workflow,

sketched in Figure 1. I stop at each of its steps to look at what tends to go wrong there, and at what a reader of the paper would need in order to spot it.

Two worked examples run through the text. In the first, I fit the nonlinearity of a synthetic device to noisy pulse data and ask how many measurement cycles give a usable estimate. In the second, I carry the leftover uncertainty into a network simulation and ask how many random seeds are needed before two configurations can be told apart. The device in both examples, which I call Device X, is invented. That is deliberate. Because its true parameters are known, one can see exactly how far each step drifts from the truth.

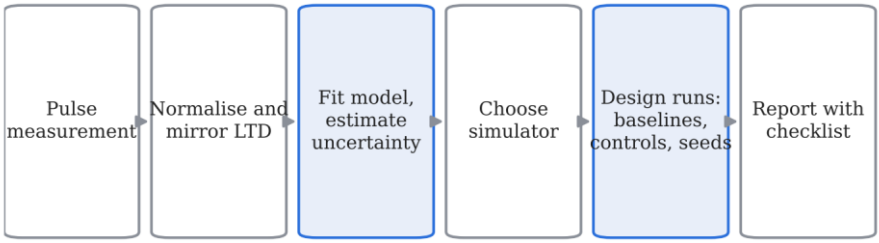

Figure 1: Workflow from device to system used in this chapter

The two shaded steps are where the worked examples measure an uncertainty that most papers leave out.

FROM PULSE MEASUREMENT TO A DEVICE MODEL

What to Measure

Everything in this chapter starts from one kind of data: the conductance read, at a low voltage that does not disturb it, after each pulse in a train of identical pulses. For Device X a cycle is 64 potentiating pulses followed by 64 depressing ones, and the worked examples use campaigns of one, three or ten cycles. Table 2 lists the quantities a simulator later takes from such data, and Gokmen and Vlasov (2016) give the tolerances on them that training on a

resistive cross-point array needs. Three protocol details change the curves and belong in the methods section:

- *Pulse scheme.* Device X is driven with identical pulses, which is why both of its branches bend. Ramping the amplitude would straighten them, at the price of a more complex driver.
- *Read voltage.* If it is too high, each read slightly reprograms the cell.
- *Number of cycles and devices.* For Device X, one cycle left the depression nonlinearity anywhere between 1.62 and 2.42 (Figure 3).

Normalising and Mirroring

The simulated cycles of Device X were normalised the way the NeuroSim fitting scripts do it (Chen et al., 2018; Luo et al., 2019). Conductance becomes $g = (G - G_{min})/(G_{max} - G_{min})$, and the depression branch is flipped so that it, too, runs from 0 at pulse 0 to 1 at pulse 64. That is why both panels of Figure 2 share one axis, and why one formula with a separate curvature per branch can fit both.

For Device X the conductance window was fixed by construction. With measured data I would take G_{min} and G_{max} from the whole data set rather than cycle by cycle, so that a drifting window stays visible to the model. I would also keep the first pulses of each branch: on the true depression curve of Device X, half of the change happens in the first 18 of 64 pulses.

Fitting the Nonlinear Update Model

NeuroSim models each branch as a saturating exponential in the pulse number p (Chen et al., 2018):

$$g_{LTP}(p) = B(1 - e^{-p/A}) \tag{1}$$

$$B = \frac{1}{1 - e^{-P_{max}/A}} \tag{2}$$

B fixes $g(P_{max}) = 1$ and A sets the curvature. Device papers rarely quote A. What they quote is NL, a label without units, and the step from NL to A differs between simulators; NeuroSim has a lookup for it. Here I use

$$A = \frac{P_{max}}{NL} \tag{3}$$

With $P_{max} = 64$ this gives A = 64 for the potentiation branch of Device X (NL = 1.0) and A = 32 for its depression branch (NL = 2.0); NL = 0 would be a straight line. I chose the relation because it is easy to invert, not because it is a standard. Whichever mapping a study uses should be written down,

because the same NL passed to two simulators with different conventions describes two different devices.

Each branch then has one unknown. I fitted it by least squares over a grid of 1 200 NL values between 0.01 and 6; a grid cannot get stuck, and with 65 points per branch it is cheap. How far the fitted value can be trusted is the subject of the first example.

How Many Cycles? A Worked Example

Device X has P_{max} = 64 levels, NL_{LTP} = 1.0 and NL_{LTD} = 2.0. In the simulated measurements every point gets Gaussian read noise (σ = 0.02 in normalised units), and the curvature of each cycle varies randomly by about 10 % (log-normal, σ = 0.1) as a stand-in for cycle-to-cycle variation. I simulated campaigns of 1, 3 and 10 cycles, fitted NL to each, and repeated every campaign 200 times. Figure 2 shows one three-cycle campaign; Figure 3 summarises all 200 repetitions.

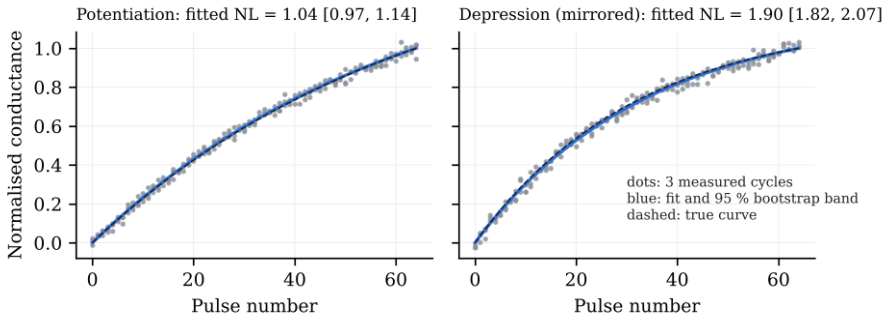

Figure 2: One simulated three-cycle measurement of Device X

Dots are the measured points, the blue line is the least-squares fit with its 95 % bootstrap band over cycles, and the dashed line is the true curve. The depression branch is mirrored so that it starts at pulse 0.

With one cycle, the middle 95 % of fitted values spanned 0.82 to 1.19 for potentiation and 1.62 to 2.42 for depression. Three cycles brought the depression range down to 1.77 to 2.25, and ten cycles brought it to 1.89 to 2.12. The more curved branch was the harder one to pin down; its interval was about twice as wide as the potentiation interval at every campaign size. None of this shows in a plotted curve. In Figure 2 the fit lies almost on top of the true curve, yet repeating the same three-cycle protocol could have produced an NL_{LTD} anywhere in a band 0.48 wide.

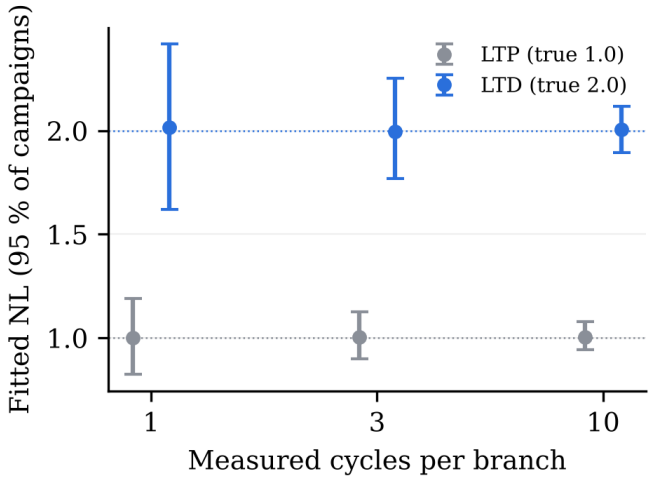

Figure 3: Spread of the fitted nonlinearity of Device X over 200 repeated measurement campaigns with 1, 3 and 10 cycles per branch

Points are means, bars cover the central 95 % of campaigns, and dotted lines mark the true values.

In the laboratory nobody repeats a campaign 200 times. What one can do is resample the cycles already measured. For the campaign in Figure 2, a bootstrap with 400 resamples of its three cycles (Efron and Tibshirani, 1993) gave $NL_{LTD} = 1.90$ with a 95 % interval from 1.82 to 2.07. The true value lies inside that interval, but the interval is only about half as wide as the real campaign-to-campaign spread for three cycles (0.25 against 0.48). Three cycles are simply too few for the bootstrap to pick up variation that never showed up in them, so I read its interval as a lower bound. What I take from this example is not complicated. Three cycles should be the minimum and more than one device is better; a fitted value should come with an interval, and the paper should say how that interval was obtained.

Other Update Models

The exponential form is not the only choice. In soft-bounds models (see Fusi and Abbott, 2007), each pulse moves the conductance by an amount proportional to how far it still is from the bound it is heading for:

$$\Delta g_{+} = \alpha_{+}(1 - g), \quad \Delta g_{-} = -\alpha_{-} g \tag{4}$$

The resulting curves saturate just as Equation (1) does. What changes is that the asymmetry between the two branches is now carried by just two

numbers, the step sizes α_+ and α_-. Device models of this type are included in the IBM Analog Hardware Acceleration Kit (Rasch et al., 2021). A third route drops the functional form altogether and drives the simulation from a lookup table of measured conductance changes against present conductance. Tables keep details that a two-parameter model smooths out. The price is data. A table needs many more measurements per device before it can be trusted, and two devices described by tables are hard to compare at a glance. Whichever model is chosen, it should be tested on cycles held out of the fit, not only on the cycles it was fitted to.

Variability

The same fitting procedure yields variability estimates almost for free. Fitting each cycle on its own and taking the spread of the fitted parameters gives the cycle-to-cycle variation; doing the same across devices gives the device-to-device variation. Simulators usually take these as standard deviations of either the update step or the model parameters, and a paper should say which. A common slip, and an easy one to avoid, is to quote variability as a bare percentage without saying what it is a percentage of: the conductance window, the step per pulse or the curvature parameter.

CHOOSING A SIMULATION FRAMEWORK

Several open-source frameworks take device parameters in and give network-level results out. They differ in whether they simulate training on the array or only inference, in whether they estimate circuit costs such as area, energy and latency, and in how they connect to deep-learning libraries. Table 1 lists five that are widely used. These tools change quickly. The entries describe the public releases as of October 2026, and they should be checked against each project's documentation before use.

Table 1: Open-source frameworks for device-to-system simulation of analog in-memory learning (as of October 2026)

Framework	On-chip training	Circuit cost estimates	Notes
MLP+NeuroSim V3.0 (Luo et al., 2019)	Yes, two-layer perceptron	Area, latency, energy	C++; nonlinear LTP/LTD with variation; close to the device-paper use case
DNN+NeuroSim V2.0 (Peng et al., 2021)	Yes, deep CNNs	Area, latency, energy	PyTorch front end with C++ hardware model
NeuroSim V1.5 (Read et al., 2025)	No (inference)	Yes	Statistical noise injection; reported up to 6.5× faster than V1.4
IBM AIHWKit 1.1 (Rasch et al., 2021; Le Gallo et al., 2023)	Yes, analog SGD and Tiki-Taka	No	PyTorch; device presets; PCM inference noise model
CrossSim 3.2 (Feinberg et al., 2026)	Hardware-aware training in PyTorch; full on-array training in V2.0	No	Python, GPU; PyTorch and Keras interfaces

Source: Compiled by the author from the documentation and release notes of each project (October 2026).

Which one to pick depends on the question. If the aim is to see how a measured device behaves when a small network is trained on chip, with energy and area reported next to accuracy, the NeuroSim training versions are the natural fit. If the aim is to see how a device or an update algorithm scales to modern deep networks, with accuracy as the main output, AIHWKit is better suited; it also implements training algorithms intended for asymmetric devices, such as Tiki-Taka (Gokmen and Haensch, 2020). For inference accuracy under read noise and quantisation, NeuroSim V1.5 and CrossSim both work. The NeuroSim circuit models were checked against post-layout simulation of a 40 nm RRAM macro (Lu et al., 2021). Any paper that reports circuit costs, whichever framework produced them, should say whether a calibration of this kind covers its own technology node.

Whatever the choice, the version or commit and every edited source file go into the paper (Table 2, simulator row). And before anything is changed, the framework's reference configuration should be run once, with a note that its published baseline came out as expected.

DESIGNING THE SIMULATION STUDY

Baselines and Controls

Two reference runs belong in almost every device study:

- a floating-point network with the same architecture, data and schedule;
- an ideal device, linear and symmetric, with the same number of levels as the real one.

With these, the hardware cost splits in two:

- floating point minus ideal device: the cost of the finite number of levels;
- ideal device minus real device: the cost of nonlinearity.

Controls then change one factor at a time. In the second example below only NL_{LTD} changes, and everything else, the learning rates included, stays fixed. The 0.58-point gap between NL_{LTD} = 1.62 and 2.0 in Figure 4 is therefore a device effect and nothing else.

Hyperparameters

In the second example the learning rates (0.30 and 0.15) were the same for every configuration and were not tuned. Holding them fixed is one defensible policy. The other is to tune every configuration on a validation split with the same search budget. Either way the paper should say which was done and how large the search was (Bouthillier et al., 2021). Tuning only one configuration would not have been defensible here, because a learning rate that suits NL_{LTD} = 1.62 need not suit 2.42.

What a Seed Controls

In the second example the seed set the initial weights, the order in which the 60 000 training images were presented, and every coin flip of the stochastic rounding (Gupta et al., 2015). At NL_{LTD} = 2.0 the ten seeds gave accuracies from 92.04 % to 93.00 %, a standard deviation of 0.38 points (Figure 4). Differences between configurations have to be read against that spread (Henderson et al., 2018). Simulators that also draw write noise or device variation from the seed will show a wider one.

How Many Seeds? A Worked Example

The second example uses the following set-up:

- network: a 400-100-10 sigmoid perceptron trained online on MNIST (LeCun et al., 1998);
- weights: one Device X cell per weight, read against a constant reference;
- updates: stochastically rounded pulses, with learning rates of 0.30 and 0.15 for the two layers.

Only NL_{LTD} changed between configurations. I set it to the low end, the true value and the high end of the single-cycle range from the first example, that is 1.62, 2.0 and 2.42. Each configuration was trained for three epochs with ten seeds. The simulator is a NumPy implementation of the model in Equations (1) to (3), written for this chapter; it is a teaching tool and does not replace the frameworks in Table 1.

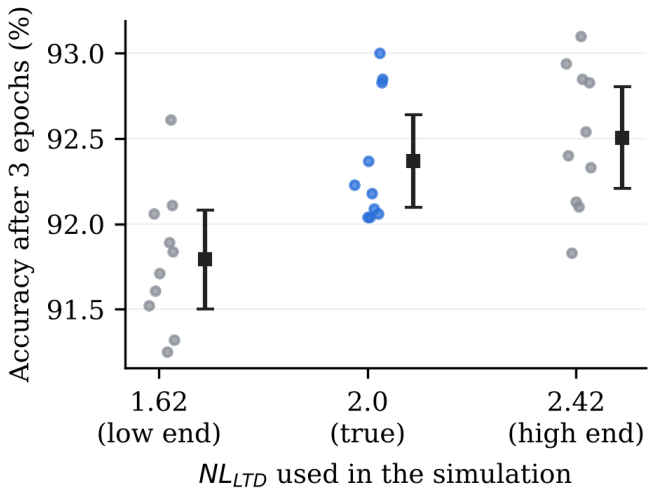

Figure 4: Test accuracy after three epochs for Device X simulated with the low end, the true value and the high end of the single-cycle range of NL_{LTD}

Dots are single seeds; squares and bars give the mean and 95 % confidence interval over ten seeds.

Figure 4 shows the outcome. Mean accuracy was 91.79 ± 0.41 % at NL_{LTD} = 1.62, 92.37 ± 0.38 % at 2.0 and 92.51 ± 0.42 % at 2.42 (mean ± standard deviation over ten seeds). Two things stand out. The uncertainty left by a single-cycle measurement shifts the predicted accuracy by 0.7 points,

which is more than the standard deviation between seeds. Also, in this setting the more curved depression branch did not give the lower accuracy. I did not investigate why. For the purposes of this chapter it is enough to note that the direction of a device effect has to be measured; it cannot be assumed.

How many seeds does it take to rank these configurations correctly? I drew n seeds at random from the ten available for each configuration, compared the two means, and counted how often their order matched the ten-seed order (Figure 5). For 1.62 against 2.0, whose means are 0.58 points apart, a single seed gave the right order 87 % of the time, two seeds 95 % and three seeds 99 %. For 2.0 against 2.42, only 0.14 points apart, one seed was right 64 % of the time and five seeds still only 78 %. These figures come from resampling just ten runs, so they are approximate. The conclusion does not hinge on that, however. When an effect is a fraction of the seed-to-seed spread, one run is not much better than a coin flip, and three runs are not enough either.

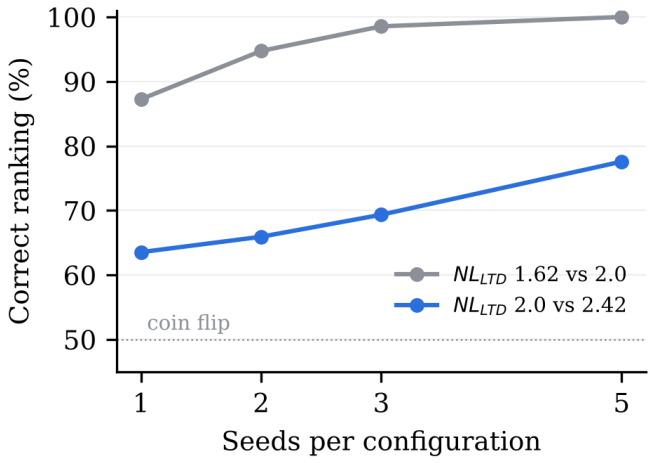

Figure 5: How often the mean of n seeds puts two configurations in the same order as the mean of all ten seeds, estimated by resampling
The grey pair differs by 0.58 points and the blue pair by 0.14 points.

Before the main runs, a quick check helps. With the numbers of Figure 4:

1. Train one configuration with five to ten seeds and compute the standard deviation s; here s ≈ 0.4 points.

2. For n planned seeds the standard error is s/√n; for n = 5 that is about 0.18 points.
3. Compare it with the smallest difference to be reported. The 0.14-point difference in Figure 5 is below one standard error at n = 5, consistent with five seeds ranking it correctly only 78 % of the time. Such a difference needs many more seeds, or should be reported as inconclusive.

Metrics

Accuracy in this chapter is always the test accuracy after the last epoch. The rules behind that choice:

- report test accuracy at an epoch fixed in advance, usually the last;
- do not pick the best test epoch afterwards, since that rewards lucky fluctuations, most of all in noisy configurations;
- if early stopping is used, base it on a validation split.

For cost metrics the definition matters more than the value. An energy figure should state:

- what it includes (array reads, array writes, peripheral circuits, digital logic);
- whether it refers to training or inference;
- the unit: per image, per epoch or per training run.

When a framework gives no energy estimate, a pulse count works, as long as it says whether reset and verify pulses are included and whether it is a total over the array or the count for the most-written cell.

A REPORTING CHECKLIST

Table 2 puts in one place every item that the two worked examples showed to matter, together with the usual ones on training and cost. Pineau et al. (2021) make the case for checklists of this kind in machine learning research.

Table 2: Reporting checklist for device-to-system simulation studies of memristive learning

Area	What to report
Measurement	Pulse amplitude, width and count; read voltage; number of cycles and devices; any discarded data
Device model	Normalisation (global or per cycle); model form; fitted parameters with intervals and how they were obtained; NL-to-parameter conversion
Variability	Cycle-to-cycle and device-to-device values, and what each percentage refers to
Simulator	Name, version or commit, modified files, what is and is not modelled; reproduction of the reference configuration
Network and training	Architecture, data set and preprocessing, weight mapping, update rule and rounding, learning rate and how it was chosen, epochs, batch size
Baselines and controls	Floating-point network, ideal device, one-factor controls for each claimed effect
Statistics	Number of seeds, what a seed controls, mean with standard deviation or confidence interval, epoch at which accuracy is reported
Cost metrics	Definition of energy, latency or pulse count, including what is counted and per what unit
Artefacts	Code, configuration files and logs, or where they can be obtained

Source: Prepared by the author.

COMMON PITFALLS

Most of the problems in studies of this kind come from a short list of mistakes.

- Fitting a single clean cycle and treating the fitted nonlinearity as exact. In the first worked example the same protocol returned values about 20 % apart.
- Taking a nonlinearity label from a device paper and passing it to a simulator that converts labels into curvature differently.
- Normalising each cycle separately, so that the drift of the conductance window drops out of the model.
- Comparing a tuned proposal with an untuned baseline, or changing several factors in one step.
- Reporting a single seed, or the best epoch on the test set.

- Quoting energy without saying what it covers, or setting energy numbers from different frameworks side by side as if they measured the same thing.
- Changing simulator source code without listing the changes.

CONCLUSION

A simulated accuracy for a memristive device sits at the end of a chain that starts at the probe station, and this chapter has followed that chain one link at a time. On a synthetic device, the two worked examples put numbers on uncertainties that papers usually leave out. One measured cycle left the depression nonlinearity uncertain over a band of 0.8 in NL, and that band on its own moved simulated accuracy by 0.7 points. When two configurations differed by 0.14 points, five seeds still ranked them the wrong way round about one time in five. These numbers are small, but they are of the same size as many effects reported in the literature. Measuring a few more cycles, running a few more seeds and reporting the items in Table 2 costs little compared with fabricating a device, and it is what allows a reader to trust the comparison that comes next.

REFERENCES

Bouthillier, X., Delaunay, P., Bronzi, M., Trofimov, A., Nichyporuk, B., Szeto, J., Mohammadi Sepahvand, N., Raff, E., Madan, K., Voleti, V., Ebrahimi Kahou, S., Michalski, V., Arbel, T., Pal, C., Varoquaux, G., and Vincent, P. (2021). Accounting for variance in machine learning benchmarks. *Proceedings of Machine Learning and Systems, 3*, 747–769. https://doi.org/10.48550/arXiv.2103.03098

Chen, P.-Y., Peng, X., and Yu, S. (2018). NeuroSim: A circuit-level macro model for benchmarking neuro-inspired architectures in online learning. *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems, 37*(12), 3067–3080. https://doi.org/10.1109/TCAD.2018.2789723

Efron, B., and Tibshirani, R. J. (1993). *An introduction to the bootstrap*. New York: Chapman and Hall. https://doi.org/10.1201/9780429246593

Feinberg, B., Xiao, T. P., Brinker, C. J., Bennett, C. H., Marinella, M. J., and Agarwal, S. (2026). *CrossSim: Accuracy simulation of analog in-memory computing* (Version 3.2.1) [Computer software]. Sandia National Laboratories. https://github.com/sandialabs/cross-sim

Fusi, S., and Abbott, L. F. (2007). Limits on the memory storage capacity of bounded synapses. *Nature Neuroscience, 10*(4), 485–493. https://doi.org/10.1038/nn1859

Gokmen, T., and Haensch, W. (2020). Algorithm for training neural networks on resistive device arrays. *Frontiers in Neuroscience, 14*, 103. https://doi.org/10.3389/fnins.2020.00103

Gokmen, T., and Vlasov, Y. (2016). Acceleration of deep neural network training with resistive cross-point devices: Design considerations. *Frontiers in Neuroscience, 10*, 333. https://doi.org/10.3389/fnins.2016.00333

Gupta, S., Agrawal, A., Gopalakrishnan, K., and Narayanan, P. (2015). Deep learning with limited numerical precision. In *Proceedings of the 32nd International Conference on Machine Learning* (pp. 1737–1746). PMLR. https://doi.org/10.48550/arXiv.1502.02551

Henderson, P., Islam, R., Bachman, P., Pineau, J., Precup, D., and Meger, D. (2018). Deep reinforcement learning that matters. In *Proceedings of the AAAI Conference on Artificial Intelligence, 32*(1). https://doi.org/10.1609/aaai.v32i1.11694

LeCun, Y., Bottou, L., Bengio, Y., and 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

Le Gallo, M., Lammie, C., Büchel, J., Carta, F., Fagbohungbe, O., Mackin, C., Tsai, H., Narayanan, V., Sebastian, A., El Maghraoui, K., and Rasch, M. J. (2023). Using the IBM analog in-memory hardware acceleration kit for neural network training and inference. *APL Machine Learning, 1*(4), 041102. https://doi.org/10.1063/5.0168089

Lu, A., Peng, X., Li, W., Jiang, H., and Yu, S. (2021). NeuroSim simulator for compute-in-memory hardware accelerator: Validation and benchmark. *Frontiers in Artificial Intelligence, 4*, 659060. https://doi.org/10.3389/frai.2021.659060

Luo, Y., Peng, X., and Yu, S. (2019). MLP+NeuroSimV3.0: Improving on-chip learning performance with device to algorithm optimizations. In *Proceedings of the International Conference on Neuromorphic Systems* (pp. 1–7). ACM. https://doi.org/10.1145/3354265.3354266

Peng, X., Huang, S., Jiang, H., Lu, A., and Yu, S. (2021). DNN+NeuroSim V2.0: An end-to-end benchmarking framework for compute-in-memory accelerators for on-chip training. *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems, 40*(11), 2306–2319. https://doi.org/10.1109/TCAD.2020.3043731

Pineau, J., Vincent-Lamarre, P., Sinha, K., Larivière, V., Beygelzimer, A., d'Alché-Buc, F., Fox, E., and Larochelle, H. (2021). Improving reproducibility in machine learning research. *Journal of Machine Learning Research, 22*(164), 1–20. https://doi.org/10.48550/arXiv.2003.12206

Rasch, M. J., Moreda, D., Gokmen, T., Le Gallo, M., Carta, F., Goldberg, C., El Maghraoui, K., Sebastian, A., and Narayanan, V. (2021). A flexible and fast PyTorch toolkit for simulating training and inference on analog crossbar arrays. In *2021 IEEE 3rd International Conference on Artificial Intelligence Circuits and Systems (AICAS)* (pp. 1–4). IEEE. https://doi.org/10.1109/AICAS51828.2021.9458494

Read, J., Lee, M.-Y., Huang, W.-H., Luo, Y.-C., Lu, A., and Yu, S. (2025). *NeuroSim V1.5: Improved software backbone for benchmarking compute-in-memory accelerators with device and circuit-level non-idealities* (arXiv:2505.02314). arXiv. https://doi.org/10.48550/arXiv.2505.02314

Sebastian, A., Le Gallo, M., Khaddam-Aljameh, R., and Eleftheriou, E. (2020). Memory devices and applications for in-memory computing. *Nature Nanotechnology, 15*(7), 529–544. https://doi.org/10.1038/s41565-020-0655-z

Yu, S. (2018). Neuro-inspired computing with emerging nonvolatile memorys. *Proceedings of the IEEE, 106*(2), 260–285. https://doi.org/10.1109/JPROC.2018.2790840

Evidential Concept Bottlenecks for Colposcopic Swede Scoring: A Data-Audited Case Study on the BUET Multi-Center Colposcopy Dataset

Halil İbrahim COŞAR[1]
Esra GÜNGÖR ULUTAŞ[2]

1- Asst. Prof. Dr.; Yozgat Bozok University, Faculty of Engineering-Architecture, Department of Electrical and Electronics Engineering. halil.cosar@bozok.edu.tr ORCID No: 0000-0001-8064-2385
2- Lect.; Yozgat Bozok University, Sorgun Vocational School, Department of Computer Technologies. esra.ulutas@bozok.edu.tr ORCID No: 0000-0001-8084-6644

ABSTRACT

Deep learning applications in colposcopy are typically presented as a singular accuracy metric for a black-box classifier. This chapter takes a different path on a recent public, multi-centre colposcopy dataset from Bangladesh (768 patients, four images per patient under saline, green filter, acetic acid and Lugol's iodine). We audit the data prior to modelling and find three issues that change how the dataset should be used: the modality order assumed by the repository's own baseline notebooks differs from the dataset documentation, which we verify on the images; a routine download path silently delivers placeholder files instead of images; and, most importantly, the released CIN grade is a deterministic function of the total Swede score in all 768 cases, i.e. a provisional colposcopic diagnosis rather than an independent reference standard. Thus, predicting CIN from these images is equivalent to estimating the five Swede components. We then build an Evidential Concept Bottleneck Model (E-CBM) which is a DINOv2 backbone adapted with LoRA, attention fusion across the four views, five evidential ordinal heads for the Swede components, and a CIN head that only sees those five concepts. We document the training failures we witnessed – most notably a silent "evidence collapse" not betrayed by a reasonable-looking loss curve – and the monitoring that uncovered them. The E-CBM reaches a quadratic weighted kappa of 0.72 for lesion size and of 0.45-0.47 for three of the remaining four components, of 0.61 for the total Swede score, and of an AUC of 0.79 for CIN2+ on a fixed held-out test set of 154 patients. In a head to head comparison with an otherwise identical black box model, the concept bottleneck does not incur any detectable cost at the treatment threshold (CIN2+ AUC 0.792 vs 0.793; difference +0.001, 95% CI –0.034 to +0.038) but does lose some ability to recognise perfectly normal cervices (screening AUC –0.04) and produces fewer sharp probabilities, even after temperature scaling. We explain the mechanism behind this cost and conclude with concrete recommendations for users of this dataset and for evidential models in general.

Keywords – colposcopy; cervical cancer screening; Swede score; concept bottleneck models; evidential deep learning; uncertainty; calibration; data audit; DINOv2; LoRA

I. INTRODUCTION

Cervical cancer remains one of the most preventable cancers, and screening followed by treatment of high-grade precancerous lesions is central to the World Health Organization's elimination strategy (World Health Organization, 2020). Colposcopy magnified visual inspection of the cervix after the sequential application of saline, acetic acid and Lugol's iodine, often

complemented by a green filter that accentuates vascular patterns is the step at which the clinician decides whether a lesion warrants biopsy or treatment (Bornstein et al., 2012). The examination is strongly skill-dependent. Studies of digitized images from the ALTS trial found only modest agreement between expert colposcopists on the overall colposcopic impression (Ferris & Litaker, 2005; Jeronimo et al., 2007). Agreement on individual grading components such as margins, vessels and acetowhitening was similarly limited (Massad et al., 2008). A high-grade colposcopic impression also detected high-grade cervical intraepithelial neoplasia with limited sensitivity (Massad et al., 2009). This variability, compounded by the shortage of trained colposcopists in low- and middle-income countries, motivates most work on automated colposcopy: a model that reads images consistently could support less experienced examiners and extend screening to settings with few specialists (Xue, Ng, & Qiao, 2020). A dataset used here ships with two baseline notebooks: a PyTorch model that fuses image pairs with an attention mechanism on EfficientNet features to predict CIN, and a TensorFlow model that predicts the Swede components (BUET mHealth Lab, 2026). The typical study in this area trains a convolutional or transformer classifier end to end and reports its accuracy on a held-out split (Hu et al., 2019; Cho et al., 2020; Xue et al., 2020; Yuan et al., 2020).

Evidential deep learning (Sensoy et al., 2018) replaces the softmax output of a classifier with the parameters of a Dirichlet distribution over class probabilities. The network outputs non-negative "evidence" for each class; low total evidence means that the model has seen little support for any class and should be uncertain. This yields an epistemic uncertainty estimate from a single forward pass. The training objective combines an expected-risk term with a Kullback–Leibler (KL) penalty that pushes evidence for incorrect classes towards zero, and the penalty is usually annealed — increased gradually during early training.

Two properties of this framework matter later in the chapter. First, the KL term rewards a model that expresses no evidence at all when it cannot fit a task; if nothing counteracts this, a head can settle into an uninformative state. Second, a Dirichlet prior with unit concentration keeps predicted probabilities away from 0 and 1, which affects calibration. Calibration whether predicted probabilities match observed frequencies is commonly summarised by the expected calibration error (ECE) (Naeini et al., 2015; Guo et al., 2017). ECE, however, only measures calibration and not sharpness: a model that always predicts the class frequencies is perfectly calibrated and useless. Strictly

proper scoring rules such as the Brier score (Brier, 1950) and the logarithmic loss reward calibration and sharpness together (Gneiting & Raftery, 2007), and we report them alongside ECE throughout. Post-hoc temperature scaling (Guo et al., 2017) rescales predicted probabilities with a single parameter fitted on held-out data; it improves calibration without changing which class is predicted.

This chapter works through several problems on one concrete, publicly available dataset — the BUET Multi-Center Colposcopy Dataset released in 2026 by the mHealth Lab at the Bangladesh University of Engineering and Technology (BUET) — and it does so in the order in which they should be tackled in practice: data first, model second, claims last. Our contributions are:

- **A pre-modelling data audit** that finds, among smaller issues, that the released CIN grade is a deterministic function of the total Swede score (768 of 768 patients), and that the modality order assumed by the dataset's own baseline notebooks contradicts its documentation. Together this change what the dataset can and cannot be used to claim.
- **An Evidential Concept Bottleneck Model** that predicts the five Swede components as ordinal, uncertainty-aware "concepts" and derives the CIN grade only from them, so that every grade can be traced back to clinically meaningful scores.
- **An honest account of training failure modes**, including an evidence collapse that went unnoticed while the loss looked reasonable, and a monitoring protocol that would have caught it immediately.
- **A controlled, paired comparison** against an otherwise identical black-box model on the same 154 test patients, which locates the cost of interpretability precisely and explains its mechanism.

II. MATERIALS AND METHODS

A. Dataset

The BUET Multi-Center Colposcopy Dataset (repository name Colposcence) was collected at several clinical sites in Dhaka, Bangladesh, under institutional review board approval, with informed consent and anonymisation (BUET mHealth Lab, 2026). It is released under a Creative Commons Attribution 4.0 licence. Each patient folder contains four images,

numbered by the application used (Table 1), and a spreadsheet provides, per patient, the TZ type, the five Swede components, the total Swede score and a CIN grade.

Table 1. Composition of the dataset as released and as used in this chapter. Component distributions are identical to those published in the dataset documentation.

Item	Value
Patients (folders / spreadsheet rows)	768 / 768
Images per patient (documented order)	001 saline, 002 green filter, 003 acetic acid, 004 Lugol's iodine
Image files found	3,074 (762 patients × 4, 4 patients × 5, 2 patients × 3)
Patients used (all four modalities present)	766
Aceto uptake, scores 0 / 1 / 2	293 / 409 / 66
Margin and surface, 0 / 1 / 2	441 / 312 / 15
Vessels, 0 / 1 / 2	635 / 84 / 49
Lesion size, 0 / 1 / 2	336 / 357 / 75
Iodine staining, 0 / 1 / 2	207 / 439 / 122
CIN grade: Normal / CIN1 / CIN2 / CIN3	186 / 410 / 108 / 64
Transformation zone: type 1 / type 2 / type 3	718 / 50 / 0

B. Audit findings

Before writing any model code we looked at the files, the spreadsheet and the images themselves. Easy things to check – file sizes, cross tabs, colour stats, looking at images but several changed the study.

Placeholder files instead of images: The images are stored using Git Large File Storage. Download the repository through the web interface's "Download ZIP" option and you get 3,074 files, 131 bytes each: text pointers, not JPEGs. At this point, nothing fails loudly; a data loader that doesn't check file sizes will either crash later, or worse, silently train on errors. When cloning with Git LFS (or downloading from the LFS media endpoint) the real images are fetched, as we verified by checking that no file smaller than 500 bytes was left.

Order of modality. Dataset documentation shows the order of saline (001), green filter (002), acetic acid (003), and Lugol's iodine (004). However, the baseline notebooks in the repository read file 001 as the acetic-acid image, 002 as the iodine image, and 003 as the "vascular" (green-filter) image. We solved the conflict on the images: for a sample of 100 patients, position 002

was strongly green-dominant, position 003 was the brightest and least saturated (acetowhite), position 004 was the brown-mustard palette typical of iodine, and position 001 was unstained mucosa; visual inspection of individual patients agreed. This means that the documentation is correct and a model built from the notebooks as written would feed each branch a different modality than intended. We follow here the documented order.

Minor discrepancies: The documentation says 3,072 images, but the folders contain 3,074 files: four patients have a fifth image and two patients have only three. We exclude the two incomplete patients, thus 766 remain. One TZ entry says "type type 1" which we normalise to type 1. There are no patients with a type 3 transformation zone, so it is impossible to analyse type 3 cases, a natural subgroup of interest, on this dataset. The type 2 subgroup is small (50 patients; 9 in our test set). The CIN grade was determined by the total Swede score. The two label columns are cross-tabulated producing a perfect partition (Figure 1, Table 2):

All patients with total Swede score 0 are labelled Normal, all patients with total Swede score 1-4 are labelled CIN1, all patients with total Swede score 5-6 are labelled CIN2, all patients with total Swede score 7-10 are labelled CIN3; 768 patients without a single exception. The documentation hints at this, when it describes the total score as one which can be mapped on to a provisional diagnosis. Histopathology is not referred to. Whatever the clinical workflow behind the spreadsheet, the released CIN column is a thresholded total Swede score.

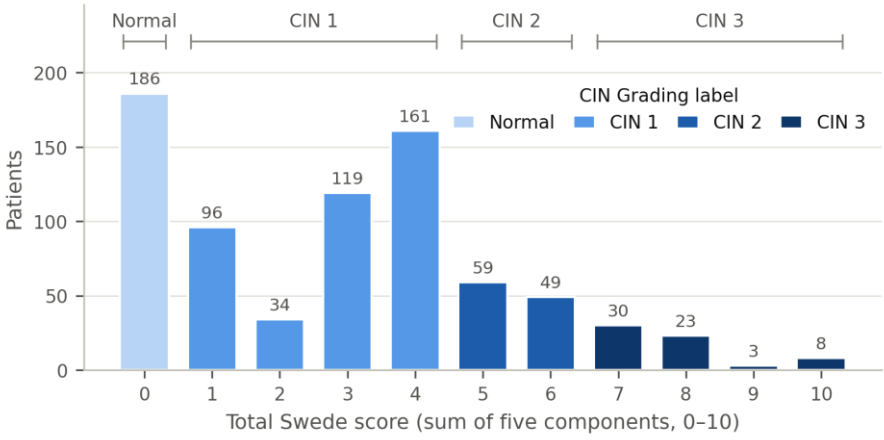

Figure 1. Number of patients at each total Swede score, coloured by the released CIN grade. Each score maps to exactly one grade (bands above the bars); no score is shared by two grades. Data: all 768 patients.

Table 2. The label-generating rule implied by the data. The rule holds for all 768 patients; in the 3-class scheme used for modelling, CIN2 and CIN3 are merged into "high grade".

Total Swede score	Released CIN grade	Patients	3-class label
0	Normal	186	Normal
1–4	CIN1	410	CIN1
5–6	CIN2	108	High grade
7–10	CIN3	64	High grade

To quantify what this means for learning we trained a simple multinomial logistic regression taking as input the clinician's own five Swede components and predicting the 3-class CIN label. On our held-out test patients it achieves a quadratic weighted kappa of 0.986 and an AUC of 0.999 for CIN2+; the true total score alone separates CIN2+ perfectly (AUC 1.000), and thresholding it reproduces the test labels exactly.

C. Consequences for the task

The audit changes the scientific question. Because the CIN label is a function of the Swede components, all of the information needed to predict it is contained in those components, and the only thing an image model can contribute is to estimate the components from the images. Three consequences follow.

- **CIN results on this dataset are not histological results.** They should be reported as agreement with a provisional colposcopic grading derived from the Swede score. Any study that reports "CIN classification" on this dataset — including the repository's own baseline is, in effect, reporting Swede-score estimation.
- **The Swede components are the primary endpoints.** Agreement on each component and on the total score are the clinically meaningful image-level measures.
- **A concept bottleneck is the natural architecture.** The label-generating process is "concepts, then diagnosis", so a CBM does not impose an artificial constraint here; it mirrors how the labels were made. It also means that a black-box model cannot, in principle, exploit

information that the concepts lack, which makes the comparison between the two unusually clean.

D. Methods

Figure 2 summarises the model. The four images of a patient pass through a shared, parameter-efficiently adapted vision transformer; a learned attention module fuses the four view embeddings into one patient embedding; five evidential ordinal heads predict the Swede components; and the CIN grade is obtained from the five predicted components only. We implement three ways of doing the last step, which the experiments compare. A black-box variant, used as a control, differs in a single respect: its CIN head reads the fused image embedding instead of the concepts.

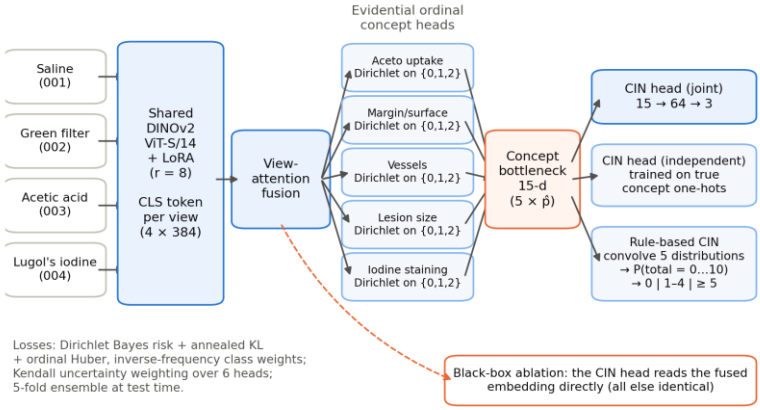

Figure 2. The Evidential Concept Bottleneck Model (E-CBM). Four views per patient share one DINOv2 backbone with LoRA adapters.

We use the small DINOv2 vision transformer (ViT-S/14; Dosovitskiy et al., 2021; Oquab et al., 2023), pre-trained by self-distillation on a large curated natural-image corpus, at 224 × 224 input resolution. Self-supervised features of this kind transfer well to small labelled datasets, and the small variant keeps four images per patient within the memory of a single 20 GB GPU. Rather than fine-tune all weights on roughly 490 training patients per fold, we insert low-rank adapters (LoRA; Hu et al., 2022) of rank 8 (scaling $\alpha = 16$, dropout 0.05) into the query and value projections of every attention layer and train only these, together with the heads. The backbone is frozen for the first five epochs so that the freshly initialised heads can settle before the adapters start to move.

Each image yields the embedding of the classification token, $v_m \in \mathbb{R}^{384}$ for view $m \in \{\text{saline, green, acetic, iodine}\}$. A light attention module fuses the four: a learned query vector q scores linearly projected keys, the scores are normalised with a softmax over views, and the patient embedding is the weighted sum of the view embeddings. This is cheaper than a mixture-of-experts design and returns, for every patient, a weight per modality that can be inspected.

Each Swede component c is predicted by a two-layer perceptron (384 → 128 → 3, ReLU, dropout 0.5) whose three outputs z are turned into Dirichlet parameters following Sensoy et al. (2018):

$$e = softplus(z), \quad \alpha = e + 1, \quad S = \sum_k \alpha_k, \quad \hat{p}_k = \frac{\alpha_k}{S}, \quad u = \frac{K}{S} \tag{1}$$

where K = 3 classes, $\hat{p}$ is the expected class probability and $u \in (0, 1]$ is the epistemic uncertainty: u = 1 when the head emits no evidence. For a sample with one-hot label y the loss has three parts. The Bayes risk of the cross-entropy under the Dirichlet,

$$L_{BR} = \sum_k y_k \left[\psi(S) - \psi(\alpha_k) \right] \tag{2}$$

with ψ the digamma function; a KL penalty that shrinks evidence for the wrong classes towards the uniform Dirichlet (α' sets the target-class entry of α to 1 and leaves the others unchanged),

$$L_{KL} = \lambda_t \, KL(Dir(\alpha') \parallel Dir(1)), \alpha' = y + (1 - y) \odot \alpha, \lambda_t = \lambda \, min(1, \frac{t}{T_a}) \tag{3}$$

annealed linearly over $T_a = 15$ epochs to a final weight $\lambda = 0.3$; and, because the Swede scores are ordinal, a term that penalises the distance between the expected score and the true score,

$$L_{ord} = Huber_\delta(\sum_k k \, \hat{p}_k - y), \quad \delta = 1 \tag{4}$$

Standard evidential learning is categorical and treats a 0-versus-2 error like a 0-versus-1 error; the ordinal term restores the ordering. It is a simple combination of two established ingredients rather than a method taken from a specific prior paper. The per-sample loss of head c is

$$L_c = w(y) \, (L_{BR} + L_{KL} + \beta \, L_{ord}), \quad \beta = 0.5 \tag{5}$$

where w(y) is the class weight of the true class.

E. Three ways from concepts to CIN

Joint CIN head. The bottleneck $\mathbf{b} \in \mathbb{R}^{15}$ concatenates the expected probability vectors $\hat{p}$ of the five components. A small perceptron ($15 \to 64 \to 3$, dropout 0.3) with an evidential output predicts the 3-class grade (Normal, CIN1, high grade = CIN2 or CIN3) from **b** alone. It is trained together with everything else, and its gradients flow back into the concept heads — the joint regime of Koh et al. (2020).

Independent CIN head: An identical perceptron receives, during training, the *ground-truth* concepts as one-hot vectors; at validation and test time it receives the same predicted bottleneck as the joint head. Its input during training carries no gradient to the rest of the network, so it cannot influence the concept heads, and it is added to the total loss with a fixed weight of 1, outside the multi-task weighting. It learns the concept-to-grade mapping from clean inputs immediately instead of waiting for the concept heads to become informative.

Rule-based CIN: Given the audit, the grade can also be computed without any learned parameters. Treating the five predicted component distributions as conditionally independent given the images, the distribution of the total score $T \in \{0, \ldots, 10\}$ is their convolution, and the dataset's own rule converts it into grade probabilities:

$$P(Normal) = P(T = 0), P(CIN1) = P(1 \leq T \leq 4), P(high) = P(T \geq 5) \quad (6)$$

The same distribution provides an estimate of the total Swede score (its expectation) and a point prediction (its mode). The rule is taken from the label definition, not fitted to any data.

The five concept heads and the joint CIN head are combined with homoscedastic uncertainty weighting (Kendall et al., 2018): each head h has a learnable log-variance s_h, and the total loss is $\Sigma_h [\exp(-s_h) L_h + \tfrac{1}{2} s_h]$. This balances heads with very different difficulty without an extra backward pass. Several components are strongly imbalanced 83% of patients have a vessel score of 0, and only 15 of 768 have a margin score of 2, which makes "never predict the rare class" an attractive local minimum. We therefore weight each sample by the inverse frequency of its true class, computed on the training fold, with the balanced weight $n / (K \cdot n_k)$ capped at 5 and the weights renormalised to mean 1. For the vessel head this gives weights of about 0.15,

1.06 and 1.79 for scores 0, 1 and 2; for the 3-class CIN label, 1.18, 0.53 and 1.28, so the majority class CIN1 is down-weighted.

III. RESULTS

All splits are at patient level and each training sample is one patient with all four images, so that the information from one patient can never appear on both sides of a split. We first hold out a fixed test set of 20% of patients (154) stratified according to the 3-class CIN label (38 Normal, 82 CIN1, 34 high grade; 145 type 1 and 9 type 2 transformation zones). The remaining 612 patients are randomly split into five folds by iterative multi-label stratification (Sechidis et al., 2011) over five one-hot encoded components and the CIN label, so that even rare classes are represented in each fold; the distribution of development and test are nearly identical. All random operations use a fixed seed (42) so that the splits can be reconstructed exactly. We verified this by recomputing from the saved checkpoints the validation kappa logged at training time for all ten folds of both final runs (all matched to four decimals).

Table 3 lists the configuration of the final runs. The values are the outcome of the diagnostic process, not of a hyperparameter search on the test set. Models are optimised with AdamW (Loshchilov & Hutter, 2019) and implemented in PyTorch (Paszke et al., 2019) with the Hugging Face Transformers (Wolf et al., 2020) and PEFT (Mangrulkar et al., 2022) libraries; splits and metrics use scikit-learn (Pedregosa et al., 2011) and the iterative-stratification package.

Table 3. Configuration of the final E-CBM and black-box runs.

Component	Setting
Backbone	DINOv2 ViT-S/14, 224 × 224 input, classification-token embedding (384-d)
Adaptation	LoRA on query/value projections, rank 8, $\alpha = 16$, dropout 0.05; backbone frozen for the first 5 epochs
Heads	Concepts: 384 → 128 → 3, dropout 0.5; CIN on bottleneck: 15 → 64 → 3, dropout 0.3
Loss per head	Dirichlet Bayes risk + KL ($\lambda = 0.3$, linear annealing over 15 epochs) + 0.5 × ordinal Huber ($\delta = 1$); inverse-frequency class weights (cap 5)
Multi-task weighting	Homoscedastic uncertainty weighting over 6 heads; independent CIN head added with weight 1

Component	Setting
Optimisation	AdamW; learning rate 3×10^{-4} (heads) and 1×10^{-4} (LoRA); weight decay 2×10^{-4}; gradient-norm clipping at 1.0; mixed precision
Batch	8 patients (32 images) per step, gradient accumulation over 2 steps
Augmentation	Horizontal and vertical flips, rotation ±15°, brightness/contrast jitter 0.05 (kept small to preserve colour cues)
Schedule and selection	At most 40 epochs; per fold, the epoch with the highest mean validation QWK over 7 heads; patience 10
Test-time prediction	Mean of the five fold models' expected Dirichlet probabilities
Compute	One NVIDIA RTX 4000 Ada (20 GB); about 2.5 h per five-fold run

To separate calibration from information content, we apply temperature scaling (Guo et al., 2017) to each head of both models. The temperature T is fitted by minimising the logarithmic loss on out-of-fold validation predictions — each of the 612 development patients predicted by the one-fold model that did not train on it and then applied to the test-set ensemble as p_T ∝ p^(1/T). No test data enter the fit. Temperature scaling never changes the predicted class, so kappa, accuracy and confusion matrices are unaffected; only the confidence of the probability's changes. Fitting on single-model predictions and applying to an ensemble is an approximation, which we accept for its simplicity.

Logging kappa immediately revealed a serious problem (Figure 3). In the third run, the validation kappa of the CIN head was precisely 0.000 in 45 of the 46 logged epochs across the five folds and that of the vessel head in all 46. If the kappa is zero exactly then the prediction is constant. The loss told the same story to anyone who knew what to look for: the validation loss of the CIN head climbed to 1.508–1.527, close to $\psi(3) - \psi(1) = 1.5$, the Bayes-risk term of a Dirichlet with no evidence at all ($\alpha = 1$ for every class). Such a head returns $\hat{p} = (⅓, ⅓, ⅓)$ for each patient, arg-max ties and returns the first class, and all patients get the same label. But the loss itself didn't seem to be anything special, and the dead heads went unnoticed for two full runs. Even worse, the new selection criteria, the kappa of a dead head, was stuck at zero, so patience never reset and every fold stopped as soon as the eight-epoch patience ran out (at epoch 8, or 9 in one-fold) while the other heads were still improving.

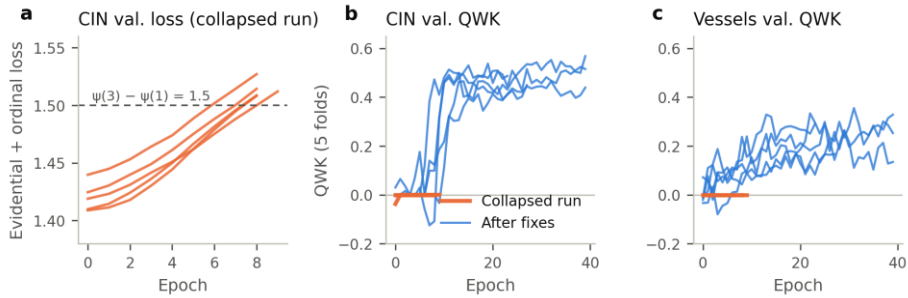

Figure 3. Evidence collapse and its repair. (a) Validation loss of the CIN head in the collapsed run (five folds) rises towards 1.5, the Bayes-risk value of an evidence-free Dirichlet. (b, c) Validation QWK of the CIN and vessel heads: exactly zero throughout in the collapsed run (orange; every fold stopped at epoch 8 or 9), and rising once class weighting, a weaker KL penalty and a robust selection criterion were introduced (blue).

For separate reasons, both heads fell. Vessel head opposed an 83% majority class where emitting no evidence is a comfortable minimum and the KL term actively favours it. The CIN head faces a cold-start issue with idea bottlenecks, as it initially only receives the concept vector, which is close to (⅓, ⅓, ⅓, …) for each patient. Head collapses before inputs are informative and has little gradient to recover from (nothing to learn from).

Inverse-frequency class weights, a lower final KL weight (0.3 instead of 1.0) with a slower ramp (15 instead of 10 epochs), the independent CIN head, which removes the cold start for the concept-to-grade mapping, model selection on the mean kappa over all heads, allowing a dead head to lower the criterion but not freeze it, ten epochs of patience, and logging of the mean epistemic uncertainty u of each head. In this run, all heads showed evidence (mean validation u: 0.28–0.55 at selected epochs), training for 24-40 epochs, and mean validation kappa: 0.484 ± 0.019 across folds. It cannot rescue the joint head because the independent head's gradients are decoupled from the rest of the network. The rescue is attributable to the class weighting, lesser KL penalty, and longer training, which we altered simultaneously and cannot disentangle.

Estimating the Swede components

Table 4 summarises the performance of the concept heads of the E-CBM. Agreement was highest for lesion size (QWK 0.72) and moderate for

aceto uptake, iodine staining and margins (0.45-0.47). The vessel component is just slightly above chance (0.16, lower confidence limit 0.02). The test-set values generally agree with the cross-validation estimates on the development folds (all within the test confidence intervals), so there is no sign of a generalisation gap. The uncertainty of each head is positively correlated with the errors of the head (r = 0.21–0.41): on average when the model is wrong, it is less certain. Supplementary Figure 4 shows confusion matrices of all components of the two models.

Table 4. Swede-component estimation by the E-CBM on the test set (n = 154). Validation QWK is the mean ± SD across the five development folds at the selected epochs. Counts of the rarest score in the test set are given because they bound what can be measured.

Component	Test QWK [95% CI]	Validation QWK	Acc.	Macro-F1	AUC	ECE	r(u, error)	Rarest score in test
Lesion size	0.724 [0.633, 0.807]	0.651 ± 0.035	0.773	0.705	0.888	0.111	0.21	score 2: 11
Aceto uptake	0.472 [0.335, 0.604]	0.502 ± 0.084	0.630	0.556	0.823	0.127	0.30	score 2: 13
Iodine staining	0.468 [0.348, 0.585]	0.521 ± 0.051	0.552	0.521	0.769	0.050	0.21	score 2: 26
Margin/surface	0.451 [0.304, 0.595]	0.484 ± 0.046	0.721	0.643	0.849	0.086	0.30	score 2: 3
Vessels	0.162 [0.020, 0.293]	0.271 ± 0.061	0.656	0.396	0.711	0.100	0.41	score 2: 8

That ranking makes sense. Lesion size is a coarse, spatially extended feature that survives down-sampling to 224 x 224 pixels, while vessel patterns, i.e. punctation, mosaicism, atypical vessels, are fine structures that are likely to be degraded at that resolution; the vessel component is also the most imbalanced (130 of 154 test patients score 0). The margin component is not a modelling limit, but a measurement limit: only three of the test patients have a margin score of 2, so the kappa for that class is based on a handful of cases.

Summing the component distributions gives an estimate of the total score. Its mean absolute error is 1.65 points on the 0–10 scale; the mode of the

predicted distribution agrees with the true total with a QWK of 0.608 [0.512, 0.688] and lies within ±1 point for 57% of patients (Spearman ρ = 0.655). The total is estimated better than any single component except lesion size, because independent errors in the components partly cancel when they are added.

Table 5 compares the three ways of turning concepts into a grade. The two learned heads behave similarly: QWK 0.47 and 0.44, an AUC of 0.79 for the treatment question and 0.81 for the screening question. They differ mainly in their operating point: with arg-max decisions the independent head is more sensitive for CIN2+ (0.82 versus 0.68) at the cost of specificity (0.60 versus 0.72). The rule-based variant, although it reproduces the label definition exactly, is clearly worse in agreement (QWK 0.33) and calibration (ECE 0.21), and it never predicts "Normal": its screening sensitivity is 1.00 and its specificity 0.00. Yet its AUCs are similar as those of the learned heads. All three variants therefore rank patients equally well; they differ in how they convert the ranking into probabilities and decisions.

Table 5. CIN grading by the three E-CBM variants on the test set (38 Normal, 82 CIN1, 34 high grade). Sensitivity and specificity use arg-max decisions. Brier score and log loss are shown before temperature scaling; a model that always predicts the class frequencies scores 0.607 and 1.014.

Metric	Joint head	Independent head	Rule-based
QWK [95% CI]	0.471 [0.352, 0.584]	0.435 [0.308, 0.556]	0.328 [0.221, 0.427]
Accuracy / macro-F1	0.455 / 0.458	0.448 / 0.447	0.506 / 0.378
AUC, CIN2+ (treatment) [95% CI]	0.792 [0.707, 0.859]	0.790 [0.702, 0.857]	0.783 [0.698, 0.853]
Sensitivity / specificity, CIN2+	0.676 / 0.717	0.824 / 0.600	0.706 / 0.733
AUC, any CIN (screening) [95% CI]	0.806 [0.727, 0.883]	0.809 [0.732, 0.883]	0.809 [0.732, 0.883]
Sensitivity / specificity, any CIN	0.664 / 0.816	0.724 / 0.737	1.000 / 0.000
ECE	0.023	0.036	0.211
Brier score / log loss	0.611 / 1.010	0.620 / 1.024	0.660 / 1.195

The rule's failure is illuminating. For the 38 truly normal test patients, the average probability assigned to score 0 by each component head was 0.60–0.71, and for 26 of the test patients, the most probable score was 0 for each of the five components. The concept heads “saw” that these cervices were

normal. But “Normal” means all five components must be 0, and the probability of a conjunction is the product of the individual probabilities: about 0.19 on average for these patients, and never more than 0.43 for any test patient, always below the probability mass that the convolution assigns to totals of 1–4. This is supported by the Dirichlet prior, which prevents any component probability from reaching 1: normal patients have an expected total overestimated by 2.5 points, high-grade patients have an expected total underestimated by 1.0 point, a regression toward the middle of the scale. This bias cannot be corrected by the rule. The learned heads can, because they are trained on the grades. As an exploratory check – chosen after seeing these results and so not a headline result – applying the same rule to the most probable score of each component instead of to their distributions gives a QWK of 0.487 (vs 0.328) and correctly identifies 26 of the 38 normal patients (vs none), which confirms that the information is there and it is the probabilistic aggregation that loses it. However, the joint head has problems with CIN1 (Figure 4): 36 of 82 CIN1 patients are called Normal and 30 high grade. CIN1 has a narrow range of totals (1-4), hence an error in any one component can push the estimated total outside of the range in either direction, and the class weight of CIN1 (0.53) makes this middle class the cheapest to misclassify during training. Normal and high-grade patients are identified much more frequently (82%, 68%).

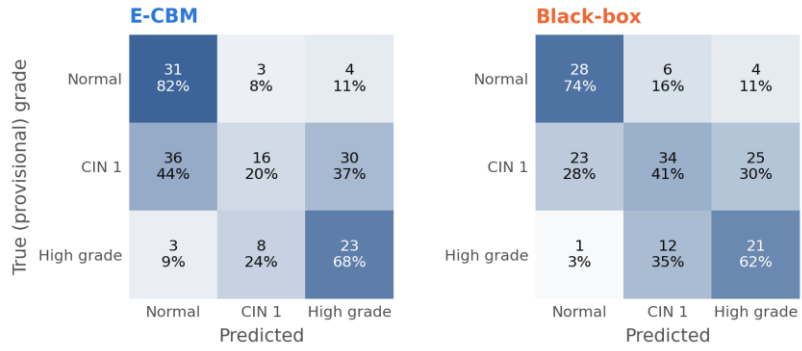

Figure 4. Confusion matrices of the joint CIN head of the E-CBM and of the black-box model on the test set (counts and row percentages). Temperature scaling does not change these matrices.

Figure 5 shows the test-set agreement of both models for every component, the total score and the grade. Differences on the components are small and inconsistent in direction; the only nominal difference favours the E-CBM for lesion size (0.724 versus 0.662; paired difference −0.062 [−0.132,

−0.005], p = 0.032). The total Swede score is estimated equally well (0.608 versus 0.623; p = 0.45).

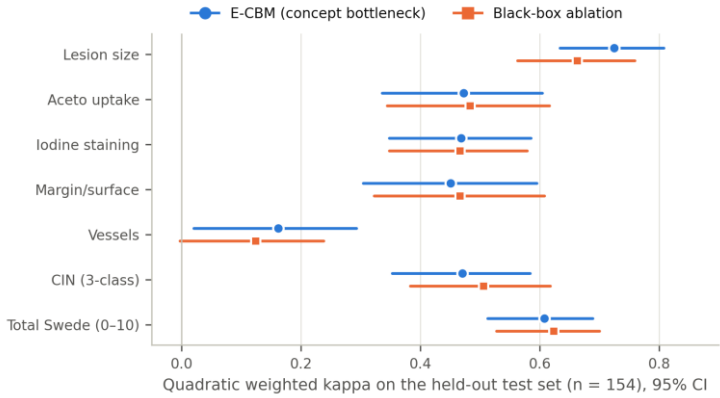

Figure 5. Test-set quadratic weighted kappa with 95% bootstrap confidence intervals for the E-CBM and the black-box ablation (n = 154). The two models share everything except the input of the CIN head.

For the grade itself, Table 6 and Figure 6 give the paired comparison after both models have been calibrated. No difference is detectable for the treatment question (CIN2+ AUC 0.792 versus 0.793; difference +0.001 [−0.034, +0.038]) — the interval rules out a disadvantage of the E-CBM larger than about 0.04 AUC — nor in ordinal agreement (QWK 0.471 versus 0.505; difference +0.035 [−0.073, +0.130], p = 0.49). The black box is better at the screening question (any-CIN AUC 0.806 versus 0.849; difference +0.043 [+0.009, +0.082], p = 0.010) and has better Brier scores and log losses. Of the 24 paired comparisons, only the log-loss difference (p ≈ 0.001) survives a Bonferroni correction; the screening and Brier differences are consistent in direction with it but should be read as suggestive.

Table 6. Paired comparison of the CIN heads of the two models on the same 154 test patients after temperature scaling (T fitted on out-of-fold validation predictions). The difference is black-box minus E-CBM; for ECE, Brier score and log loss lower is better. Before calibration the ECE was 0.023 versus 0.137.

Metric	E-CBM	Black-box	Difference [95% CI]	p
AUC, CIN2+ (treatment)	0.792	0.793	+0.001 [−0.034, +0.038]	0.95
AUC, any CIN (screening)	0.806	0.849	+0.043 [+0.009, +0.082]	0.010
QWK, 3-class	0.471	0.505	+0.035 [−0.073, +0.130]	0.49
ECE	0.082	0.063	−0.019 [−0.087, +0.072]	0.89

Metric	E-CBM	Black-box	Difference [95% CI]	p
Brier score	0.603	0.562	−0.041 [−0.069, −0.013]	0.004
Log loss	0.990	0.929	−0.060 [−0.097, −0.021]	0.001

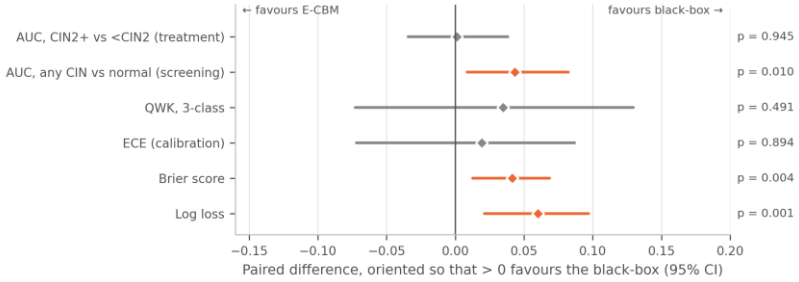

Figure 6. Paired differences between the CIN heads of the two models after calibration, oriented so that positive values favour the black-box model

The CIN head of the E-CBM appeared nearly perfectly calibrated before calibration (ECE 0.023 compared to 0.137 for the black box). But this was a warning sign, not a strength. The E-CBM never gave a confidence above 0.53 to any class for any test patient (mean confidence 0.43 for an accuracy of 0.455; Figure 7b), and its Brier score (0.611) and log loss (1.010) were no better than those of a trivial predictor that always outputs the class frequencies (0.607 and 1.014). A model that hedges toward the prior is automatically well calibrated, ECE rewards the hedging and does not care about the lack of sharpness. The black box, however, used the full range of confidences up to 0.85 and was somewhat overconfident in the middle of that range. Temperature scaling makes clear how much of the difference is a matter of scale. The fitted temperatures moved in opposite directions—T = 0.67 sharpened the under-confident E-CBM, T = 1.29 softened the over-confident black box—and after scaling both models were similarly calibrated (ECE 0.082 versus 0.063; p = 0.89). Yet, the difference in Brier score and log loss was only slightly decreased (Brier 0.611 –> 0.603 for the E-CBM, still at the level of the class-frequency predictor). The weaker probabilities of the E-CBM are therefore not just a matter of scale that can be fixed by a single parameter: less information about the grade is coming through to the output. Among the E-CBM variants, the rule-based head benefited most from scaling ($T \approx 3.1$; Brier 0.660 → 0.585, log loss 1.195 → 0.945) which is consistent with the over-concentration of its probability mass on CIN1 as described above.

Where does the E-CBM information loss occur? Not at the treatment threshold where both models are equal. The loss is concentrated at the border between normal and abnormal (see Figure 7a). After calibration, the black box gives a mean probability of "Normal" of 0.52 for truly normal patients and 0.27 for abnormal patients, a gap of 0.25. For the E-CBM, the corresponding values are 0.49 and 0.31, a gap of 0.18. This is another instance of a conjunction problem: to judge a cervix completely normal you need the evidence that all five components are 0 to survive the passage through five soft probability vectors, while a head that reads the image directly can learn the holistic judgement “nothing abnormal here” in one step.

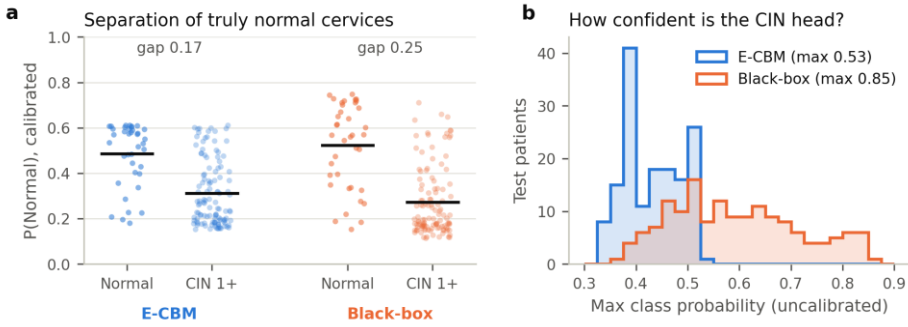

Figure 7. Where the bottleneck loses information. (a) Calibrated probability of "Normal" for truly normal and abnormal (CIN1 or worse) test patients; bars are group means. The black box separates the two groups more widely. (b) Distribution of the CIN head's confidence (highest class probability) before calibration: the E-CBM never exceeds 0.53, the black box reaches 0.85.

The bottleneck returns what is shown in Figure 8. The two cases were selected from the prediction table using a fixed rule before looking at the images: case A is the correctly graded high-grade test patient that has most correctly estimated components (ties broken by the confidence of the grade), and case B is the missed high-grade patient with highest CIN uncertainty. Coincidentally, the two patients have identical true Swede profiles (aceto 2, margin 1, vessels 0, lesion 1, iodine 2; total 6, high grade). For case A, the most probable score for each component is correct, the expected total is 6.1 and the grade is correctly high, but only with a probability of 0.45, an example of a model's cautious output. In case B, the most probable score of each component is wrong: acetowhitening, margins, lesion size and iodine staining are underestimated and vessels overestimated, and the expected total is 3.8. The resulting grade distribution is almost flat (0.34, 0.33, 0.33); its arg-max, Normal, is a missed high-grade lesion. The same miss would have been

reported by a black box, with no indication of where it came from. The E-CBM highlights the clinical observations it misinterpreted, and the almost uniform grade distribution and high uncertainty (u = 0.53, the highest among missed high-grade cases) indicate that the prediction is unreliable—just the kind a clinician would like to review.

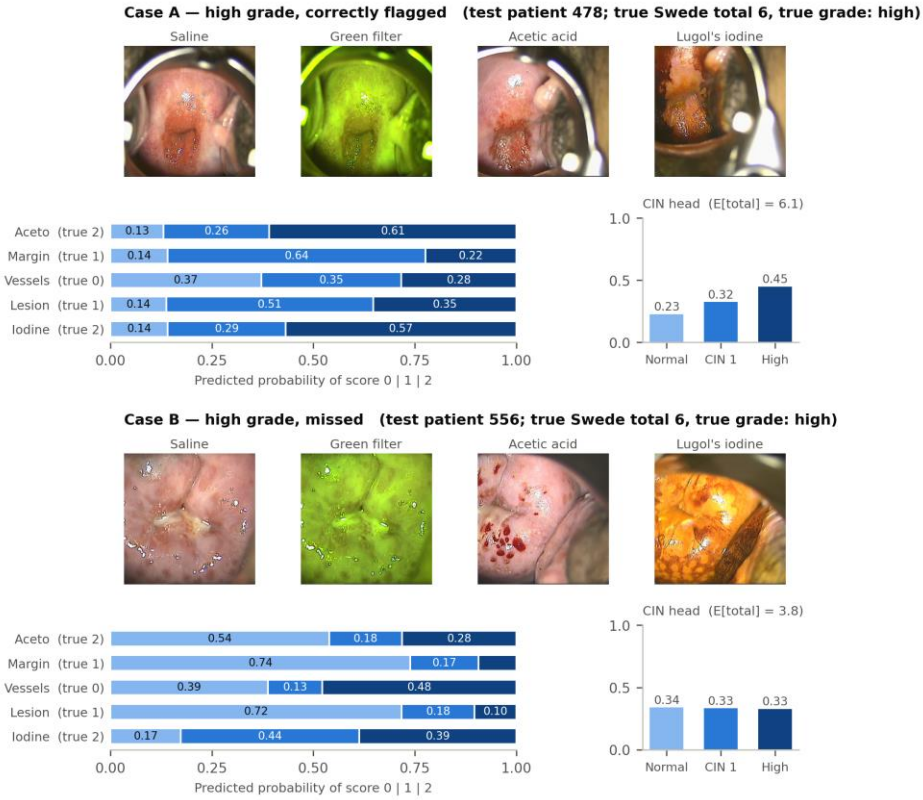

Figure 8. Two test patients with identical true Swede profiles (total 6, high grade). Top of each panel: the four views. Bottom left: the predicted distribution over scores 0, 1 and 2 for each component

IV. DISCUSSION

Our results cannot be judged against published figures, because other colposcopy studies differ in label source, task definition and split unit (Cho et al., 2020; Xue et al., 2020; Yuan et al., 2020). On this dataset, "CIN classification" is in fact Swede-score estimation, and the repository's baselines read the modalities in the wrong order. We therefore used internal reference points. Given the clinician's own Swede components, the grade is recovered almost perfectly (QWK 0.986), so all remaining error lies in estimating the

components from the images. Colposcopists themselves agree only imperfectly on these observations (Jeronimo et al., 2007; Massad et al., 2008). The paired comparison with an identical black-box model supports a qualified claim. Forcing predictions through the five Swede components caused no detectable loss for the treatment decision (CIN2+ AUC 0.792 vs. 0.793; 95% CI of the difference −0.034 to +0.038) and made every prediction traceable to clinical observations. It did, however, cost 0.04 AUC at the normal–abnormal boundary and produced less sharp probabilities. We attribute this cost to the fact that a normal cervix requires five normal observations at once, and such a conjunction is fragile when each observation is a soft probability. The independent and joint heads performed equivalently, which is expected when the concept-to-label mapping is deterministic. The independent head has the added advantage that it cannot collapse from a cold start. Three methodological lessons emerged. Simple checks run before modelling revealed data problems that would otherwise have silently invalidated the study. Evidential heads can fail without an obvious warning sign in the loss, so per-head metrics and uncertainty should be monitored. And a low ECE turned out to be a symptom of excessive caution rather than a strength, which the Brier score and log loss revealed immediately.

This study has several limitations. The labels are Swede-based colposcopic gradings rather than histopathology, and the annotation protocol is undocumented. All data come from a single dataset, and no external validation was performed. The test set is small (154 patients), with very few cases in some classes. Only one backbone, one resolution and one seed were used. The black-box ablation and temperature scaling were designed after the first test evaluation and should be regarded as secondary analyses.

V. CONCLUSION

This study developed a colposcopy model whose predictions can be traced to clinical concepts and evaluated it critically. The data audit showed that the dataset's CIN label is a deterministic relabelling of the Swede score. We therefore recommend using the Swede components as primary endpoints, verifying the modality order and splitting at the patient level. At the clinically decisive treatment threshold, the concept-bottleneck model performed as well as an equivalent black-box model. Its cost was limited to weaker recognition of fully normal cervices and less sharp probabilities. For a screening tool whose recommendations clinicians must be able to question, this trade-off is acceptable.

REFERENCES

Bornstein, J., Bentley, J., Bösze, P., Girardi, F., Haefner, H., Menton, M., et al. (2012). 2011 colposcopic terminology of the International Federation for Cervical Pathology and Colposcopy. Obstetrics & Gynecology, 120(1), 166–172. https://doi.org/10.1097/AOG.0b013e318254f90c

Brier, G. W. (1950). Verification of forecasts expressed in terms of probability. Monthly Weather Review, 78(1), 1–3.

BUET mHealth Lab. (2026). BUET Multi-Center Colposcopy Dataset (Colposcence) [Data set]. GitHub. https://github.com/aisha22-993/Colposcence

Cho, B.-J., Choi, Y. J., Lee, M.-J., Kim, J. H., Son, G.-H., et al. (2020). Classification of cervical neoplasms on colposcopic photography using deep learning. Scientific Reports, 10, Article 13652. https://doi.org/10.1038/s41598-020-70490-4

Dosovitskiy, A., … & Houlsby, N. (2021). An image is worth 16x16 words: Transformers for image recognition at scale. In International Conference on Learning Representations. arXiv. https://doi.org/10.48550/arXiv.2010.11929

Ferris, D. G., & Litaker, M. S. (2005). Interobserver agreement for colposcopy quality control using digitized colposcopic images during the ALTS trial. Journal of Lower Genital Tract Disease, 9(1), 29–35. https://doi.org/10.1097/00128360-200501000-00007

Gneiting, T., & Raftery, A. E. (2007). Strictly proper scoring rules, prediction, and estimation. Journal of the American Statistical Association, 102(477), 359–378. https://doi.org/10.1198/016214506000001437

Guo, C., Pleiss, G., Sun, Y., & Weinberger, K. Q. (2017). On calibration of modern neural networks. In Proceedings of the 34th International Conference on Machine Learning (pp. 1321–1330). PMLR.

Hu, E. J., Shen, Y., Wallis, P., Allen-Zhu, Z., Li, Y., Wang, S., Wang, L., & Chen, W. (2022). LoRA: Low-rank adaptation of large language models. In International Conference on Learning Representations. OpenReview.net.

Hu, L., Bell, D., Antani, S., Xue, Z., Yu, K., et al. (2019). An observational study of deep learning and automated evaluation of cervical images for cancer screening. JNCI: Journal of the National Cancer Institute, 111(9), 923–932. https://doi.org/10.1093/jnci/djy225

Jeronimo, J., Massad, L. S., Castle, P. E., Wacholder, S., & Schiffman, M. (2007). Interobserver agreement in the evaluation of digitized cervical images. Obstetrics & Gynecology, 110(4), 833–840. https://doi.org/10.1097/01.AOG.0000281665.63550.8f

Kendall, A., Gal, Y., & Cipolla, R. (2018). Multi-task learning using uncertainty to weigh losses for scene geometry and semantics. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (pp. 7482–7491). IEEE.

Koh, P. W., Nguyen, T., Tang, Y. S., Mussmann, S., Pierson, E., Kim, B., & Liang, P. (2020). Concept bottleneck models. In Proceedings of the 37th International Conference on Machine Learning (pp. 5338–5348). PMLR.

Loshchilov, I., & Hutter, F. (2019). Decoupled weight decay regularization. In International Conference on Learning Representations.

Mangrulkar, S., Gugger, S., Debut, L., Belkada, Y., Paul, S., & Bossan, B. (2022). PEFT: State-of-the-art parameter-efficient fine-tuning methods [Computer software]. GitHub. https://github.com/huggingface/peft

Massad, L. S., Jeronimo, J., Katki, H. A., & Schiffman, M. (2009). The accuracy of colposcopic grading for detection of high-grade cervical intraepithelial neoplasia. Journal of Lower Genital Tract Disease, 13(3), 137–144. https://doi.org/10.1097/LGT.0b013e31819308d4

Massad, L. S., Jeronimo, J., & Schiffman, M. (2008). Interobserver agreement in the assessment of components of colposcopic grading. Obstetrics & Gynecology, 111(6), 1279–1284. https://doi.org/10.1097/AOG.0b013e31816baed1

Naeini, M. P., Cooper, G. F., & Hauskrecht, M. (2015). Obtaining well calibrated probabilities using Bayesian binning. Proceedings of the AAAI Conference on Artificial Intelligence, 29(1).

Oquab, M., Darcet, T., Moutakanni, T., Vo, H., Szafraniec, M., Khalidov, V., ... & Bojanowski, P. (2023). Dinov2: Learning robust visual features without supervision. arXiv preprint arXiv:2304.07193.

Paszke, A., Gross, S., Massa, F., Lerer, A., Bradbury, J., Chanan, G., et al. (2019). PyTorch: An imperative style, high-performance deep learning library. In Advances in Neural Information Processing Systems (Vol. 32, pp. 8024–8035).

Pedregosa, F., Varoquaux, G., Gramfort, A., Michel, V., Thirion, B., Grisel, O., et al. (2011). Scikit-learn: Machine learning in Python. Journal of Machine Learning Research, 12, 2825–2830.

Sechidis, K., Tsoumakas, G., & Vlahavas, I. (2011). On the stratification of multi-label data. In Joint European Conference on Machine Learning and Knowledge Discovery in Databases (pp. 145–158). Springer. https://doi.org/10.1007/978-3-642-23808-6_10

Sensoy, M., Kaplan, L., & Kandemir, M. (2018). Evidential deep learning to quantify classification uncertainty. In Advances in Neural Information Processing Systems (Vol. 31).

Wolf, T., Debut, L., Sanh, V., Chaumond, J., Delangue, C., Moi, A.,. Rush, A. M. (2020). Transformers: State-of-the-art natural language processing. In Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: System Demonstrations (pp. 38–45). Association for Computational Linguistics. https://doi.org/10.18653/v1/2020.emnlp-demos.6

World Health Organization. (2020). Global strategy to accelerate the elimination of cervical cancer as a public health problem. https://www.who.int/publications/i/item/9789240014107

Xue, P., Ng, M. T. A., & Qiao, Y. (2020). The challenges of colposcopy for cervical cancer screening in LMICs and solutions by artificial intelligence. BMC Medicine, 18, Article 169. https://doi.org/10.1186/s12916-020-01613-x

Xue, P., Tang, C., Li, Q., Li, Y., Shen, Y., et al. (2020). Development and validation of an artificial intelligence system for grading colposcopic impressions and guiding biopsies. BMC Medicine, 18, Article 406. https://doi.org/10.1186/s12916-020-01860-y

Yuan, C., Yao, Y., Cheng, B., Cheng, Y., Li, Y., et al. (2020). The application of deep learning based diagnostic system to cervical squamous intraepithelial lesions recognition in colposcopy images. Scientific Reports, 10, Article 11639. https://doi.org/10.1038/s41598-020-68252-3

Comparative Evaluation of Nine Deep Transfer Learning Architectures for Multi-Class Brain Tumor MRI Classification: A Stratified 5-Fold Cross-Validation Study with Calibration and Grad-CAM Analysis

Halil İbrahim COŞAR[1]
Esra GÜNGÖR ULUTAŞ[2]
Recep Batuhan GÜNAY[3]

1- Asst. Prof. Dr.; Yozgat Bozok University, Faculty of Engineering-Architecture, Department of Electrical and Electronics Engineering. halil.cosar@bozok.edu.tr ORCID No: 0000-0001-8064-2385
2- Lect.; Yozgat Bozok University, Sorgun Vocational School, Department of Computer Technologies. esra.ulutas@bozok.edu.tr ORCID No: 0000-0001-8084-6644
3- Lect.; Yozgat Bozok University, Sorgun Vocational School, Department of Computer Technologies. r.batuhan.gunay@bozok.edu.tr ORCID No: 0009-0001-8913-9775

ABSTRACT

Accurate, automated classification of brain tumor subtypes from magnetic resonance imaging (MRI) is an important component of computer-aided diagnosis, yet much of the existing literature relies on a single train/test split, omits calibration analysis, and reports little to no formal statistical testing or explainability evidence. This study presents a controlled, head-to-head comparison of nine ImageNet-pretrained convolutional neural network (CNN) architectures VGG16, VGG19, ResNet50, ResNet101, DenseNet121, MobileNetV2, EfficientNetB0, InceptionV3, and Xception fully fine-tuned under one identical training protocol for four-class brain MRI classification (glioma, meningioma, pituitary macroadenoma, and healthy) on a 1,137-image dataset derived from a public Mendeley repository. All architectures were evaluated with stratified 5-fold cross-validation, with performance summarized both as fold-level mean ± standard deviation and as pooled out-of-fold (OOF) predictions spanning the entire dataset. Beyond discriminative accuracy, we report calibration (expected calibration error, Brier score, negative log-likelihood), a battery of omnibus and pairwise statistical significance tests (Friedman, Wilcoxon signed-rank, Nadeau–Bengio corrected paired t-test, Cochran's Q, exact McNemar, all Holm-corrected), and Gradient-weighted Class Activation Mapping (Grad-CAM / Grad-CAM++) visual explanations. DenseNet121 achieved the best pooled performance (accuracy 91.29%, macro-F1 91.25%, macro-ROC-AUC 0.984) while simultaneously showing the lowest expected calibration error (ECE) of all nine models, though not the lowest Brier score or log-loss (Section IV-E), despite using only 6.96 million parameters roughly 20 times fewer than VGG19. Meningioma was the most difficult class for every single architecture (F1 74.9–84.5%), and misclassification as meningioma was the single largest confusion source in the pooled confusion matrix, most prominently for glioma (12.5% of all glioma instances). Sample-level McNemar tests distinguished only the two smallest, mobile-oriented architectures (MobileNetV2, EfficientNetB0) as significantly weaker than the remaining seven, which were statistically indistinguishable from one another at this sample size despite visible ranking differences. These findings offer practical, statistically grounded guidance for architecture selection in resource-constrained brain tumor MRI classification pipelines, though we note that our efficiency comparison is based on parameter count alone; FLOPs, VRAM usage, and inference latency were not separately measured and would refine this guidance further.

Keywords—brain tumor classification; magnetic resonance imaging; convolutional neural networks; transfer learning; cross-validation; model calibration; explainable artificial intelligence; Grad-CAM.

I. INTRODUCTION

Gliomas, meningiomas, and pituitary adenomas are among the most frequently encountered primary intracranial neoplasms (Price et al., 2025), and magnetic resonance imaging (MRI) remains the principal non-invasive modality used for their detection (Martucci et al., 2023), subtype differentiation, and treatment planning. Manual interpretation of MRI slices is time-consuming, subject to inter- and intra-observer variability (Mohebbi et al., 2025), and dependent on radiologist experience, which has motivated substantial interest in deep learning-based computer-aided diagnosis (CAD) systems in particular, convolutional neural networks (CNNs) fine-tuned via transfer learning from large natural-image corpora for automating or supporting brain tumor classification from MRI.

Despite a large and rapidly growing body of work in this area, a recent systematic review of 60 studies published between 2020 and 2024 identified recurring methodological weaknesses across the literature: inadequate cross-validation methodology, heavy reliance on a small number of convenience datasets, limited attention to the reliability of model confidence (calibration), and explainability analysis that was inconsistently or rarely reported (Bouhafra & El Bahi, 2025). Individual single-architecture studies frequently report very high accuracy, but controlled, multi-architecture comparisons carried out under one unified protocol, accompanied by formal statistical significance testing and visual model explanations, remain comparatively uncommon.

This study directly addresses these gaps. We (i) fine-tune nine standard ImageNet-pretrained CNN backbones under one identical, fully controlled training protocol; (ii) evaluate every architecture with stratified 5-fold cross-validation, reporting both fold-level and pooled out-of-fold performance; (iii) extend evaluation beyond raw accuracy to a comprehensive metric suite that includes calibration; (iv) formally test the statistical significance of observed performance differences at both the fold level and the individual-sample level; and (v) provide Grad-CAM and Grad-CAM++ visual explanations to qualitatively probe model behavior. The remainder of the paper is organized as follows. Section II reviews related work. Section III describes the dataset and methodology. Section IV reports results. Section V discusses the findings in light of the literature. Section VI states the limitations of the study, and Section VII concludes.

II. RELATED WORK

A systematic review covering 60 brain tumor MRI classification studies published between 2020 and January 2024 found that reported accuracies frequently exceed 97%, that VGG16, ResNet-50, DenseNet, InceptionV3, and Xception are the most commonly evaluated backbones, and that transfer learning is the dominant strategy for coping with limited medical imaging data (Bouhafra & El Bahi, 2025). The same review, however, explicitly flagged persistent gaps: many studies use a single train/test split rather than cross-validation, few report any explainability analysis, class imbalance is often left unaddressed, and clinical validation beyond a held-out test partition is rare (Bouhafra & El Bahi, 2025). These gaps directly motivate the design choices adopted in the present study, in particular our use of stratified 5-fold cross-validation, class-weighted loss, calibration reporting, and Grad-CAM-based explainability.

Several studies have compared multiple pretrained CNN backbones on brain tumor MRI classification in a design similar in spirit to ours. Disci et al. (2025) fine-tuned six pretrained architectures Xception, MobileNetV2, InceptionV3, ResNet50, VGG16, and DenseNet121, all six of which also appear in our nine-architecture set on a four-class, 7,023-image brain tumor MRI dataset, reporting Xception as the strongest model with a weighted accuracy of 98.73% and weighted F1 of 95.29%. Gorenshtein et al. (2026) compared lightweight ResNet variants (ResNet-18, ResNet-34, ResNet-50, both pretrained and trained from scratch) against a custom CNN on the same four-class, 7,023-image collection, reporting pretrained-ResNet accuracies of 98.5–99.2% with an overall one-vs-rest AUC of 1.0. Both studies substantially exceed the absolute accuracies obtained in the present work, a discrepancy we return to in Section V and attribute in large part to the roughly six-fold larger dataset available in those two studies (Disci et al., 2025; Gorenshtein et al., 2026).

A separate line of work pairs a single backbone with Grad-CAM-based explainability. Mohamed Musthafa et al. (2024) fine-tuned ResNet50 for binary (tumor vs. no-tumor) brain tumor detection and paired it with Grad-CAM visualizations, reporting a testing accuracy of 98.52% and arguing that gradient-based localization maps increase clinical trust by aligning model attention with radiologically plausible regions. Our study adopts the same explainability rationale but extends it across nine architectures and formalizes the accuracy-versus-trust trade-off through explicit calibration metrics.

EfficientNet-family architectures have been reported to perform particularly well in this domain when tuned carefully. Vimala et al. (2023) fine-tuned five EfficientNet variants (B0–B4) on the three-class, 3,064-image Figshare CE-MRI dataset, with EfficientNetB2 reaching 99.06% accuracy; Islam et al. (2024) optimized EfficientNetB3 on the same dataset family, reaching 99.69% accuracy. In contrast, our EfficientNetB0 evaluated as only one variant under one shared, non-architecture-specific training recipe, on a smaller four-class dataset that additionally includes a healthy control class reached a considerably lower mean accuracy (87.25%), a contrast we revisit in Section V as evidence that EfficientNet's reported strength in this literature may depend on variant selection, dataset scale, and per-architecture tuning rather than being an intrinsic, protocol-independent property.

Protocols also differ in how the pretrained backbone is configured and how performance is estimated. Dutta et al. (2022) truncated an ImageNet-pretrained InceptionV3 at its "mixed8" block, attached a customized dense classification head (a 1,024-unit fully connected layer with dropout), and trained the network end to end on 150 × 150 inputs, reporting 96.25% accuracy for three-class glioma/meningioma/pituitary classification on the 3,064-image Figshare CE-MRI dataset. This accuracy was measured on a single 80/20 training/validation split, with no separate test partition. Our study likewise fine-tunes every parameter of every backbone (Section III-D), but uses each architecture's full depth and native input resolution and estimates performance with stratified 5-fold cross-validation on held-out test folds, differences we account for explicitly when interpreting cross-study comparisons.

Beyond brain tumor imaging specifically, a growing body of work emphasizes that classification accuracy alone is an incomplete descriptor of model quality for clinical deployment. Guo et al. (2017) showed that modern, high-capacity neural networks are frequently poorly calibrated confident even when wrong and popularized the Expected Calibration Error (ECE) as a standard diagnostic. Sambyal et al. (2023) examined calibration specifically for medical image classifiers and argued that well-calibrated confidence estimates are a precondition for trustworthy clinical decision support, independent of raw accuracy. These findings motivate our systematic reporting of ECE, Brier score, negative log-likelihood, and reliability diagrams alongside conventional discriminative metrics (Section III-F).

The Grad-CAM (Selvaraju et al., 2017) and Grad-CAM++ (Chattopadhay et al., 2018) techniques used for explainability in this study, and in Mohamed Musthafa et al. (2024), generate coarse, class-discriminative localization maps from the gradients flowing into a network's final convolutional layer, without requiring architectural modification or retraining properties that make them broadly applicable across the heterogeneous set of backbones evaluated here.

Finally, correctly quantifying whether one classifier is statistically better than another under k-fold cross-validation is itself a well-studied and frequently mishandled problem. Standard paired t-tests violate the independence assumption because CV folds share overlapping training data, inflating false-positive rates; Nadeau and Bengio (2003) proposed a corrected-variance paired t-test that explicitly accounts for this dependence, and Demšar (2006) provided broader guidance on appropriate omnibus (e.g., Friedman) and post hoc (e.g., Wilcoxon signed-rank with correction for multiple comparisons) procedures for comparing classifiers across multiple independent datasets; the present study adapts this same statistical framework to compare architectures across cross-validation folds, a related but distinct unit of comparison than Demšar's original multiple-dataset setting, and interprets the resulting fold-level tests accordingly. With the partial exception of Disci et al. (2025), which reports point accuracy without confidence intervals or significance tests, none of the brain tumor MRI studies reviewed above apply this class of correction. The present study applies the full recommended battery fold-level Friedman, Wilcoxon signed-rank with Holm correction, and Nadeau–Bengio corrected paired t-tests, together with sample-level Cochran's Q and exact McNemar tests which, to our knowledge, constitutes a more rigorous statistical treatment than is typical in this specific application literature.

III. MATERIALS AND METHODS

A. Dataset

The dataset used in this study is derived from the publicly available Mendeley Data repository “Multi-Class Brain Tumor MRI Dataset: Glioma, Healthy Brain, Meningioma, and Pituitary Macroadenoma” (DOI 10.17632/82mtzd8x72), contributed by Jamil and Khan (2025) and released under a CC BY 4.0 license. The source collection comprises T1- and T2-

weighted MRI slices acquired on hospital-grade scanners and organized into four class folders. The working copy used for the experiments reported here comprised 1,137 usable images distributed as summarized in Table 1: glioma, a primary tumor of glial origin and one of the most common malignant intracranial neoplasms; meningioma, a typically slow-growing, largely extra-axial tumor arising from the meninges; pituitary macroadenoma, a pituitary gland tumor measuring 10 mm or more in diameter (Shafiq & Anastasopoulou, 2025); and a healthy (no-tumor) control class. As in the great majority of the reviewed literature (Bouhafra & El Bahi, 2025), classification is performed at the level of individual 2D slices rather than full 3D volumes, and no tumor-mask segmentation ground truth is provided with the dataset.

Table 1. Class distribution of the working dataset (N = 1,137 images)

Class	N Images	Proportion
Glioma	295	25.9%
Healthy	291	25.6%
Meningioma	277	24.4%
Pituitary Macroadenoma	274	24.1%
Total	1,137	100.0%

Table 2. Per-fold sample sizes under the stratified 5-fold cross-validation protocol

Fold	Train	Validation	Test	Total
1	749	160	228	1,137
2	749	160	228	1,137
3	750	160	227	1,137
4	750	160	227	1,137
5	750	160	227	1,137

B. Data Preprocessing and Augmentation

All images were converted to 3-channel RGB (replicating the single MRI intensity channel) to match the input expectations of ImageNet-pretrained backbones, then resized to each architecture's native input resolution 224×224 pixels for VGG16, VGG19, ResNet50, ResNet101, DenseNet121, MobileNetV2, and EfficientNetB0, and 299×299 pixels for InceptionV3 and Xception (Table 3) and normalized using the standard ImageNet channel-wise mean and standard deviation. For the training split

only, mild augmentation was applied to increase effective sample diversity while preserving diagnostically relevant morphology: random horizontal flipping ($p = 0.5$), random rotation within ±10°, and color jitter (brightness and contrast, each ±0.1). Validation and test images were resized and normalized only, with no augmentation applied.

C. Cross-Validation Protocol

All 1,137 images were partitioned into five stratified folds using scikit-learn's StratifiedKFold (shuffle = true, random seed = 42), which preserves the per-class proportions of Table 1 in every fold. For each fold k, that fold served as the held-out test partition (≈ 20% of the data), while the remaining four folds were further stratified-split into a training subset (≈ 82.5% of the remainder) and a validation subset (≈ 17.5% of the remainder), yielding an overall approximate 66/14/20 train/validation/test ratio per fold; exact per-fold sample counts are given in Table 2. Critically, all nine architectures were trained and evaluated on the identical five fold assignments, producing a fully paired experimental design, which is a precondition for the paired statistical tests used in Section III-G.

Two complementary levels of performance reporting are used throughout Section IV. Fold-level reporting summarizes the five per-fold test-set metric values as mean ± standard deviation, directly reflecting across-fold variability. Pooled out-of-fold (OOF) reporting concatenates the held-out test-set predictions from all five folds, yielding exactly one prediction for every one of the 1,137 images, produced by a model instance that never observed that particular image during its own training. This guarantees image-level exclusivity for every prediction; however, because subject/patient identifiers were not available in the source dataset (Section VI), patient-level independence between training and held-out data could not be verified, so these OOF predictions should not be characterized as leakage-free in the stronger, patient-independent sense.

D. CNN Architectures and Transfer Learning Configuration

Nine architectures were fine-tuned and fully evaluated: VGG16, VGG19 (Simonyan & Zisserman, 2015), ResNet50, ResNet101 (He et al., 2016), DenseNet121 (Huang et al., 2017), MobileNetV2 (Sandler et al., 2018), EfficientNetB0 (Tan & Le, 2019), InceptionV3 (Szegedy et al., 2016), and

Xception (Chollet, 2017), as summarized in Table 3. All backbones were instantiated with ImageNet-1k pretrained weights (Deng et al., 2009) via the timm library (Wightman, 2019), with the original 1000-way classification head replaced by a newly initialized linear layer with four outputs. Every parameter of every network was fine-tuned no layers were frozen so that each architecture could adapt its learned filters to the MRI domain under one shared optimization recipe. InceptionV3's auxiliary classifier branch was disabled to simplify the training loop.

A tenth architecture, InceptionResNetV2 (Szegedy et al., 2017), was included in the original experimental design but is excluded from the comparative analysis reported here; see Section IV-I for details.

Table 3. CNN architectures evaluated, ordered by trainable parameter count

Architecture	Family	Input Size	Params (M)	Mean Epochs to Early Stop
MobileNetV2	Inverted-residual / depthwise-separable	224×224	2.23	21.8 ± 4.0
EfficientNetB0	Compound-scaled MBConv	224×224	4.01	25.2 ± 2.4
DenseNet121	Densely-connected	224×224	6.96	22.0 ± 2.8
Xception	Depthwise-separable	299×299	20.82	17.6 ± 5.1
InceptionV3	Multi-branch Inception	299×299	21.79	14.8 ± 3.2
ResNet50	Residual	224×224	23.52	29.8 ± 0.4
ResNet101	Residual	224×224	42.51	28.6 ± 2.3
VGG16	Sequential	224×224	134.28	13.2 ± 3.3
VGG19	Sequential	224×224	139.59	14.8 ± 1.9
InceptionResNetV2[a]	Inception-residual hybrid	299×299	—	incomplete

[a] *InceptionResNetV2 was part of the original design; its fifth cross-validation fold did not complete, so it is excluded from all quantitative comparisons (Section IV-I).*

E. Training Configuration

All models were optimized with Adam (Kingma & Ba, 2015) with an initial learning rate of 1×10^{-4} and a weight decay of 1×10^{-4}, batch size 32, for a maximum of 30 epochs, with early stopping (patience 7 epochs, monitored on validation loss) and a ReduceLROnPlateau learning-rate schedule (factor 0.5, patience 3 epochs). Training used automatic mixed precision where GPU hardware was available. To counteract the mild class imbalance evident in

Table 1, the cross-entropy loss was weighted per class using scikit-learn's balanced class-weighting scheme (inverse class frequency), recomputed from each fold's own training split; because class proportions in this dataset are close to balanced (24.1-25.9% per class, Table 1), the resulting weights are themselves close to uniform (approximately 0.96-1.04), so this choice functions as a standard protective default rather than a targeted correction for severe imbalance and is not expected to materially bias the calibration analysis in Section IV-E. A global random seed of 42 was fixed, with deterministic cuDNN settings, to support reproducibility. The implementation used PyTorch (Paszke et al., 2019), torchvision, and timm (Wightman, 2019) for modeling, and scikit-learn (Pedregosa et al., 2011) together with SciPy and statsmodels for metric computation and statistical testing.

F. Evaluation Metrics

For every fold and for the pooled OOF predictions, three families of metrics were computed. Threshold-based metrics include accuracy, balanced accuracy, precision/recall/F1 (macro-, weighted-, and micro-averaged), Jaccard index, Hamming loss, top-2 accuracy, macro G-mean, macro Youden's J statistic, macro specificity and negative predictive value, Matthews correlation coefficient (MCC), and Cohen's kappa. Ranking-based metrics include one-vs-rest ROC-AUC and PR-AUC (macro-, weighted-, and micro-averaged). Probabilistic/calibration-based metrics include the multiclass Brier score, negative log-likelihood (equivalently, log-loss), Expected Calibration Error (ECE) and Maximum Calibration Error (MCE) computed over 15 equal-width confidence bins following Guo et al. (2017), class-wise ECE, mean predicted confidence, and reliability diagrams. Class-level variants of the relevant metrics (precision, recall/sensitivity, specificity, F1, NPV, Youden's J, ROC-AUC, PR-AUC, class-wise ECE) were additionally computed for each of the four classes.

G. Statistical Significance Testing

Statistical significance was assessed with two complementary designs, both at $\alpha = 0.05$ with Holm–Bonferroni correction applied to every family of pairwise comparisons. At the fold level ($n = 5$ paired observations per model), an omnibus Friedman test ($k = 9$ related samples) was computed for each metric in Table 6, followed, where the omnibus test was informative, by pairwise Wilcoxon signed-rank tests and by Nadeau–Bengio corrected

resampled paired t-tests (Nadeau & Bengio, 2003), which inflate the naive variance estimate of the mean fold-wise difference by a factor that accounts for the train/test overlap inherent to k-fold cross-validation and are widely recommended as a more conservative, better-calibrated alternative to a naive paired t-test in this setting (Demšar, 2006; Nadeau & Bengio, 2003). At the sample level, using the pooled OOF predictions aligned across all nine models on the same 1,137 images, an omnibus Cochran's Q test was computed on binary correctness, together with a Friedman test on per-sample negative log-likelihood and Brier score; pairwise comparisons used McNemar's test on correctness the exact binomial variant for architecture pairs with fewer than 25 discordant predictions, and the continuity-corrected asymptotic χ^2 variant otherwise (in practice every pair exceeded 25 discordant predictions, so the continuity-corrected asymptotic form was used throughout) and Wilcoxon signed-rank tests on per-sample negative log-likelihood and Brier score, all Holm-corrected across the 36 pairwise architecture comparisons.

H. Explainability: Grad-CAM and Grad-CAM++

Gradient-weighted Class Activation Mapping (Grad-CAM; Selvaraju et al., 2017) and its generalized variant Grad-CAM++ (Chattopadhay et al., 2018) were computed for every fold of every architecture using an architecture-agnostic implementation that hooks the last convolutional layer of each backbone, requiring no structural modification. For each fold, two correctly and two incorrectly classified examples per class (16 images per fold per architecture per method) were selected for visualization, allowing qualitative inspection of (i) whether activated regions correspond to anatomically plausible intracranial content rather than background or artifact, and (ii) whether misclassified examples reveal interpretable, spatially localized explanations for specific, recurring error patterns.

IV. RESULTS

A. Overall Fold-Level Performance

Table 4 and Figure 1 summarize fold-level performance, averaged across the five test folds. DenseNet121 achieved the highest mean accuracy (91.30 ± 2.19%), balanced accuracy, macro-F1, and MCC, followed closely by Xception (90.94 ± 1.53%); VGG19 ranked third (89.97 ± 0.47%). The remaining six architectures spanned 85.93% (MobileNetV2) to 89.54%

(ResNet50). Standard deviations across folds varied considerably: VGG19 was the most stable architecture by a wide margin (accuracy SD = 0.47 points, versus 1.5–4.5 points for all other models), indicating that while it did not achieve the single highest mean accuracy, it showed the lowest observed across-fold variability of any architecture in this study a practically relevant property distinct from peak average performance. MobileNetV2 and EfficientNetB0 were both the lowest-performing and the least stable architectures (accuracy SD = 4.23 and 4.50 points, respectively), compounding their weaker average performance with greater unpredictability across folds.

Table 4. Fold-level test-set performance, mean ± SD across five folds (ranked by accuracy)

Architecture	Accuracy (%)	Bal. Acc. (%)	Macro-F1 (%)	MCC	ROC-AUC	PR-AUC
DenseNet121	91.30 ± 2.19	91.31 ± 2.24	91.23 ± 2.35	0.885 ± 0.028	0.984 ± 0.006	0.960 ± 0.015
Xception	90.94 ± 1.53	90.93 ± 1.51	90.92 ± 1.52	0.880 ± 0.020	0.985 ± 0.009	0.961 ± 0.020
VGG19	89.97 ± 0.47	90.01 ± 0.48	90.06 ± 0.52	0.868 ± 0.007	0.984 ± 0.004	0.962 ± 0.011
ResNet50	89.54 ± 2.70	89.56 ± 2.74	89.45 ± 2.91	0.862 ± 0.035	0.983 ± 0.005	0.956 ± 0.012
VGG16	89.36 ± 2.34	89.38 ± 2.30	89.29 ± 2.34	0.859 ± 0.030	0.977 ± 0.008	0.942 ± 0.014
ResNet101	89.01 ± 2.56	89.04 ± 2.53	89.10 ± 2.49	0.855 ± 0.033	0.981 ± 0.007	0.950 ± 0.022
InceptionV3	88.83 ± 3.24	88.85 ± 3.21	88.75 ± 3.33	0.852 ± 0.042	0.980 ± 0.010	0.953 ± 0.022
EfficientNetB0	87.25 ± 4.50	87.26 ± 4.50	87.19 ± 4.61	0.831 ± 0.060	0.972 ± 0.017	0.930 ± 0.041
MobileNetV2	85.93 ± 4.23	85.94 ± 4.27	85.76 ± 4.54	0.814 ± 0.055	0.967 ± 0.017	0.921 ± 0.035

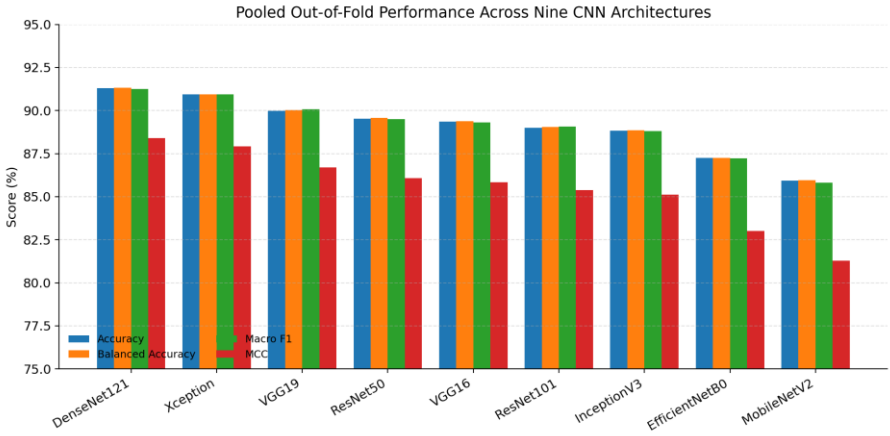

Figure 1. Fold-level mean accuracy, balanced accuracy, macro-F1, and Matthews correlation coefficient (MCC) (± SD across five folds) for all nine architectures, ranked left to right by accuracy.

B. Pooled Out-of-Fold Performance

Table 5 reports pooled out-of-fold performance across the full 1,137-image dataset. The ranking of all nine architectures is identical to the fold-level ranking in Table 4, providing evidence that the observed performance ordering is stable rather than an artifact of how results are aggregated. DenseNet121 attained an OOF accuracy of 91.29%, macro-F1 of 91.25%, MCC of 0.884, Cohen's κ of 0.884, and macro ROC-AUC of 0.984; Xception followed closely (accuracy 90.94%, macro-F1 90.95%, MCC 0.879), only 0.35 percentage points behind. VGG19 ranked third (89.97%), 0.97 percentage points behind Xception and 1.32 points behind DenseNet121, while the gap from DenseNet121 down to the lowest-ranked MobileNetV2 (85.93%) reached 5.36 percentage points.

Table 5. Pooled out-of-fold (OOF) performance across all 1,137 images (ranked by accuracy)

Architecture	Accuracy (%)	Macro-F1 (%)	MCC	ROC-AUC	ECE	Brier	NLL
DenseNet121	91.29	91.25	0.884	0.984	0.036	0.135	0.273
Xception	90.94	90.95	0.879	0.984	0.037	0.133	0.270
VGG19	89.97	90.07	0.867	0.982	0.042	0.155	0.325
ResNet50	89.53	89.52	0.861	0.980	0.040	0.156	0.309
VGG16	89.36	89.31	0.858	0.976	0.047	0.170	0.367
ResNet101	89.01	89.08	0.854	0.982	0.040	0.162	0.311
InceptionV3	88.83	88.81	0.851	0.979	0.065	0.182	0.428
EfficientNetB0	87.25	87.22	0.830	0.973	0.084	0.202	0.524
MobileNetV2	85.93	85.81	0.813	0.967	0.090	0.233	0.560

C. Per-Class Performance

Figure 2 presents per-class F1-scores for all nine architectures. Two patterns are completely consistent across every architecture: pituitary macroadenoma was the easiest class to classify for all nine models (F1 93.3–96.8%), and meningioma was the hardest class for all nine models (F1 74.9–84.5%), with healthy (89.9–94.9%) and glioma (85.2–90.2%) falling in between. This uniformity across architecturally diverse backbones suggests that meningioma's classification difficulty in this dataset is a property of the class itself plausibly its greater radiological heterogeneity and variable size/location rather than a weakness specific to any one model family.

Precision–recall balance for meningioma varied meaningfully across architectures. VGG19 achieved the highest meningioma recall of any model (86.64%) alongside a comparatively low meningioma precision (78.18%, fourth-lowest of nine), suggesting some tendency to over-predict the meningioma label; DenseNet121, by contrast, showed more balanced meningioma precision and recall (85.29% and 83.75%, respectively), consistent with its higher overall meningioma F1 (84.52%, the best of all nine models). MobileNetV2 recorded both the lowest meningioma precision (75.46%) and the lowest meningioma recall (74.37%) of any architecture, the weakest precision–recall balance for this class overall.

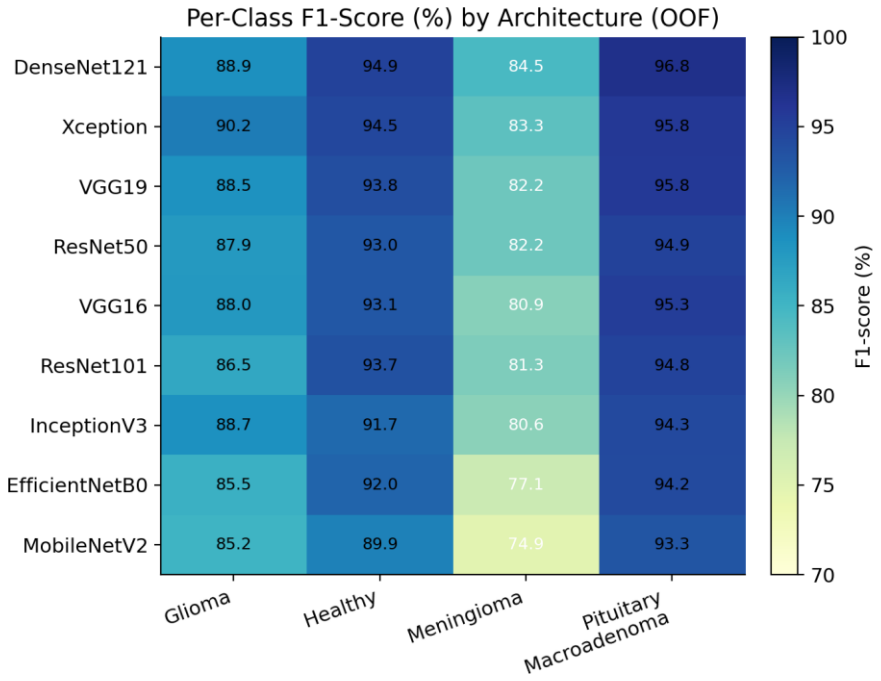

Figure 2. Per-class F1-score (%) for each architecture on pooled OOF predictions. Meningioma is the hardest class and pituitary macroadenoma the easiest for every architecture.

D. Confusion Analysis

Figure 3 shows the pooled confusion matrix summed across all nine architectures' OOF predictions (10,233 classification instances in total). The single largest source of error in the entire study was glioma predicted as meningioma: 332 instances, corresponding to 12.5% of all glioma instances and roughly 85% of all glioma misclassifications. Meningioma's own errors were considerably more diffuse, spreading across glioma (196 instances, 7.9%), healthy (140, 5.6%), and pituitary macroadenoma (112, 4.5%). Healthy and pituitary macroadenoma, the two easiest classes, were also the most mutually isolated: cross-confusion between these two classes totaled only 35 of 10,233 instances (0.34%). Aggregate (pooled) recall was 85.3% for glioma, 93.4% for healthy, 82.0% for meningioma, and 95.9% for pituitary macroadenoma.

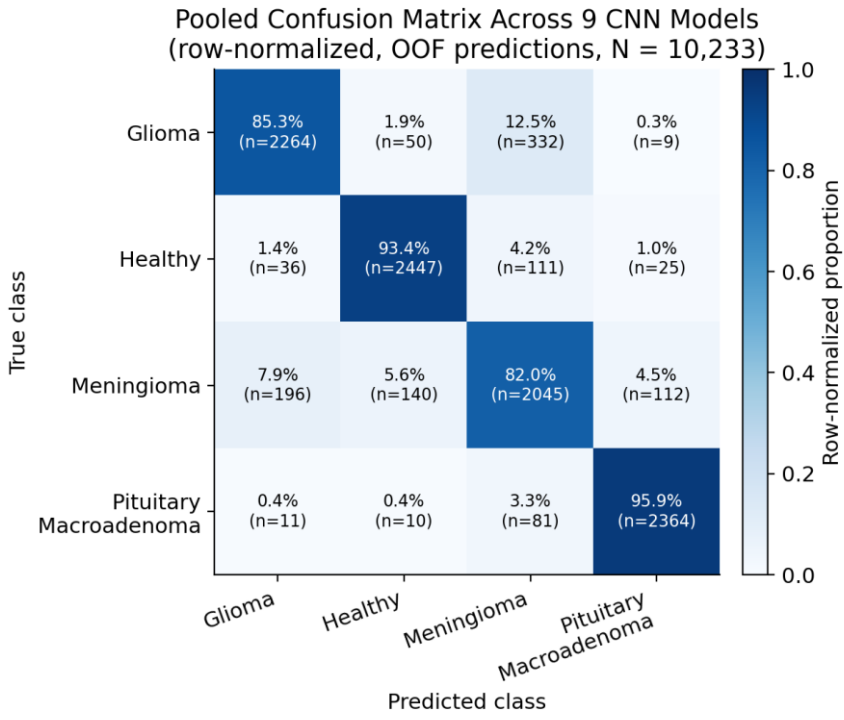

Figure 3. Pooled confusion matrix summed across all nine architectures' out-of-fold predictions (row-normalized; N = 10,233 classification instances). Glioma-as-meningioma is the single largest confusion pair in the study.

E. Calibration Analysis

Figure 4 and Table 5 summarize calibration. Because these are pooled OOF metrics, each reported value reflects the combined calibration of five separately fitted fold models per architecture rather than a single final model; a model retrained on the complete dataset for deployment would need its calibration reassessed independently. DenseNet121 had the lowest ECE (0.036) of all nine architectures, making it the best-calibrated model by the most direct, bin-based calibration measure an alignment with its top accuracy that is not guaranteed a priori, since larger or more expressive networks often become overconfident even as their accuracy improves (Guo et al., 2017). DenseNet121 was not, however, uniformly best across every probabilistic metric: Xception obtained a marginally lower Brier score (0.133 versus 0.135) and a marginally lower negative log-likelihood (0.270 versus 0.273), while ranking a close second on both ECE (0.037) and accuracy. Because Brier score

and NLL are proper scoring rules that jointly reflect calibration and sharpness/discrimination rather than calibration alone, we treat ECE as the primary calibration indicator in this comparison; no single architecture was uniformly superior across all three probabilistic metrics, and DenseNet121 and Xception are best regarded as jointly the best-calibrated pair. MobileNetV2 and EfficientNetB0 were simultaneously the two least accurate and the two worst-calibrated models on every probabilistic metric (ECE 0.090 and 0.084, Brier 0.233 and 0.202, respectively), compounding weaker discrimination with less trustworthy confidence estimates.

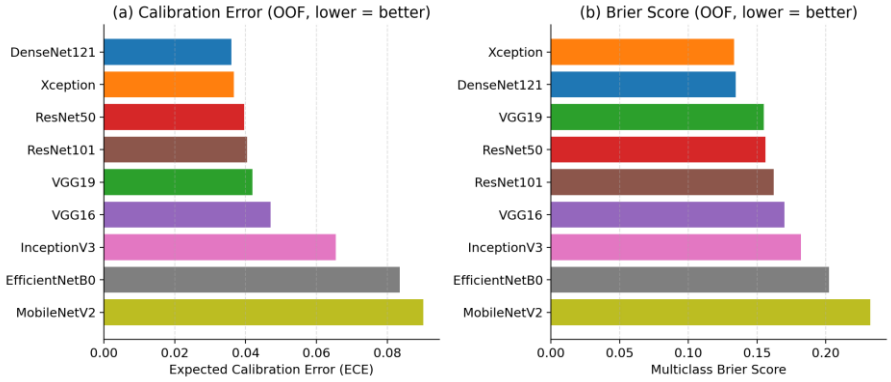

Figure 4. Calibration comparison across the nine architectures on pooled OOF predictions: (a) Expected Calibration Error; (b) multiclass Brier score. Lower values indicate better calibration.

Figure 5(a) shows the reliability diagram for DenseNet121 on the pooled OOF predictions. The great majority of predictions (923 of 1,137, or 81.2%) fell in the highest confidence bin (0.93–1.0), where the model was very well calibrated (mean predicted confidence 0.988 versus observed accuracy 0.970). The remaining 214 predictions (18.8% of the dataset) fell into nine lower-confidence bins spanning 0.33–0.93 confidence, where calibration was markedly less reliable. Within this minority, the six bins spanning 0.53–0.93 confidence (194 samples, 17.1% of the dataset) were consistently overconfident, with observed accuracy trailing predicted confidence by 4.5–21.2 percentage points. The single largest calibration gap, and the source of the non-zero maximum calibration error (MCE = 0.553), occurred in the smallest, lowest-confidence bin with any correct predictions (0.40–0.47, only 4 samples), where all four predictions happened to be correct despite comparatively low confidence an underconfident, opposite-direction outlier driven by a very small bin count rather than a systematic pattern. This

indicates that DenseNet121's low overall ECE is driven primarily by excellent calibration in its dominant high-confidence regime, while its comparatively rare low- and mid-confidence predictions are noisier and, in the mid-confidence range, mildly overconfident a distinction that would be invisible from ECE alone and illustrates the practical value of inspecting the full reliability diagram, not just its scalar summary, before clinical use (Sambyal et al., 2023). Figure 5(b) presents the corresponding one-vs-rest ROC curves for DenseNet121: meningioma shows the lowest per-class AUC (0.965), consistent with its status as the hardest class identified in Section IV-C, while healthy and pituitary macroadenoma reach near-ceiling discrimination (AUC = 0.995) and glioma falls in between (AUC = 0.980).

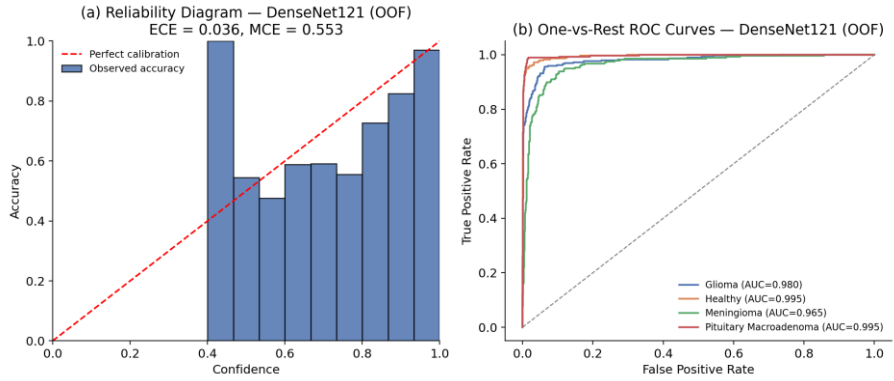

Figure 5. (a) Reliability diagram and (b) one-vs-rest ROC curves for DenseNet121, the best-performing architecture, on pooled OOF predictions.

F. Statistical Significance Testing Results

Table 6 reports the fold-level omnibus Friedman test across all nine architectures for each of eleven metrics. The omnibus null hypothesis of no difference among architectures was rejected ($p < 0.05$) for ten of eleven metrics, most strongly for the calibration-related metrics negative log-likelihood ($\chi^2 = 25.81$, $p = 0.001$) and Brier score ($\chi^2 = 21.97$, $p = 0.005$), and for MCC ($\chi^2 = 18.51$, $p = 0.018$); only balanced accuracy narrowly failed to reach significance ($p = 0.056$). Despite this clear omnibus signal, none of the 36 pairwise fold-level comparisons reached significance after Holm correction, whether tested with the Wilcoxon signed-rank test or with the more conservative Nadeau–Bengio corrected paired t-test, for either accuracy or macro-F1. This combination a significant omnibus test alongside null pairwise tests is an expected consequence of the low statistical power available from

only five paired fold-level observations per comparison combined with a Holm correction across 36 simultaneous comparisons, rather than evidence that no true pairwise differences exist; it is a direct illustration of the caution urged by Nadeau and Bengio (2003) and Demšar (2006) against over-interpreting fold-averaged rankings from a single k-fold study as if they were statistically confirmed.

Table 6. Fold-level omnibus Friedman test results by metric (k = 9 architectures, n = 5 folds)

Metric	χ^2 (Friedman)	p-value	Significant (α = 0.05)
Accuracy	16.84	0.032	Yes
Balanced accuracy	15.15	0.056	No
Macro-F1	16.11	0.041	Yes
Weighted-F1	16.11	0.041	Yes
MCC	18.51	0.018	Yes
Cohen's κ	16.59	0.035	Yes
ROC-AUC (macro)	17.39	0.026	Yes
PR-AUC (macro)	15.95	0.043	Yes
ECE	17.55	0.025	Yes
Brier score	21.97	0.005	Yes
NLL	25.81	0.001	Yes

The sample-level tests, which leverage all 1,137 pooled OOF predictions per model rather than only five fold averages, were considerably more sensitive. Cochran's Q test on binary correctness across all nine models was highly significant ($Q = 46.48$, $p = 1.9\times10^{-7}$), as were Friedman tests on per-sample negative log-likelihood ($\chi^2 = 1269.2$, $p \approx 1.0\times10^{-268}$) and Brier score ($\chi^2 = 1219.7$, $p \approx 5.4\times10^{-258}$). Table 7 lists the six of 36 pairwise exact McNemar comparisons that remained significant after Holm correction; every single one involves either MobileNetV2 or EfficientNetB0 — the two smallest, mobile-oriented architectures being outperformed by a larger model (DenseNet121, Xception, VGG19, or ResNet50). No pairwise McNemar comparison among the remaining seven architectures (VGG16, VGG19, ResNet50, ResNet101, DenseNet121, InceptionV3, Xception) reached significance, indicating that although these seven models are numerically ranked in Tables 4 and 5, their pooled predictive correctness is not statistically distinguishable at this sample size. Sample-level pairwise Wilcoxon tests on

per-sample negative log-likelihood were far more discriminating: 27 of 36 pairs differed significantly, showing that confidence-quality (calibration/sharpness) differences between architectures are detectable even where raw correctness differences are not underscoring the value of reporting calibration alongside accuracy rather than accuracy alone.

Table 7. Statistically significant pairwise sample-level McNemar comparisons after Holm correction (of 36 pairs tested)

Comparison	Discordant n	χ^2 (McNemar)	p (Holm)	Favors
DenseNet121 vs. MobileNetV2	143	25.17	< 0.001	DenseNet121
MobileNetV2 vs. Xception	155	20.23	< 0.001	Xception
DenseNet121 vs. EfficientNetB0	126	16.07	0.002	DenseNet121
VGG19 vs. MobileNetV2	158	12.82	0.011	VGG19
EfficientNetB0 vs. Xception	138	12.18	0.015	Xception
ResNet50 vs. MobileNetV2	143	11.19	0.026	ResNet50

G. Model Complexity and Training Dynamics

Figure 6 plots trainable parameter count against pooled OOF accuracy. The relationship is clearly non-monotonic: the most accurate architecture, DenseNet121, uses only 6.96 million parameters fewer than every other model except MobileNetV2 (2.23M) and EfficientNetB0 (4.01M), which were in fact the two least accurate architectures overall. VGG16 and VGG19, at 134.28M and 139.59M parameters respectively (roughly 19–20 times larger than DenseNet121), ranked fifth and third. This pattern suggests that, on a dataset of this size, an architecture's connectivity pattern and inductive bias (e.g., DenseNet's dense feature reuse) is a stronger predictor of transfer-learning performance than raw parameter count.

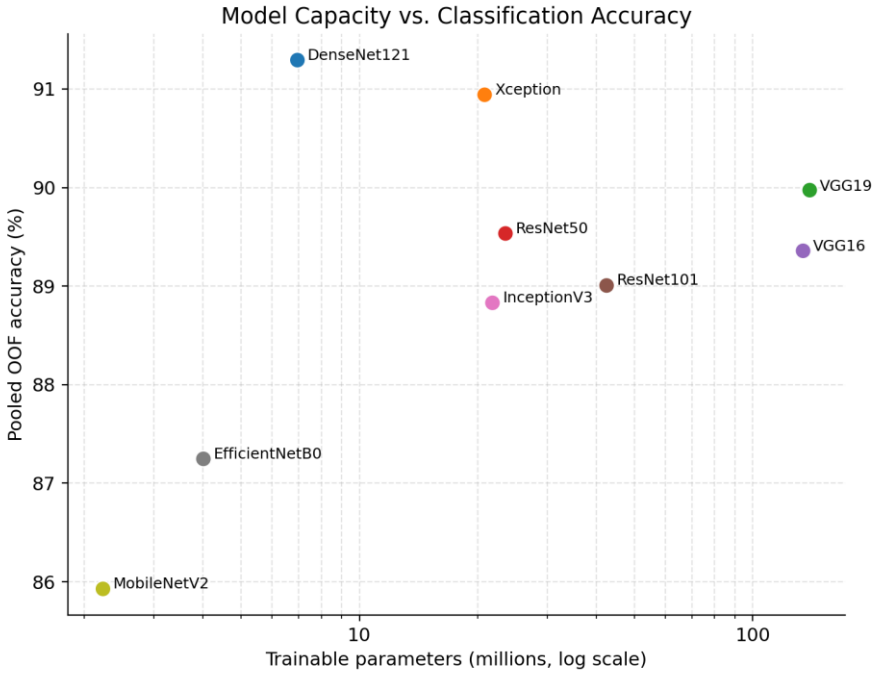

Figure 6. Trainable parameter count (millions, log scale) versus pooled OOF accuracy. DenseNet121 achieves the highest accuracy with far fewer parameters than the VGG models.

Figure 7 reports the mean number of epochs completed before early stopping. ResNet50 and ResNet101 nearly exhausted the full 30-epoch training budget in almost every fold (29.8 ± 0.4 and 28.6 ± 2.3 epochs, respectively), rarely triggering early stopping, yet finished only fourth and sixth by accuracy. VGG16, VGG19, and InceptionV3 required the fewest training epochs before early stopping (13.2–14.8 mean epochs) despite VGG16/VGG19 being by far the largest models by parameter count a further indication that parameter count predicts neither final accuracy nor the number of epochs to convergence in a simple way for this task. Note that fewer epochs do not necessarily imply lower wall-clock training cost, since per-epoch compute time also differs by architecture and was not separately measured here.

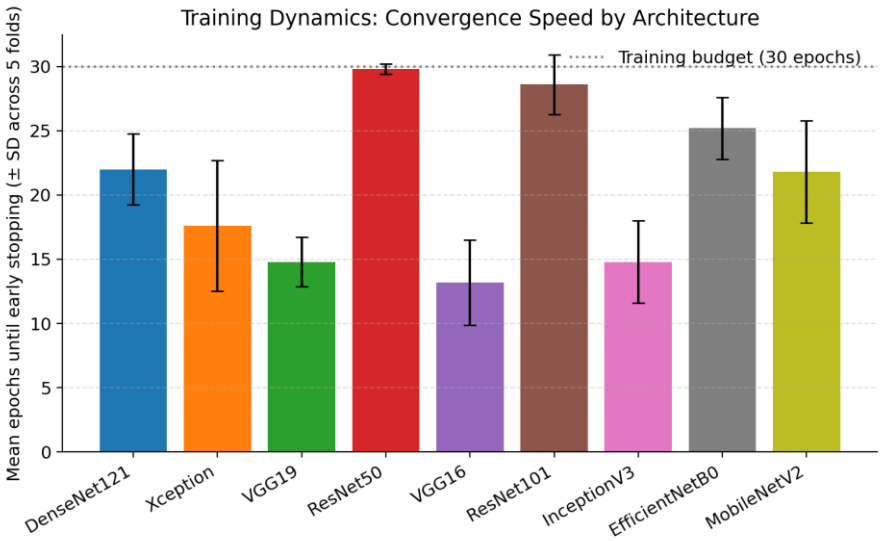

Figure 7. Mean number of training epochs completed before early stopping, averaged across the five folds (error bars: ± SD). The dashed line marks the 30-epoch training budget.

H. Explainability: Grad-CAM Analysis

Figure 8 shows a representative Grad-CAM comparison for DenseNet121 on two glioma test images. For a correctly classified glioma (predicted confidence 0.981), activation was concentrated over central intracranial parenchyma. For a glioma image misclassified as meningioma (predicted confidence 0.989), the activated region extended toward the peripheral, dural margin of the brain rather than remaining confined to a focal central lesion a pattern spatially consistent with the typically extra-axial, peripheral radiological presentation of meningiomas, and offering a plausible, anatomically grounded qualitative explanation for this specific error type, which Section IV-D identified as the single largest confusion source in the entire study. Grad-CAM++ maps were likewise generated for every fold and architecture (Section III-H); for this same misclassified example, the Grad-CAM++ map showed the same peripheral/dural shift as its Grad-CAM counterpart, and only Grad-CAM is shown in Fig. 8 for brevity. Across the correctly classified examples inspected for all four classes, activated regions were consistently located within intracranial tissue rather than background, skull, or scan-artifact regions, providing a basic sanity check that model decisions are anchored in anatomically relevant image content. We emphasize

that this is an illustrative, case-level qualitative observation rather than a systematic, quantitatively validated localization analysis, since the dataset provides no expert tumor-mask annotations against which Grad-CAM maps could be scored (Section VI).

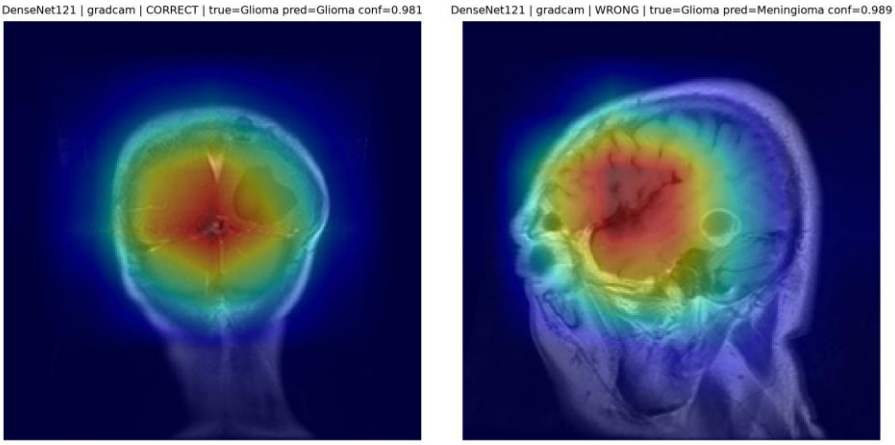

Figure 8. Grad-CAM visualizations for DenseNet121 on two glioma test images: (left) correctly classified, activation centered on intracranial parenchyma; (right) misclassified as meningioma, activation shifted toward the peripheral/dural margin.

I. Note on the Excluded Architecture (InceptionResNetV2)

InceptionResNetV2 was included in the original ten-architecture experimental design, and its first four cross-validation folds trained and evaluated successfully. However, the fifth fold's training run did not complete within the available computing session. Because the paired statistical design used throughout Section IV requires results from all five folds for every architecture, a partial four-fold result for InceptionResNetV2 would not be directly comparable to the complete nine-architecture set on equal footing. InceptionResNetV2 is therefore excluded from all quantitative comparisons in this paper; completing its evaluation is noted as future work in Section VII.

V. DISCUSSION

The best result obtained in this study DenseNet121 at 91.29% pooled OOF accuracy is noticeably lower than the 96–99%+ accuracies commonly reported in the reviewed literature (Bouhafra & El Bahi, 2025; Disci et al., 2025; Dutta et al., 2022; Gorenshtein et al., 2026; Islam et al., 2024; Mohamed

Musthafa et al., 2024; Vimala et al., 2023). We do not interpret this gap as evidence that DenseNet121 is a weaker architecture; rather, dataset scale appears to be one plausible contributor, alongside other differences discussed below. Disci et al. (2025) and Gorenshtein et al. (2026) both trained on 7,023 images (roughly 5,700 training images per split), compared with the 749–750 training images available per fold in this study a difference of nearly an order of magnitude. Vimala et al. (2023) and Islam et al. (2024) trained EfficientNet variants on 3,064 images without a healthy control class, versus our 1,137 images across four classes including healthy. Given that our EfficientNetB0 (87.25% mean accuracy) trails the EfficientNetB2 accuracy of 99.06% reported by Vimala et al. (2023) and the EfficientNetB3 accuracy of 99.69% reported by Islam et al. (2024) by roughly 12 percentage points, while also differing in EfficientNet sub-variant and in the presence of per-architecture hyperparameter tuning that our shared-recipe design deliberately avoided, dataset scale and per-architecture tuning intensity are plausible contributors to the absolute accuracy gap, though differences in dataset provenance, patient-level composition, split strategy, MRI acquisition protocol, and preprocessing across studies also differ simultaneously and prevent attributing the gap to any single factor; we can, however, rule out any deficiency in the architectures themselves as the primary explanation. This interpretation is consistent with the systematic review's own observation that the field's very high reported accuracies frequently coincide with small, convenience-sampled test partitions (Bouhafra & El Bahi, 2025).

A more direct point of comparison is architecture ranking rather than absolute accuracy. Disci et al. (2025) evaluated six architectures that overlap exactly with six of our nine, and found Xception to be the single best performer; in our study, Xception ranked a close second behind DenseNet121, with less than 0.4 percentage points separating their pooled accuracies. That Xception and DenseNet121 both emerge as top-tier performers across two independently trained, differently sized, and differently sourced datasets even as the identity of the single ‘best’ model differs is a modestly reassuring sign of external validity for these two architectures specifically in this application domain, and is consistent with our sample-level finding (Section IV-F) that the top-tier architectures are not reliably distinguishable from one another at typical brain-tumor-MRI dataset sizes; small ranking differences between competitive architectures should not, on their own, be read as robust superiority claims.

That VGG19 and VGG16 — with 19–20 times as many parameters as the strongest-performing DenseNet121 — nonetheless finished third and fifth of nine respectively, is consistent with the broader transfer-learning literature's finding that early, over-parameterized CNN backbones remain competitive on small target datasets when combined with ImageNet initialization, data augmentation, class-weighted loss, and early stopping as regularizers, effectively compensating for the parameter/sample-size mismatch. Conversely, MobileNetV2 and EfficientNetB0, architectures explicitly designed for parameter efficiency on large-scale natural-image data, were the two weakest performers here and, per Section IV-F, the only architectures statistically distinguishable from the rest. A plausible explanation is that these architectures' compound-scaling and inverted-residual design choices are tuned for a training regime large datasets, extensive augmentation, long schedules that our uniform, comparison-oriented protocol did not provide; this is testable in future work through architecture-specific hyperparameter search (Section VII).

The finding that DenseNet121 was simultaneously the most accurate and the best-calibrated architecture (Section IV-E) is notable because accuracy and calibration are not intrinsically aligned: modern high-capacity networks are frequently overconfident even as their accuracy rises (Guo et al., 2017). DenseNet's dense connectivity and explicit feature reuse across layers (Huang et al., 2017) is a plausible mechanistic contributor to both properties simultaneously, though confirming this would require controlled ablation beyond the scope of the present comparison.

The dominant, architecture-independent confusion between glioma and meningioma (Section IV-D) has a plausible radiological grounding: meningiomas are typically extra-axial tumors arising from the meninges at the brain's periphery, whereas gliomas are intra-axial, arising from glial tissue within the brain parenchyma; when a glioma's imaging presentation is comparatively peripheral or cortically based, its appearance can superficially resemble the typical peripheral location of a meningioma. The qualitative Grad-CAM pattern observed for one such misclassified case in Section IV-H activation extending toward the dural margin rather than remaining central is consistent with, though not proof of, this explanation, and the fact that all nine architecturally diverse models make this same error at comparable rates strengthens the case that it reflects a genuine, dataset-level ambiguity rather than an idiosyncratic weakness of any single model.

Finally, the statistical-testing results in Section IV-F carry a direct methodological implication for this literature: fold-level pairwise tests, which are the more conservative and arguably more appropriate unit of comparison for k-fold cross-validation (Demšar, 2006; Nadeau & Bengio, 2003), found no significant differences among any of the nine architectures on accuracy or macro-F1, despite an omnibus signal that real differences exist somewhere among them. Sample-level McNemar tests, which pool all individual predictions and are correspondingly higher-powered, were able to confirm that the two smallest, mobile-oriented architectures are significantly weaker than the rest, but could not statistically separate the remaining seven. Because the large majority of the studies reviewed in Section II report a single point accuracy without any significance test (Bouhafra & El Bahi, 2025), we regard the explicit demonstration in an application domain where such testing is rare that most pairwise architecture rankings in a study of this scale are not statistically confirmed, as a substantive methodological contribution in its own right, independent of which architecture ultimately ranked highest.

VI. LIMITATIONS

This study has several limitations. First, the dataset originates from a single public repository with 1,137 images in total (≈ 227 images per fold test partition); this sample size, while sufficient to reveal a significant omnibus effect across architectures, limits the statistical power of pairwise comparisons, as demonstrated directly in Section IV-F. Second, classification was performed at the level of individual 2D slices rather than 3D volumes, and the dataset does not document how many slices, if any, originate from the same underlying patient/scan; if multiple slices per subject are present, the effective number of independent samples underlying the sample-level statistical tests in Table 7 would be smaller than the nominal 1,137, a data-documentation gap that is itself representative of a broader issue noted across the reviewed literature (Bouhafra & El Bahi, 2025). Third, no tumor-segmentation ground truth is available, so the Grad-CAM analysis in Section IV-H is necessarily qualitative and illustrative rather than a quantitatively validated localization assessment (e.g., via intersection-over-union against expert-annotated masks). Fourth, all nine architectures were trained under one shared, non-architecture-specific hyperparameter recipe, a deliberate design choice that enables fair, controlled comparison but likely understates the accuracy each architecture could reach with dedicated per-architecture tuning, as suggested by the higher accuracies reported for individually tuned

EfficientNet variants elsewhere in the literature (Islam et al., 2024; Vimala et al., 2023). Fifth, InceptionResNetV2 could not be included in the final comparison due to an incomplete training run (Section IV-I). Finally, ground-truth labels were inherited from the source dataset without independent re-adjudication by the present authors; label noise, if present in the source data, would affect all nine architectures similarly but cannot be separately quantified here.

VII. CONCLUSION AND FUTURE WORK

This study compared nine ImageNet-pretrained CNN architectures for four-class brain tumor MRI classification under one identical optimization and evaluation protocol (with architecture-specific native input resolutions) and a shared stratified 5-fold cross-validation design, reporting discriminative accuracy, calibration, per-class behavior, formal statistical significance, and Grad-CAM-based explainability. DenseNet121 achieved the best pooled accuracy (91.29%) and macro-F1 (91.25%) while also achieving the lowest expected calibration error (ECE) among all nine architectures though Xception obtained a marginally lower Brier score and log-loss, so no single architecture was uniformly best-calibrated (Section IV-E) despite using substantially fewer parameters than six of the eight other models, with Xception a close second in accuracy, macro-F1, and ECE. Meningioma was the most difficult class for every architecture, and glioma-as-meningioma was the single largest, architecture-independent source of error, a pattern with a plausible basis in the differing typical radiological locations of the two tumor types. Formal statistical testing showed a significant omnibus effect of architecture choice on most metrics, but pairwise differences among the seven strongest architectures were not statistically confirmed at this sample size; only the two smallest, mobile-oriented architectures (MobileNetV2, EfficientNetB0) were reliably distinguishable from the rest in pairwise tests of binary prediction correctness; per-sample calibration (negative log-likelihood) differences were significant for the large majority of architecture pairs (Section IV-F), so this indistinguishability is specific to McNemar-tested correctness and should not be read as a general absence of performance differences.

Future work should prioritize: completing the InceptionResNetV2 evaluation to restore the original ten-architecture design; per-architecture hyperparameter and augmentation tuning to establish each backbone's

achievable ceiling rather than its performance under one shared recipe; external, multi-institution validation to test generalization beyond a single-source dataset; expert tumor-mask annotation to enable quantitative, IoU-based evaluation of Grad-CAM localization quality; and ensembling of the top-performing, partially diverse architectures identified here (DenseNet121 and Xception) as a natural next step toward higher and more robust accuracy.

REFERENCES

Bouhafra, S., & El Bahi, H. (2025). Deep learning approaches for brain tumor detection and classification using MRI images (2020 to 2024): A systematic review. *Journal of Imaging Informatics in Medicine, 38*(3), 1403–1433. https://doi.org/10.1007/s10278-024-01283-8

Chattopadhay, A., Sarkar, A., Howlader, P., & Balasubramanian, V. N. (2018). Grad-CAM++: Generalized gradient-based visual explanations for deep convolutional networks. In *2018 IEEE Winter Conference on Applications of Computer Vision (WACV)* (pp. 839–847). IEEE. https://doi.org/10.1109/WACV.2018.00097

Chollet, F. (2017). Xception: Deep learning with depthwise separable convolutions. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)* (pp. 1251–1258). IEEE. https://doi.org/10.1109/CVPR.2017.195

Demšar, J. (2006). Statistical comparisons of classifiers over multiple data sets. *Journal of Machine Learning Research, 7*, 1–30.

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

Disci, R., Gurcan, F., & Soylu, A. (2025). Advanced brain tumor classification in MR images using transfer learning and pre-trained deep CNN models. *Cancers, 17*(1), Article 121. https://doi.org/10.3390/cancers17010121

Dutta, P., Sathi, K. A., & Islam, M. S. (2022). *Multi-classification of brain tumor images using transfer learning based deep neural network* (arXiv:2206.08543). arXiv. https://doi.org/10.48550/arXiv.2206.08543

Gorenshtein, A., Liba, T., & Goren, A. (2026). Lightweight transfer learning models for multi-class brain tumor classification: Glioma, meningioma, pituitary tumors, and no tumor MRI screening. *Journal of Imaging Informatics in Medicine, 39*, 2309–2318. https://doi.org/10.1007/s10278-025-01686-1

Guo, C., Pleiss, G., Sun, Y., & Weinberger, K. Q. (2017). On calibration of modern neural networks. In *Proceedings of the 34th International Conference on Machine Learning* (Proceedings of Machine Learning Research, Vol. 70, pp. 1321–1330). PMLR.

He, K., Zhang, X., Ren, S., & Sun, J. (2016). Deep residual learning for image recognition. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)* (pp. 770–778). IEEE. https://doi.org/10.1109/CVPR.2016.90

Huang, G., Liu, Z., van der Maaten, L., & Weinberger, K. Q. (2017). Densely connected convolutional networks. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)* (pp. 4700–4708). IEEE. https://doi.org/10.1109/CVPR.2017.243

Islam, M. M., Talukder, M. A., Uddin, M. A., Akhter, A., & Khalid, M. (2024). BrainNet: Precision brain tumor classification with optimized EfficientNet architecture. *International Journal of Intelligent Systems, 2024*, Article 3583612. https://doi.org/10.1155/2024/3583612

Jamil, N., & Khan, A. A. (2025). *Multi-class brain tumor MRI dataset: Glioma, healthy brain, meningioma, and pituitary macroadenoma* (Version 3) [Data set]. Mendeley Data. https://doi.org/10.17632/82mtzd8x72.3

Kingma, D. P., & Ba, J. (2015). Adam: A method for stochastic optimization. In *Proceedings of the 3rd International Conference on Learning Representations (ICLR)*. https://doi.org/10.48550/arXiv.1412.6980

Martucci, M., Russo, R., Schimperna, F., D'Apolito, G., Panfili, M., Grimaldi, A., Perna, A., Ferranti, A. M., Varcasia, G., Giordano, C., & Gaudino, S. (2023). Magnetic resonance imaging of primary adult brain tumors: State of the art and future perspectives. *Biomedicines, 11*(2), Article 364. https://doi.org/10.3390/biomedicines11020364

Mohamed Musthafa, M., Mahesh, T. R., Vinoth Kumar, V., & Guluwadi, S. (2024). Enhancing brain tumor detection in MRI images through explainable AI using Grad-CAM with Resnet 50. *BMC Medical Imaging, 24*, Article 107. https://doi.org/10.1186/s12880-024-01292-7

Mohebbi, A., Mohammadzadeh, S., Zare, A. H., Moradi, Z., Abbasian Ardakani, A., & Mohammadi, A. (2025). Assessing inter-rater reliability of MRI features in glioma: A multi-radiologist agreement study. *BMC Medical Imaging, 25*, Article 480. https://doi.org/10.1186/s12880-025-01941-5

Nadeau, C., & Bengio, Y. (2003). Inference for the generalization error. *Machine Learning, 52*(3), 239–281. https://doi.org/10.1023/A:1024068626366

Paszke, A., Gross, S., Massa, F., Lerer, A., Bradbury, J., Chanan, G., Killeen, T., Lin, Z., Gimelshein, N., Antiga, L., Desmaison, A., Köpf, A., Yang, E., DeVito, Z., Raison, M., Tejani, A., Chilamkurthy, S., Steiner, B., Fang, L., . . . Chintala, S. (2019). PyTorch: An imperative style, high-performance deep learning library. In *Advances in Neural Information Processing Systems 32* (pp. 8024–8035). Curran Associates.

Pedregosa, F., Varoquaux, G., Gramfort, A., Michel, V., Thirion, B., Grisel, O., Blondel, M., Prettenhofer, P., Weiss, R., Dubourg, V., Vanderplas, J., Passos, A., Cournapeau, D., Brucher, M., Perrot, M., & Duchesnay, É. (2011). Scikit-learn: Machine learning in Python. *Journal of Machine Learning Research, 12*, 2825–2830.

Price, M., Ballard, C. A. P., Benedetti, J. R., Kruchko, C., Barnholtz-Sloan, J. S., & Ostrom, Q. T. (2025). CBTRUS statistical report: Primary brain and other central nervous system tumors diagnosed in the United States in 2018–2022. *Neuro-Oncology, 27*(Suppl. 4), vi1–vi66. https://doi.org/10.1093/neuonc/noaf194

Sambyal, A. S., Niyaz, U., Krishnan, N. C., & Bathula, D. R. (2023). Understanding calibration of deep neural networks for medical image classification. *Computer Methods and Programs in Biomedicine, 242*, Article 107816. https://doi.org/10.1016/j.cmpb.2023.107816

Sandler, M., Howard, A., Zhu, M., Zhmoginov, A., & Chen, L.-C. (2018). MobileNetV2: Inverted residuals and linear bottlenecks. In *Proceedings of the IEEE*

Conference on Computer Vision and Pattern Recognition (CVPR) (pp. 4510–4520). IEEE. https://doi.org/10.1109/CVPR.2018.00474

Selvaraju, R. R., Cogswell, M., Das, A., Vedantam, R., Parikh, D., & Batra, D. (2017). Grad-CAM: Visual explanations from deep networks via gradient-based localization. In *Proceedings of the IEEE International Conference on Computer Vision (ICCV)* (pp. 618–626). IEEE. https://doi.org/10.1109/ICCV.2017.74

Shafiq, I., & Anastasopoulou, C. (2025, November 7). Pituitary adenoma. In *StatPearls*. StatPearls Publishing. https://www.ncbi.nlm.nih.gov/books/NBK554451/

Simonyan, K., & Zisserman, A. (2015). Very deep convolutional networks for large-scale image recognition. In *Proceedings of the 3rd International Conference on Learning Representations (ICLR)*. https://doi.org/10.48550/arXiv.1409.1556

Szegedy, C., Ioffe, S., Vanhoucke, V., & Alemi, A. A. (2017). Inception-v4, Inception-ResNet and the impact of residual connections on learning. *Proceedings of the AAAI Conference on Artificial Intelligence, 31*(1), 4278–4284. https://doi.org/10.1609/aaai.v31i1.11231

Szegedy, C., Vanhoucke, V., Ioffe, S., Shlens, J., & Wojna, Z. (2016). Rethinking the Inception architecture for computer vision. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)* (pp. 2818–2826). IEEE. https://doi.org/10.1109/CVPR.2016.308

Tan, M., & Le, Q. (2019). EfficientNet: Rethinking model scaling for convolutional neural networks. In *Proceedings of the 36th International Conference on Machine Learning* (Proceedings of Machine Learning Research, Vol. 97, pp. 6105–6114). PMLR.

Vimala, B. B., Srinivasan, S., Mathivanan, S. K., Mahalakshmi, Jayagopal, P., & Dalu, G. T. (2023). Detection and classification of brain tumor using hybrid deep learning models. *Scientific Reports, 13*, Article 23029. https://doi.org/10.1038/s41598-023-50505-6

Wightman, R. (2019). *PyTorch image models* [Computer software]. GitHub. https://doi.org/10.5281/zenodo.4414861

Runtime-Reflexive Architectures: Continuous Evaluation Mechanisms as the Backbone of Adaptive AI-Powered Software Systems

Halit ÇETİNER[1]
Sedat METLEK[2]

1- Assoc. Prof.; Isparta University of Applied Sciences, Vocational School of Technical Sciences, Computer Technology. halitcetiner@isparta.edu.tr ORCID No: 0000-0001-7794-2555
2- Assoc. Prof.; Burdur Mehmet Akif Ersoy University, Faculty of Engineering and Architecture, Burdur Mehmet Akif Ersoy University. sedatmetlek@mehmetakif.edu.tr ORCID No: 0000-0002-0393-9908

ABSTRACT

Artificial intelligence stands out as one of the most dynamic and interdisciplinary fields within contemporary technological developments. To fully grasp the impact of this influential technology on data science and information technology, it is necessary to examine it through the lenses of various disciplines. The complexity of software systems has continued to grow over time. By incorporating artificial intelligence into software solutions and moving away from static structures toward dynamic, learning-oriented architectural models, the field has gained mechanisms for flexibility, adaptability, and continuous evaluation, all aimed at improving architectural design processes. It is possible to develop models based on machine learning, deep learning, large language models, and autonomous agents by employing architectures that have evolved into an interdisciplinary space, where fields such as cognitive science, linguistics, organizational theory, ethics, and law directly influence the transformation of traditional software engineering. This study examines the factors involved in the creation of such models under two main headings: the first concerns the impact of interdisciplinary interactions on architectural design, while the second addresses the role of continuous architectural evaluation mechanisms in facilitating adaptation. It is observed that continuous evaluation mechanisms are now being integrated into AI-based architectural design processes to address issues that arise during real-time operation problems that might not have been present when systems were initially designed. The adaptable, flexible nature of artificial intelligence systems has also contributed to the automation of decision-making processes. In light of this, it is evident that by integrating machine learning technologies with natural language processing techniques, it is possible to strengthen such integration across a broad spectrum of applications, ranging from parameter optimization to the Internet of Things.

Keywords –Machine learning, natural language processing, scalable intelligent, runtime architectural, learning-oriented architecture

INTRODUCTION

The architectural evolution of AI-powered intelligent software systems stands out as one of the most dynamic and multidisciplinary areas in contemporary technological development. These systems need to be examined not only through the lens of software engineering but also from the perspectives of other disciplines, including artificial intelligence, human-computer interaction, data science, and information technology. The development of software systems can be analysed by treating both their

technological infrastructures and their design paradigms as processes that are constantly changing and evolving. As AI-based technologies have advanced, software systems have become considerably more complex, and this has made new approaches to architectural design necessary. Traditional software architectures, with their static and deterministic structures, do not sit well with the dynamic, learning-oriented nature of artificial intelligence, and this mismatch has made an architectural transformation unavoidable. Consequently, concepts such as flexibility, adaptation, and continuous evaluation have moved to the forefront of architectural design processes. A systematic literature review by Sobhy et al. highlights the challenges involved in addressing uncertainties in software architectures and underscores the critical importance of continuity and dynamism. The authors note that existing architectural evaluation approaches fail to adequately bridge the gap between the design phase and runtime, and they point to a lack of systematic methods for continuous architectural evaluation (Dalia Sobhy Rami Bahsoon, 2021). This observation matters because it brings to light the need for continuous feedback mechanisms and adaptive structures within the architectural evolution of AI-powered systems.

Artificial intelligence matters a great deal in intelligent software systems, since it does more than automate decision-making: it also strengthens these systems' capacity for self-optimization and learning. In the literature, Erişen's work offers a good illustration of this. The study explores how technology is being transformed in architectural applications, discussing concepts such as "intelligent spaces" and "learning environments" through the lens of the changing relationship between humans and technology (Erişen, 2021). As AI-powered components are built into software, systems become more adaptable and more responsive to changes in their environment. And as architectural design moves away from static forms toward dynamic, learning-based structures, the use of artificial intelligence keeps growing.

Significant transformations have begun with the integration of natural language processing and machine learning techniques into AI-powered intelligent software systems. A review of the literature reveals a study focused on developing natural language processing models for iOS applications, in which an AI-based natural language processing application was created using Apple’s tools, such as CreateML and CoreML (Sahin,

2021). With developments aimed at enabling intelligent software systems to communicate more effectively with users and derive meaningful insights from complex datasets, it is evident that artificial intelligence components are advancing in a modular and scalable manner.

Given that quality management systems enhance efficiency in industrial applications by reducing quality-related costs, there is existing literature that highlights how artificial intelligence technologies through their software-based capabilities can impact quality-related costs within business enterprises (Ever & Demircioğlu, 2022). It is evident that software architectures which must be approached from the perspectives of both technical and economic sustainability should be designed to align with the digital transformation processes of businesses, while also necessitating cost-effective solutions.

In the evolution of AI-powered software systems, the adoption of a complex systems approach within architectural design processes as explored in a study by Lekesiz and Gürer highlights the significance of interaction networks and dynamic processes. By evaluating an architectural design studio through the lens of complex systems theory, the study demonstrates how software architectures can function coherently within heterogeneous structures, facilitating the management of interactions among multi-layered components and enabling a deeper understanding of the system's overall behaviour (Lekesiz & Gürer, 2022).

One area that can serve as an example of the transformation in software architecture is the shift observed within the field of architecture itself. Özsoy and his colleagues highlight a paradigm shift in which architectural forms are no longer viewed merely as static products, but rather as dynamic, autonomous processes structures that are not solely defined by a designer, but that instead self-organize and take shape in response to environmental factors (Özsoy et al., 2023).

Global crises, such as pandemics, directly influence the architectural evolution of intelligent software systems, leading to an increase in remote work and education, as well as highlighting the growing importance of integrating smart building systems (Coza, 2024). In the post-pandemic era, the emergence of cloud-based and distributed structures well-suited for flexible work models has driven further advancement in the scalability and

accessibility criteria for the architectural design of intelligent software systems.

Hyperparameter optimization the process of adjusting parameters to their most suitable values during the development of AI-driven architectures, algorithms, or software represents the technical aspect of architectural evolution. There are existing studies that achieve optimal performance by employing an AI-driven approach to hyperparameter optimization for the artificial bee colony algorithm, applied to datasets consisting of images of coffee leaves (Çetiner & Metlek, 2025). There are also studies focused on enhancing performance in the hyperparameter optimization of convolutional neural networks through the use of metaheuristic methods, such as genetic algorithms and particle swarm optimization (Ayla Gülcü, 2019). These two studies essentially argue that AI-powered components, built to squeeze the best performance out of software architectures by keeping them running efficiently, represent an innovative approach.

In the context of advancements in artificial intelligence technologies within the security domain, it is evident that AI-based security solutions need to be integrated into traditional anti-malware systems to address the inherent weaknesses of signature-based and dynamic analysis methods used for detecting malicious software (Aygör & Aktan, 2021). This finding can no longer be treated as a matter of mere need; implementing it has become a necessity, one that keeps security components in intelligent software systems adaptive and ready to respond rapidly to new threats.

Another significant element in the architectural evolution of intelligent software systems is the integration of Internet of Things technologies. Connecting objects to the Internet brings with it more management and control processes, which in turn fuels a growing need for AI-powered, real-time adaptive security constraints in intelligent systems that are able to keep up with fast-moving technology (Erdal & Ergüzen, 2020). Given this finding, the implementation of AI-powered solutions for the management and control of internet-connected objects is of great importance; it helps address emerging security vulnerabilities, thereby ensuring both the security and effective control of these objects.

The architectural evolution of intelligent software systems necessitates an interdisciplinary approach, enabling adaptive development through the integration of knowledge from diverse fields and the application of

continuous evaluation methods (Sobhy et al., 2021). It is stated that, in line with multidisciplinary development and sociocultural changes and taking into account pandemic situations there is a need for interdisciplinary approaches capable of adapting to such circumstances (Coza, 2024; Erişen, 2021).

To conclude, this study looks at the architectural evolution of AI-powered intelligent software systems as they move through complex processes marked by rapid technological change, intense interdisciplinary interaction, and the constant need to adapt, all considered within the context of existing literature. Key areas—adaptive design approaches, hyperparameter optimization techniques, security solutions, and IoT integration—need to be addressed through mechanisms of continuous evaluation. The aim of this study has been to offer a holistic perspective on how the architectures of these systems are evolving into increasingly flexible, self-learning, and sustainable structures, driven by the capabilities that artificial intelligence offers. The first part of the study deals with interdisciplinary interactions in the architectural evolution of AI-powered intelligent software systems; the second part then looks at how continuous architectural evaluation mechanisms shape the adaptation of these systems. The sections that follow cover the integration of natural language processing and machine learning techniques into software architecture; the adaptive nature of security architectures within AI-powered software systems; the role of hyperparameter optimization methods in the performance of AI components; the integration of Internet of Things technologies; and the architectural flexibility and scalability of intelligent software systems.

The Role of Interdisciplinary Interaction in Shaping the Architecture of AI-Powered Intelligent Software Systems

Over the past twenty years, AI-powered intelligent software systems have played a key part in helping computer science move beyond its traditional boundaries and develop into a multidisciplinary field of research. The integration of artificial intelligence components into these systems has fundamentally transformed traditional software architectures moving away from deterministic execution models, layered abstraction frameworks, and rigid, pre-defined workflows (Li et al., 2026). Architectural design, a critical

aspect of software engineering, has evolved into an interdisciplinary space directly influenced by fields such as cognitive science, linguistics, organizational theory, ethics, and law. This development is largely driven by the integration of machine learning models, deep learning models, large language models, and autonomous agents into the very core of software architecture (Hassouna et al., 2025; Khapre et al., 2026; Pambudi et al., 2026). In this section, interdisciplinary interactions a key aspect of architectural evolution are examined within an academic framework.

If we were to examine the evolution of software architecture in three stages, the first period that comes to mind is one dominated by monolithic and cloud-native architectures, in which artificial intelligence was positioned outside the system (Brojabasi et al., 2026; Fox, 2017; Mandler, 2006). During this period, artificial intelligence viewed merely as an API service or an analytical tool was not a core component of architectural design. The second phase in the evolution of software architecture marked the emergence of a period in which machine learning and large language models began to be integrated into microservices architecture (Mekki & Ksentini, 2025). The third phase, which represents the current era, encompasses a broad and comprehensive period characterized by the active use of machine learning and deep learning models specifically their capabilities for reasoning, memory management, and tool invocation.

When examining interdisciplinary interactions across three critical planes, the first plane highlights cognitive sciences and autonomy models. Within this context drawing upon multi-agent systems and distributed artificial intelligence the concept has emerged that software architecture should be viewed not merely as a technical structure, but as a cognitive system. This second plane provided the infrastructure on which microservices architecture, container orchestration, DevOps practices, and the scalable, manageable integration of artificial intelligence components could be built (Garro et al., 2018; Khapre et al., 2026; Rodrigues et al., 2025). This critical juncture, AI-powered tools designed to help human architects with tasks such as selecting architectural patterns, estimating quality, and generating documentation have pushed back the boundaries of software engineering and encouraged a more collaborative approach to architectural decision-making. A third critical development has also emerged: the natural language capabilities of large language models are

fundamentally changing the way users interact with software. These models now act as an intermediary layer that interprets what a user intends and turns it into actionable outcomes. This evolution has given rise to a new technological paradigm—one that goes beyond traditional human-computer interaction and can be described as human-agent collaboration.

The Impact of Continuous Architectural Evaluation Mechanisms on How AI-Powered Systems Adapt

Much of the push behind the shift in traditional software architecture evaluation paradigms has come from the growing use of AI-powered intelligent software systems in production environments. Conventional evaluation methods worked by testing system behaviour during the design phase with static scenarios (Sobhy et al., 2022). However, as deep learning, machine learning models, large language models, and autonomous agents have become central to software architecture, factors such as uncertainties during real-time operation and data drift have exposed the limitations of static evaluation approaches (Marda et al., 2024). The continuous evaluation of architectural integration plays a significant role in expanding the adaptability of AI-powered software systems. Rather than assessing the quality attributes of models such as those derived from deep learning architectures only at design time, the process of monitoring, analysing, and when necessary re-evaluating these systems throughout their runtime is referred to as a "continuous architectural evaluation" mechanism (Sobhy et al., 2021). The point mentioned here is highly significant in terms of model quality and can be explained through three scenarios. The first scenario, known as data drift a phenomenon affecting machine learning and deep learning models that rely on specific training data occurs when the data distribution in the production environment changes over time, leading to a decline in the accuracy of the deployed model (Wong & Perumal, 2025). Furthermore, a second situation arises in large language models and agent systems which rely on contextual inference and probabilistic reasoning making it difficult to predict and verify their architectural behaviour (Marda et al., 2024). The third case represents the unusual, exceptional situations that arise when working in complex and dynamic environments (Lu et al., 2025).

The examination of deep learning-based or machine learning-based models whether developed or proposed through a continuous architectural evaluation mechanism can be approached from both reactive and proactive perspectives. From a reactive standpoint, this process involves assessing the adequacy of architectural decisions based on historical data and proposing corrective measures should any potential degradation in the model be detected (Sobhy et al., 2021). In contrast to reactive assessment, proactive assessment performs time-series forecasting aimed at predicting future pattern behaviour (Sobhy et al., 2021).

In the implementation of a continuous architectural evaluation mechanism for AI-powered systems, the fact that the outputs of machine learning and deep learning models are probability-based reduces the predictability of architectural behaviour.

RESULTS AND DISCUSSION

A study has been conducted to examine the differences between machine learning/deep learning models and traditional software, utilizing continuous evaluation mechanisms across key areas such as adaptive design approaches, hyperparameter optimization techniques, security solutions, and Internet of Things integration. This research provides a holistic perspective on how the architectures of these systems leveraging the capabilities offered by artificial intelligence are evolving into increasingly flexible, self-learning, and sustainable structures. The first part of the study addresses the interdisciplinary interactions involved in the architectural evolution of AI-powered intelligent software systems, while the second part discusses the impact of continuous architectural evaluation mechanisms on the adaptation of such systems. Ultimately, the architectural evolution of AI-powered intelligent software systems necessitates an interdisciplinary approach, the implementation of continuous evaluation methods, and the adoption of dynamic structures capable of adapting to changing conditions. The sustainability and effectiveness of these systems appear achievable only through the establishment of an uninterrupted evaluation and adaptation cycle that spans from the design phase to runtime.

REFERENCES

Aygör, D., & Aktan, E. (2021). Kötücül yazılımların tespitinde imza temelli Aygör, D., & Aktan, E. (2021). Kötücül yazılımların tespitinde imza temelli ve dinamik analiz yöntemlerinin zayıflıkları: Örnek olay çalışması. *Gazi*

Üniversitesi Mühendislik Mimarlık Fakültesi Dergisi, *37*(1), 305–316.

Ayla Gülcü, Z. K. (2019). Konvolüsyonel Sinir Ağlarında Hiper-Parametre Optimizasyonu Yöntemlerinin İncelenmesi. *Gazi Üniversitesi Fen Bilimleri Dergisi Part C: Tasarım ve Teknoloji, 7(2), 503-522*.

Brojabasi, S., Paul, S., & Mitra, A. (2026). Cloud native engineering: A comprehensive review of principles, practices, and challenges. *Advances in Computers*, *141*, 331–353.

Çetiner, H., & Metlek, S. (2025). ABC+CNN-SH: Detection of peruvian coffea leaf diseases with a new hybrid classification algorithm based on ABC optimization and CNN. *Computers and Electronics in Agriculture*, *239*, 111114.

Coza, H. (2024). The Impact of the COVID-19 Pandemic on Architectural Space Needs TT - COVID-19 Pandemisinin Mimari Mekân İhtiyaçları Üzerindeki Etkisi. *Kent Akademisi*, *17*(1), 60–71.

Dalia Sobhy Rami Bahsoon, L. M. R. K. L. L. M. (2021). Evaluation of Software Architectures under Uncertainty. *ACM Transactions on Software Engineering and Methodology, 30*(4), 1-50.

Erdal, E., & Ergüzen, A. (2020). Nesnelerin İnterneti (IoT). *International Journal of Engineering Research and Development*, *12*(3), 24–34.

Erişen, S. (2021). Creative Evolution in Architecture: A Critical Inquiry Into New Relations of Objectile-Subjectile And Intelligent Spaces. *Middle East Technical University, The Degree of Doctor of Philosophy*.

Ever, D., & Demircioğlu, E. N. (2022). Yapay Zekâ Teknolojilerinin Kalite Maliyetleri Üzerine Etkisi. *Çukurova Üniversitesi Sosyal Bilimler Enstitüsü Dergisi*.

Fox, J. (2017). Cognitive systems at the point of care: The CREDO program. *Journal of Biomedical Informatics*, *68*, 83–95.

Garro, A., Mühlhäuser, M., Tundis, A., Baldoni, M., Baroglio, C., Bergenti, F., Torroni, P., Ranganathan, S., Gribskov, M., & Nakai, K. (2018). Intelligent agents: Multi-agent systems. *Encyclopedia of Bioinformatics and Computational Biology*, *1*, 315–320.

Hassouna, A. Ben, Chaari, H., & Belhaj, I. (2025). LLM-Agent-UMF: LLM-based agent unified modeling framework for seamless design of multi active/passive core-agent architectures. *Information Fusion*, 103865.

Khapre, S., Mersha, M. A., Olalekan, Z., Shakil, H., Iskander, S., Moustafa, A., & Kalita, J. (2026). Agentic AI systems: A systematic survey of multi-agent architectures, cognitive foundations, interaction, explainability, security, and performance evaluation. *Neurocomputing*, 134049.

Lekesiz, G., & Gürer, E. (2022). Mimari Tasarım Stüdyosunun Karmaşık Sistem Kavramsal Çerçevesinde Değerlendirilmesi. *Journal of Computational Design, 3*(2), 1-26.

Li, Y., Choi, S. W., Okamoto, H., Hoxha, B., Fainekos, G., & Tran, H.-D. (2026). Quantitative verification of learning-enabled systems using ProbStar reachability. *Nonlinear Analysis: Hybrid Systems*, *59*, 101623.

Lu, Q., Zhao, D., Liu, Y., Zhang, H., Zhu, L., Xu, X., Shi, A., Tan, T., & Kazman, R. (2025). AgentArcEval: An architecture evaluation method for foundation model based agents. *Journal of Systems and Software*, 112656.

Mandler, G. (2006). *A history of modern experimental psychology: From James and Wundt to cognitive science*. Cambridge: The MIT Press.

Marda, A., Kulkarni, S., & Vaidhyanathan, K. (2024). SWITCH: An Exemplar for Evaluating Self-Adaptive ML-Enabled Systems. *2024 IEEE/ACM 19th Symposium on Software Engineering for Adaptive and Self-Managing Systems (SEAMS)*, 143–149.

Mekki, M., & Ksentini, A. (2025). Resiliency focused proactive lifecycle management for stateful microservices in multi-cluster containerized environments. *Computer Communications*, *236*, 108111.

Özsoy, A., Dursun, P., & Uçkan, B. (2023). The Evolution Narrative of Architectural Form and The State of Vital Form. *Journal of Computational Design, 4*(2), 193-226.

Pambudi, R. U., Shiddiqi, A. M., Ijtihadie, R. M., Amoriza, M. N. A. R., Tee, H., Madany, F. A., Akbar, R. J., & Navastara, D. A. (2026). DiLLeMa: An extensible and scalable framework for distributed large language models (LLMs) inference on multi-GPU clusters. *SoftwareX*, *33*, 102537.

Rodrigues, M. G., Viegas, E. K., Santin, A. O., & Enembreck, F. (2025). A mlops architecture for near real-time distributed stream learning operation deployment. *Journal of Network and Computer Applications*, *238*, 104169.

Sahin, Ö. (2021). *Develop intelligent IOS apps with swift: understand texts, classify sentiments, and autodetect answers in text using NLP.* New York: Apress.

Sobhy, D., Bahsoon, R., Minku, L., & Kazman, R. (2021). Evaluation of Software Architectures under Uncertainty: A Systematic Literature Review. *ACM Trans. Softw. Eng. Methodol.*, *30*(4).

Sobhy, D., Minku, L., Bahsoon, R., & Kazman, R. (2022). Continuous and proactive software architecture evaluation: An IoT case. *ACM Transactions on Software Engineering and Methodology (TOSEM)*, *31*(3), 1–54.

Wong, H. M., & Perumal, S. (2025). AI-Driven Model-Retraining Architecture to Sustain Operational Accuracy in Data-Drifting Environments. *2025 IEEE Symposium on Wireless Technology & Applications (ISWTA)*, 1–6.

Explainable Deep Classification of Urinary Pathogens on Chromogenic Media

Serpil USTEBAY

1- İstanbul University-Cerrahpaşa Faculty of Engineering, Department of Computer Engineering, serpil.ustebay@iuc.edu.tr ORCID No: 0000-0003-0541-0765

ABSTRACT

In laboratories, urine culture evaluations are performed by microbiology specialists. A shortage of specialists, particularly in high-volume clinical laboratories, can lead to significant delays in diagnostic processes. Therefore, integrating AI-powered tools into microbiological workflows helps accelerate culture analysis and reduce the workload of specialists. In this study, urine cultures were analyzed using a deep learning model. To detect relevant bacteria, a YOLO model trained in chromogenic urine culture images containing *Escherichia coli* and *Klebsiella pneumoniae* was employed. The performance of the trained model was evaluated using an independent test dataset obtained from a different domain (cross-domain evaluation). Grad-CAM visualizations were used to interpret the model's behavior. Grad-CAM activation maps demonstrated that the model based its classification decisions on the streaking lines and chromogenic characteristics present in the Petri dish images. This study highlights the critical importance of explainability in ensuring the interpretability and reliability of AI models intended for clinical use, alongside their performance metrics.

Index Terms—Explainable Artificial Intelligence, XAI, Deep Learning, YOLO, Clinical Microbiology, Chromogenic Agar, Urine Culture, Grad-CAM, Medical Image Analysis

1. INTRODUCTION

Urinary tract infections (UTIs) are a major public health problem and among the most common bacterial infections worldwide. Global Burden of Disease 2021 indicates that the number of UTI cases has increased by approximately 66%, and this upward trend is expected to continue until 2050 [1]. UTIs are particularly prevalent among women and elderly men. In addition, antimicrobial resistance associated with urinary tract pathogens has become an increasing global concern. Recent WHO-supported reports emphasize that resistant Gram-negative bacteria, particularly *Escherichia coli (E. coli)* and *Klebsiella pneumoniae (K. pneumoniae)*, remain among the most critical bacterial pathogens worldwide [2].

UTIs are generally diagnosed in clinical microbiology laboratories through urine culture analysis. Urine samples are inoculated onto chromogenic agar plates, and an incubation period of approximately 16–18 hours are required for bacterial colonies to become distinguishable. After incubation, experts identify bacterial colonies based on their color and morphology. However, it typically takes 48–72 hours for culture results to become clinically usable, followed by additional time for antibiotic susceptibility testing [3]. This delay may contribute to increased empirical broad-spectrum antibiotic use and the

progression of antimicrobial resistance. Therefore, there is a growing need for artificial intelligence (AI)-based systems that enable rapid, standardized, and automated analysis of culture images.

Deep learning models have shown promising performance in medical image analysis and clinical microbiology applications. However, in healthcare applications, high classification accuracy alone is not sufficient. Clinical AI systems are also expected to provide interpretable and explainable decision mechanisms. Explainable Artificial Intelligence (XAI) methods can improve clinical reliability by helping experts understand which visual features influence model predictions. In this study, deep learning-based classification of *E. coli* and *K. pneumoniae* was performed using chromogenic agar urine culture plates. In addition, Grad-CAM-based explainability analyses were conducted to investigate the decision-making mechanisms of the developed models.

Another important consideration is whether a classification model can maintain its performance when applied to images collected under different conditions. Differences in illumination, camera positioning, plate orientation, and colony appearance may affect the visual features learned during training. A model that performs well on images from a single source may not achieve the same results when tested on images obtained in another laboratory. For this reason, evaluating model performance on an independent external dataset can provide additional information about its generalizability. In this study, the use of an external dataset, together with Grad-CAM visualizations, allowed us to examine both classification performance and the visual regions that contributed to the model's predictions.

In this study, the classification performance of deep learning models was evaluated using chromogenic agar images of *E. coli* and *K. pneumoniae*. In addition to testing the models on the original dataset, an independent external dataset was used to examine their performance under different laboratory and imaging conditions. Grad-CAM was also applied to visualize the image regions that contributed to the model predictions. By combining classification results with external validation and visual explanations, this study provides a broader assessment of the models' performance and their potential use in the analysis of urine culture images.

2. RELATED WORK

Studies on the automated evaluation of culture plates in clinical microbiology laboratories have increased significantly in recent years. These studies have mainly focused on automating routine laboratory procedures, detecting bacterial growth, and reducing the time required for culture interpretation.

More recently, deep learning methods have also been investigated for bacterial colony detection and species identification. Although these approaches have shown promising results, they differ in their objectives, the types of images they use, and the level at which bacterial identification is performed.

Brenton et al. [4] evaluated the APAS® Independence system for the automated imaging and preliminary interpretation of routine urine cultures. Their study included 3,000 urine samples, and the system achieved sensitivity and specificity values of 0.919 and 0.877, respectively, for detecting significant bacterial growth. The agreement among microbiologists for colony counting was reported as 88%. These findings illustrate the difficulties associated with manual culture interpretation and the potential value of automated systems in routine laboratory workflows. However, detecting significant bacterial growth and identifying the bacterial species responsible for an infection are different tasks. The former helps determine whether a culture requires further examination, whereas the latter provides more specific information about the organism present.

Faron et al. [5] investigated the performance of WASPLab software for the automated reading of urine cultures grown on chromID CPS Elite agar. A total of 1,581 urine cultures were examined, and the reported sensitivity and specificity values were 99.8% and 69.9%, respectively. The software-based approach also reduced reporting time compared with manual evaluation. This study is particularly relevant to the present work because both studies involve urine cultures and chromogenic agar images. However, the main objective of the WASPLab study was to classify cultures as negative or non-negative rather than to distinguish between *E. coli* and *K. pneumoniae*. Species-level classification requires the model to identify visual differences between bacterial colonies, which presents a different challenge from determining whether bacterial growth is present.

The DeepColony system [6] introduced a hierarchical multi-network deep learning architecture for the overall interpretation of urine culture plates and the presumptive identification of pathogens. The study used 26,213 isolated colony images representing 32 bacterial and fungal species. The Top-1 identification accuracy was 83.4% at the single-colony level. After contextual refinement, the accuracy increased to 90.6%. These results suggest that considering the context in which colonies appear can improve identification performance. They also show that bacterial identification can benefit from analyzing information beyond the appearance of an individual colony. Nevertheless, the use of isolated colony images and contextual information differs from the image-level classification approach investigated in the present study, in which each Petri dish image is assigned a single bacterial species label.

Sunanda et al. [7] compared several convolutional neural network architectures, including GoogLeNet, AlexNet, VGG-16, SqueezeNet, and DenseNet-161, using agar plate images containing *Escherichia* coli, *Bacillus subtilis*, *Candida albicans*, *Staphylococcus aureus*, and *Pseudomonas aeruginosa*. GoogLeNet achieved the highest reported performance, with accuracy values ranging from 97.15% to 99.56%. Their findings demonstrated the potential of deep learning models to distinguish between different microorganisms based on culture images. However, the study was not specifically designed for urine culture analysis and did not include *K. pneumoniae*. This distinction is important because the performance of a model on one group of microorganisms does not necessarily reflect its ability to distinguish between other bacterial species with different or potentially similar colony appearances.

Object detection models have also been investigated for bacterial colony analysis. Wang et al. [8] trained a YOLOv8n-based model for bacterial colony detection and reported a precision of 89.4%, a recall of 89.8%, and a mean average precision (mAP) of 91.3%. Their results demonstrated the potential of YOLO-based architecture for detecting bacterial colonies in culture images. Unlike conventional image classification, object detection identifies the locations of individual objects within an image as well as their classes. Although these tasks are related, their evaluation metrics and intended outputs are different. The present study focuses on classifying the bacterial species represented in the entire Petri dish image rather than detecting and localizing individual colonies.

Sarmis et al. [3] reported an accuracy of 0.997, together with high precision, recall, F1-score, and specificity values, using a YOLOv12-based model for bacterial identification from urine culture images. Their findings demonstrated that deep learning can achieve high classification performance for this task. However, classification performance alone does not indicate which image features contribute to a model's predictions. Examining these features is particularly relevant in medical image analysis, where a model may rely on visual patterns that are not directly related to the intended classification task. Explainable artificial intelligence methods, such as Gradient-weighted Class Activation Mapping (Grad-CAM), can help researchers visualize image regions associated with model predictions.

The present study builds on this line of research by evaluating YOLO11 for the classification of *E. coli* and *K. pneumoniae* from chromogenic urine culture images. In addition to measuring classification performance, Grad-CAM was used to examine the image regions associated with the model's predictions. The model was also evaluated on an independent external dataset

containing images acquired under different laboratory and imaging conditions. These analyses provide complementary information about classification performance, visual attention, and generalizability. By considering these aspects together, the study extends the evaluation beyond classification metrics alone and examines the model's behavior on images obtained from a separate data source.

3. MATERIALS AND METHODS

3.1. Dataset

In this study, chromogenic agar urine culture images [3] were used to classification for the identification of *E. coli* and *K. pneumoniae*. The primary dataset consisted of high-resolution images of Petri dishes obtained through the Copan WASP automated microbiology system. In total, 1,547 images were included in the dataset, of which 850 corresponded to *E. coli* and 697 to *K. pneumoniae*.

The images were acquired automatically following approximately 16 hours of incubation. Each image represented a chromogenic agar plate containing bacterial growth, with colony appearance and coloration providing visual information relevant to species identification. However, the appearance of bacterial colonies may vary across individual plates, even when the same species is present. Differences in colony distribution, growth density, color intensity, and the overall appearance of the agar surface can introduce variability into the image classification process. Such variation is particularly relevant when developing an image-based method intended to assist with the interpretation of routine urine culture results.

Each Petri dish image was assigned a single class label by experts according to the bacterial species represented in the culture. Accordingly, the dataset was organized into two classes: *E. coli* and *K. pneumoniae.* Image-level labeling was used throughout the study, with each image treated as one classification sample rather than assigning separate labels to individual colonies. The resulting dataset therefore reflects the task of identifying the bacterial species from the overall appearance of a urine culture plate. The primary dataset was used to assess the model's ability to distinguish between the two classes under the imaging conditions represented in the original collection.

The study was approved by the Ethics Committee of Sancaktepe Şehit Prof. Dr. İlhan Varank Training and Research Hospital (Ethics Committee No: 2024/188).

To further investigate whether the proposed model's performance is generalizable, an independent external dataset [12] was used. This dataset comprised 91 chromogenic agar Petri dish images collected in a previously published study, including 60 images of *E. coli* and 31 images of *K. pneumoniae*. Unlike the primary dataset, the external images were obtained under different laboratory and imaging conditions. Variations were present in Petri dish appearance, illumination, camera positioning, and plate orientation. These differences introduced additional visual variability and provided an opportunity to examine the model's performance on images originating from a separate data source.

3.2. Performance Metrics

The performance of the proposed model was evaluated using accuracy, precision, recall, and F1-score. These metrics were selected to examine different aspects of classification performance and to provide a clearer understanding of the model's strengths and limitations.

Accuracy measures the proportion of images that were correctly classified among all images in the dataset. It provides an overall view of model performance. However, accuracy alone may not be sufficient to evaluate a classification model, particularly when the number of samples differs between classes.

Precision measures how many images predicted as a particular class actually belong to that class. In this study, it indicates how reliably the model identifies E.coli or K.pneumoniae when making a prediction. Recall measures the proportion of images belonging to a given class that the model correctly identifies. A high recall value indicates that fewer samples from that class are missed by the model.

The F1-score combines precision and recall into a single metric by calculating their harmonic mean. It provides a useful measure when both correct positive predictions and the identification of actual positive samples are important. Unlike accuracy, the F1-score considers the balance between precision and recall, making it useful for a more detailed assessment of classification performance.

In addition to these metrics, a confusion matrix was used to examine the model's predictions in more detail. It shows the number of correctly and incorrectly classified images for each class. This analysis makes it possible to identify which bacterial species is more frequently misclassified and whether the model has difficulty distinguishing between *E. coli* and *K. pneumoniae*. The confusion matrix was therefore used alongside the numerical metrics to

provide a more complete assessment of the model's classification performance.

3.3. Deep Learning Model and Explainable Artificial Intelligence Framework

In this study, YOLO11n-cls [10] was used for direct image-level classification of chromogenic agar urine culture images. Unlike object detection-based YOLO approaches, YOLO11n-cls performs classification directly on the whole image without requiring colony-level annotations. The model predicts the bacterial class using visual characteristics of the Petri dish image.

In clinical decision support systems, high classification performance alone is insufficient; models are also expected to provide interpretable and transparent decision-making mechanisms. Therefore, Grad-CAM (Gradient-weighted Class Activation Mapping) [11] was employed as the explainable artificial intelligence (XAI) framework in this study.

Grad-CAM generates visual explanations by combining the gradient information of the target class with activation maps obtained from the final convolutional layer, thereby highlighting image regions that contribute most to the model prediction. In this study, Grad-CAM analyses were performed to identify the discriminative regions influencing bacterial classification decisions.

For XAI evaluation, two types of visual outputs were generated for each sample: original sample and an overlayed image. In overlay images, Grad-CAM heatmap was superimposed semi-transparently onto the original image to facilitate visual interpretation of the activated regions. Furthermore, both correctly and incorrectly classified samples were examined separately to better understand the visual patterns associated with accurate predictions and misclassifications.

4. EXPERIMENTAL RESULTS AND DISCUSSION

All experiments were performed using the Python programming language. Before training, all urine culture images were resized to 224 × 224 pixels to match the input requirements of the model. This step ensured that the images had the same dimensions before being used in the experiments.

Data augmentation was applied during training to increase the variation in the images presented to the model. The augmentation process included random rotations of up to ±90°, random scaling within 5%, and horizontal and vertical flipping. These transformations allowed the model to encounter images in

different orientations and at slightly different scales. This was intended to reduce its dependence on the specific appearance or orientation of the training images and help it learn features that were useful across a wider range of samples. The original images were not limited to a single orientation during training, which may be beneficial when evaluating Petri dish images acquired under different imaging conditions.

4.1. Classification Performance of YOLO11

Table 1 presents the classification performance of the YOLO11 model on the original and external test datasets. The results are reported using four evaluation metrics: accuracy, precision, recall, and F1-score. The original dataset results show the model's performance on images from the primary data collection, whereas the external dataset results provide information about its performance on images obtained under different laboratory and imaging conditions. Presenting both results allows a direct comparison of the model's performance across the two datasets.

On the original test dataset, the model achieved 0.99 for all four evaluation metrics, indicating a high level of classification performance. The results on the external dataset were also strong, with accuracy, precision, recall, and F1-score values of 0.96, 0.96, 0.95, and 0.95, respectively. Among the 91 external test images, 87 were correctly classified, corresponding to an accuracy of 0.956. This value is reported as 0.96 when rounded to two decimal places.

Table 1. Comparison of YOLO11 performance on the original and external test datasets.

Metric	Original Dataset	External Dataset
Accuracy	0.99	0.96
Precision	0.99	0.96
Recall	0.99	0.95
F1-score	0.99	0.95

External test results showed that all 31 *K. pneumoniae* images were correctly classified, whereas four of the 60 Escherichia coli images were misclassified as *K. pneumoniae*. These results indicate that the model correctly identified all *K. pneumoniae* samples in this dataset, although some *E. coli* samples were more difficult to distinguish. The external images had been collected under different laboratory and imaging conditions, including variations in Petri dish appearance, illumination, camera positioning, and plate orientation. Despite these differences, the model maintained high classification performance.

4.2. Grad-CAM Analysis of Correctly Classified Samples

Figures 1 and 2 show Grad-CAM heatmaps of correctly classified samples from the original and external test datasets. In both figures, the first row shows an *E. coli* sample and the second row shows a *K. pneumoniae* sample. Red and yellow regions indicate high contribution, whereas blue regions represent low contribution. Activation maps are mainly concentrated on colony-rich regions and inoculation streaks. Only limited activation is observed outside the culture plates.

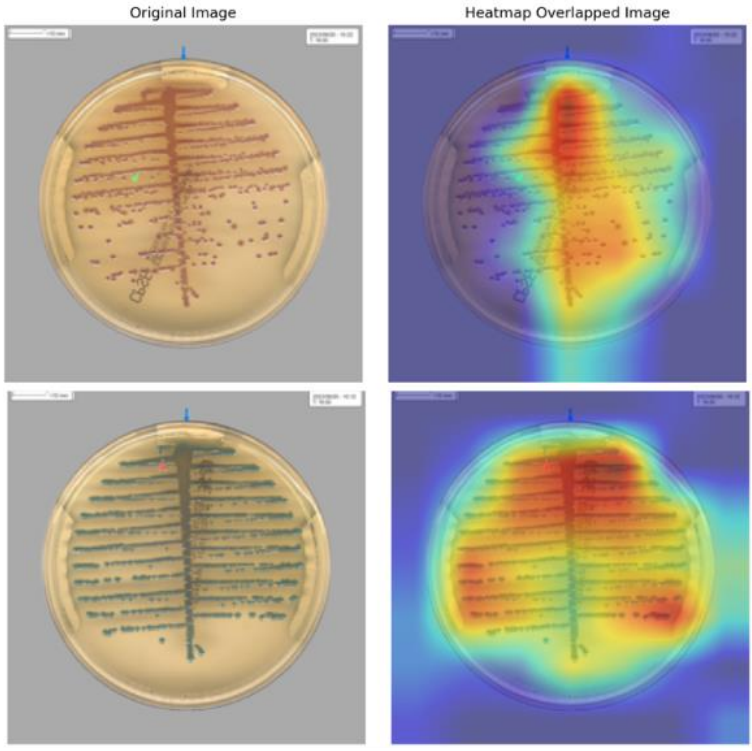

Fig. 1. Grad-CAM heatmaps of correctly classified urine culture samples generated by YOLO11.

The external test images were collected under different imaging conditions. Despite these differences, the model remained focused on the main colony regions and inoculation streaks.

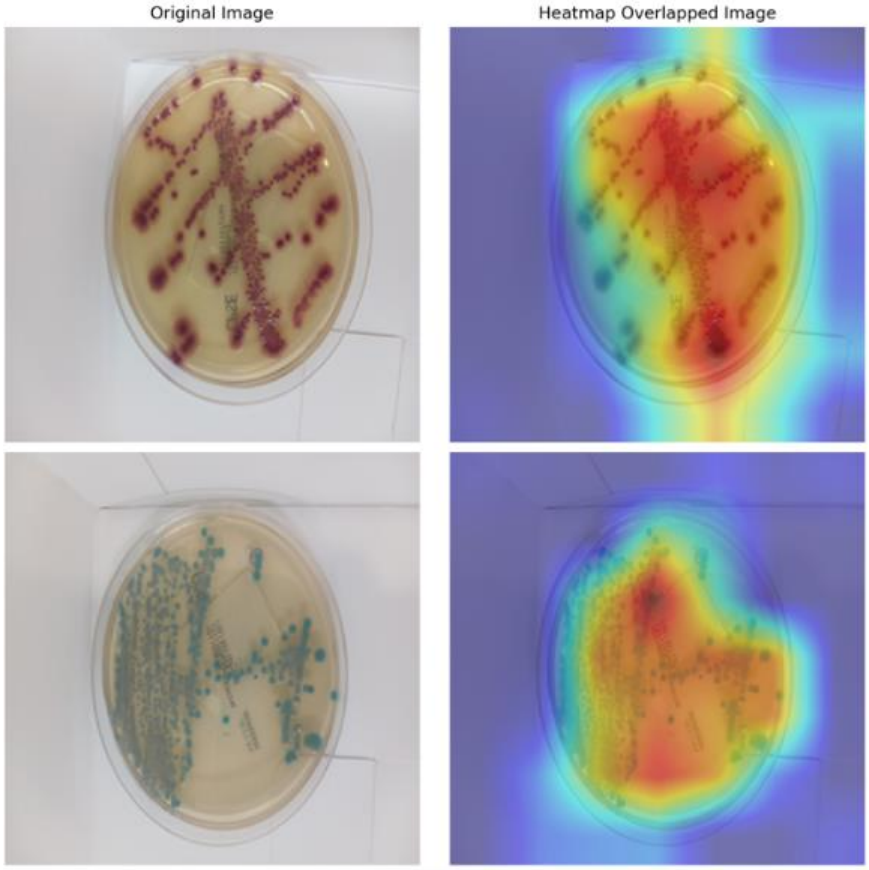

Fig. 2. Grad-CAM heatmaps of correctly classified samples from the external test dataset.

In the Copan WASP automation system, urine samples are distributed over the agar surface using predefined streak patterns. These regions usually contain dense colony growth and important visual characteristics. Their location may change because of plate rotation or image orientation. Despite these differences, the model consistently attended to the main inoculation areas. The activation maps indicate that the classification process was primarily based on microbiologically relevant regions of the culture plates.

4.3. Grad-CAM Analysis of Misclassified Samples

Figure 3 shows a misclassified urine culture sample. The true class is *E. coli*, but the model predicted it as *K. pneumoniae* with a confidence score of 0.72.

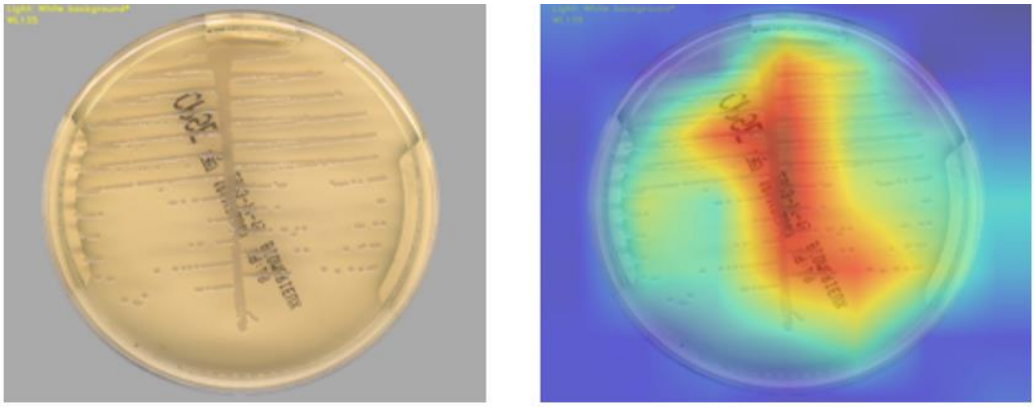

Fig. 3. Grad-CAM analysis of a misclassified urine culture sample. The true class is *E. coli*, and the predicted class is *K. pneumoniae*.

The activation map is mainly concentrated on dense colony regions and inoculation streaks. Strong activations are observed around atypical dark-golden colony structures. These regions may share visual characteristics with patterns associated with *K. pneumoniae*. On chromogenic agar plates, *E. coli* colonies are usually expected to appear in pink-burgundy tones. Atypical colony appearances can change the visual characteristics of the culture plate and may influence model predictions.

Only limited activation is observed outside the culture plate. The model remains focused on colony-rich regions instead of background structures. This suggests that the prediction was based on microbiologically relevant image features.

Figure 4 shows a misclassified sample from the external test dataset. The true class is *E. coli*, but the model predicted it as *K. pneumoniae*. Similar activation patterns are observed under different imaging conditions. The model remains focused on colony-rich regions and inoculation streaks, and only limited activation is detected outside the culture plate. Differences in colony appearance and image acquisition conditions may have influenced the prediction.

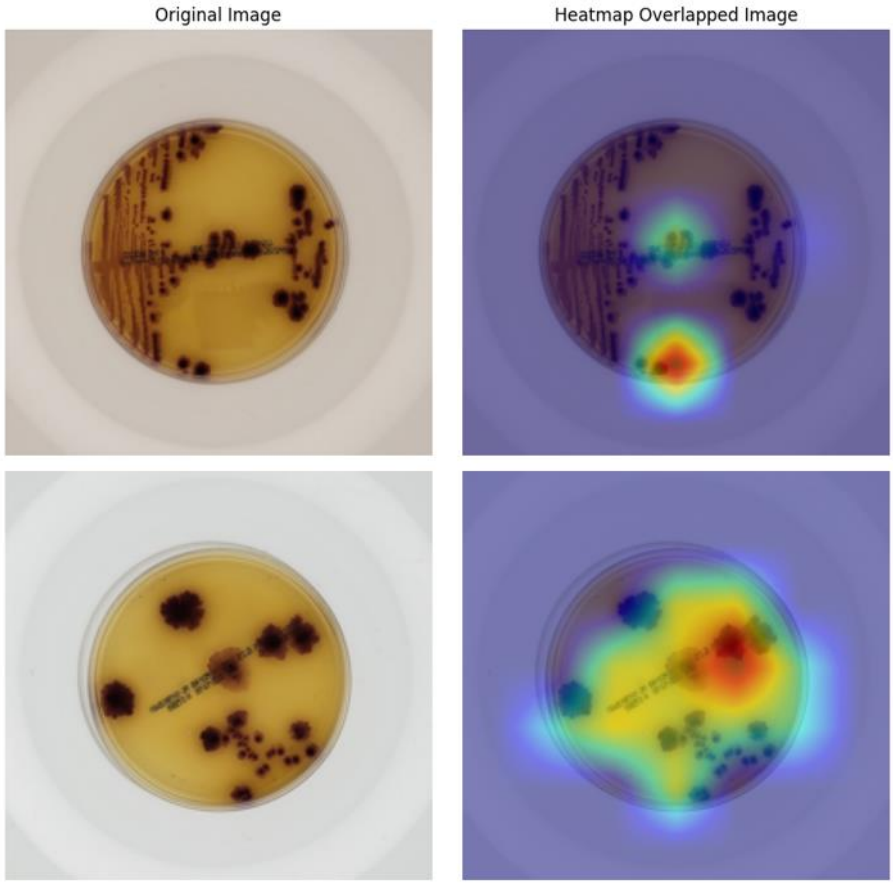

Fig. 4. Grad-CAM analysis of a misclassified sample from the external dataset. The true class is *E. coli*, and the predicted class is *K. pneumoniae*.

4.4. Potential Clinical Risks and Patient Safety Considerations

In this study, bacterial classification was mainly based on colony morphology and chromogenic color distributions observed on urine culture plates. Although *E. coli* and *K. pneumoniae* usually show characteristic color patterns, clinical microbiology laboratories may also encounter atypical variants and visually similar bacterial species. Non-lactose fermenting variants, weak pigment formation, mixed cultures, or low-density colony growth may change the appearance of culture plates. Such variations may increase risk of misclassification in AI-based systems that rely on visual features. Similar limitations have also been reported in previous automated urine culture studies [6].

Grad-CAM analysis of misclassified samples was consistent with these observations. In first example, strong activations were observed around atypical golden-yellow colony regions, whereas commonly reported appearance of *E. coli* on chromogenic media is pink-burgundy. In external test sample, activation pattern was distributed across multiple colony-rich regions. Activation maps remained concentrated on microbiologically meaningful regions. These observations indicate that atypical colony morphology and imaging variability may influence model predictions.

Image acquisition conditions are another important factor. All images used in this study were obtained using Copan WASP automation system. Different imaging conditions may change appearance of culture plates. Camera settings and incubation duration may alter visual characteristics of colonies. Agar production differences may also affect color distribution. These factors may differ between laboratories and imaging systems. Such differences can create domain shifts and reduce model performance [14]. Decrease in accuracy from 0.99 to 0.95 on external dataset (Table 1) is consistent with this observation. These results indicate that evaluating AI systems using data collected from a single laboratory may not fully reflect real-world clinical variability. Multi-center datasets collected under different laboratory conditions may improve robustness and generalizability of proposed system [15].

Patient safety is another important consideration for AI-assisted microbiology systems. Incorrect bacterial classification may influence preliminary laboratory interpretation and may indirectly affect empirical antibiotic selection. Delayed or inappropriate treatment may contribute to antimicrobial resistance and may negatively affect patient outcomes. Proposed system should not be considered a fully autonomous diagnostic tool. It should be used as a decision support system for expert microbiologists. Human supervision during final evaluation stage is considered a safer approach [16].

Understanding how AI systems make decisions is important for clinical use. Visual explanation methods help experts examine image regions associated with model predictions. In this study, Grad-CAM analysis showed that model mainly focused on colony-rich regions and inoculation streaks. Even in misclassified samples, activation maps remained concentrated on microbiologically relevant structures. This information may help experts interpret model behavior [17].

5. CONCLUSION

In this study, YOLO11 was used to classify *E. coli* and *K. pneumoniae* from chromogenic urine culture images. The model achieved high classification performance on both the original and external test datasets. These results

indicate that deep learning can be useful for distinguishing these two bacterial species based on the visual appearance of culture plates. The evaluation on the external dataset also provided information about the model's performance on images collected under different laboratories and imaging conditions.

Grad-CAM analysis showed that the model mainly focused on inoculation streaks, colony morphology, and the chromogenic characteristics of the culture plates. These findings provided insight into the image regions associated with the model's predictions. However, the evaluation of misclassified samples showed that atypical colony appearances and variations in image acquisition may affect classification results. This highlights the importance of considering image variability when developing and evaluating AI-based methods for bacterial identification.

The external dataset results further demonstrated the importance of testing models on images obtained from sources other than the original data collection. Although the model maintained high performance, some *E. coli* samples were incorrectly classified as *K. pneumoniae*. These errors indicate that certain culture appearances may be difficult for the model to distinguish and that further evaluation is needed to understand the factors contributing to misclassification.

Grad-CAM provided a useful way to examine the regions associated with the model's predictions, while external testing offered additional information about its performance under different imaging conditions. The proposed approach should be considered a decision support tool for expert microbiologists rather than a fully autonomous diagnostic system. Further studies should include larger, multicenter datasets collected under a wider range of laboratory conditions. The inclusion of more diverse bacterial isolates and additional sources of image variability would also help determine whether the model can maintain its performance in routine clinical practice. These studies are needed before the approach can be considered for broader clinical use.

REFERENCES

[1]. Y. He, J. Zhao, L. Wang, et al., “Epidemiological trends and predictions of urinary tract infections in the global burden of disease study 2021,” *Scientific Reports*, vol. 15, Art. no. 4702, 2025, doi:10.1038/s41598-025-89240-5.

[2]. H. Sati, E. Carrara, A. Savoldi, et al., “The WHO Bacterial Priority Pathogens List 2024: A prioritisation study to guide research, development, and public health strategies against antimicrobial resistance,” *The Lancet Infectious Diseases*, vol. 25, no. 9, pp. 1033–1043, 2025, doi:10.1016/S1473-3099(25)00118-5.

[3]. A. Sarmis, S. Ustebay, M. A. Mutlu, F. A. Canbaz, and G. K. Kaya, “Deep Learning-Based Rapid Identification of Escherichia coli and Klebsiella pneumoniae from Chromogenic Agar Urine Cultures Using YOLOv12,” *Risk Management and Healthcare Policy*, vol. 19, Art. no. 561761, 2026, doi:10.2147/RMHP.S561761.

[4]. L. Brenton, M. J. Waters, T. Stanford, and S. Giglio, “Clinical evaluation of the APAS® Independence: Automated imaging and interpretation of urine cultures using artificial intelligence with composite reference standard discrepant resolution,” *Journal of Microbiological Methods*, vol. 177, Art. no. 106047, 2020, doi:10.1016/j.mimet.2020.106047.

[5]. M. L. Faron, B. W. Buchan, H. Samra, and N. A. Ledeboer, “Evaluation of WASPLab software to automatically read chromID CPS Elite agar for reporting of urine cultures,” *Journal of Clinical Microbiology*, vol. 58, no. 1, Art. no. e00540-19, 2020, doi:10.1128/JCM.00540-19.

[6]. A. Signoroni, A. Ferrari, S. Lombardi, M. Savardi, S. Fontana, and K. Culbreath, “Hierarchical AI enables global interpretation of culture plates in the era of digital microbiology,” *Nature Communications*, vol. 14, Art. no. 6874, 2023, doi:10.1038/s41467-023-42563-1.

[7]. Inchara, B. K., Disha, V., Divya Shree, K., & GC, S. M. (2024, April). Classification of bacteria from agar plate using deep learning and image processing. In *2024 International Conference on Advances in Data Engineering and Intelligent Computing Systems (ADICS)* (pp. 1-6). IEEE.

[8]. M. Wang, J. Luo, K. Lin, Y. Chen, X. Huang, J. Liu, A. Wang, and D. Xiao, “Colony-YOLO: A lightweight micro-colony detection network based on improved YOLOv8n,” *Microorganisms*, vol. 13, no. 7, Art. no. 1617, 2025, doi:10.3390/microorganisms13071617.

[9]. K. He, X. Zhang, S. Ren, and J. Sun, “Deep residual learning for image recognition,” in *Proc. IEEE Conf. Computer Vision and Pattern Recognition (CVPR)*, 2016, pp. 770–778, doi:10.1109/CVPR.2016.90.

[10]. Ultralytics, “YOLO11 Documentation,” 2024. [Online]. Available: https://docs.ultralytics.com/models/yolo11/. Accessed: May 18, 2026.

[11]. R. R. Selvaraju, M. Cogswell, A. Das, R. Vedantam, D. Parikh, and D. Batra, “Grad-CAM: Visual explanations from deep networks via gradient-based localization,” *International Journal of Computer Vision*, vol. 128, no. 2, pp. 336–359, 2020, doi:10.1007/s11263-019-01228-7.

[12]. G. R. da Silva, I. B. Rosmaninho, E. Zancul, et al., “Image dataset of urine test results on petri dishes for deep learning classification,” *Data in Brief*, vol. 47, p. 109034, 2023. doi:10.1016/j.dib.2023.109034

[13]. J. Wiens, S. Saria, M. Sendak, et al., “Do no harm: A roadmap for responsible machine learning for health care,” *Nature Medicine*, vol. 25, no. 9, pp. 1337–1340, 2019, doi:10.1038/s41591-019-0548-6.

[14]. C. J. Kelly, A. Karthikesalingam, M. Suleyman, G. Corrado, and D. King, “Key challenges for delivering clinical impact with artificial intelligence,” *BMC Medicine*, vol. 17, no. 1, Art. no. 195, 2019, doi:10.1186/s12916-019-1426-2.

[15]. A. Barredo Arrieta, N. Díaz-Rodríguez, J. Del Ser, et al., “Explainable Artificial Intelligence (XAI): Concepts, taxonomies, opportunities and challenges toward responsible AI,” *Information Fusion*, vol. 58, pp. 82–115, 2020, doi:10.1016/j.inffus.2019.12.012.

[16]. E. J. Topol, “High-performance medicine: The convergence of human and artificial intelligence,” *Nature Medicine*, vol. 25, no. 1, pp. 44–56, 2019, doi:10.1038/s41591-018-0300-7.
[17]. A. Holzinger, C. Biemann, C. S. Pattichis, and D. B. Kell, “What do we need to build explainable AI systems for the medical domain?” *arXiv preprint arXiv:1712.09923*, 2017.

Modern Cryptographic Communication Algorithms and Secure Strategies

Veli AKAY[1]
Muhammet ATEŞ[2]

[1] Van Yüzüncü Yıl Üniversitesi, Van Meslek Yüksekokulu, Bilgisayar Teknolojileri Bölümü ORCID: 0000-0001-7347-5742

[2] Van Yüzüncü Yıl Üniversitesi, Van Meslek Yüksekokulu, Elektronik Teknolojileri Bölümü ORCID: 0000-0003-2223-2745

1. INTRODUCTION AND BASIC CONCEPTS

1.1. Digitalization and the Need for Data Security

Today, rapid and continuous advancements in information and communication technologies have led to individual, societal, and organizational activities increasingly shifting to digital environments and integrating with digital infrastructures. With the process of digital transformation, high-importance information assets—such as critical infrastructure, financial systems, healthcare services, public services, industrial control systems, and personal data—have become dependent on intensive data generation, transmission, and storage processes in cyberspace. This situation necessitates not only that information be accessible and processable, but also that it be protected against threats such as unauthorized access, alteration, disclosure, and destruction.

The increasing volume and variety of data exchanged in digital environments are leading to a diversification and increasing complexity of threats to data security. Unauthorised access, data manipulation, identity theft, malware attacks and denial-of-service (DoS) attacks are among the principal cyber security issues that directly threaten the reliability and continuity of information systems. In particular, the compromise or alteration of critical data by attackers can result not only in data loss at both individual and organisational levels, but also in economic, legal and social consequences [2, 3].

In this context, ensuring data security in modern information systems necessitates the implementation of comprehensive security mechanisms aimed at safeguarding the confidentiality, integrity and availability of data. Cryptographic methods, secure communication protocols, access control mechanisms and encryption algorithms based on mathematical principles play a significant role in meeting these security requirements [4]. Consequently, the increasing prevalence of digitalisation is driving up the need for reliable data transmission and secure data storage; this, in turn, has made the development of robust, mathematically analysable security mechanisms—whose resilience against various attack models can be demonstrated—one of the fundamental research areas in modern information security.

1.2. Cyber Threat Landscape: MITM and Data Eavesdropping

Data transfers conducted over open and insecure networks can be significantly vulnerable to various cyberattacks that may occur at different stages of the communication process. Particularly in scenarios where the communication channel is insecure or can be controlled by an attacker, the monitoring, interception, or alteration of data packets by unauthorized parties poses a

serious security risk. For this reason, eavesdropping can be defined as the act of unauthorized third parties passively monitoring the data flow occurring on a communication channel to obtain the transmitted information. The fact that data obtained by an attacker may include not only user credentials but also session information, financial data, or information critical to the organization makes eavesdropping attacks a significant threat, particularly for sensitive information systems [5].

A Man-in-the-Middle (MITM) attack occurs when an attacker not only silently monitors the communication but also intervenes directly in the process. By secretly inserting themselves between the two communicating parties, the attacker gives the impression that the parties are communicating directly with one another [6]. In this way, the attacker can monitor data in real time, alter its content, redirect the data flow, or send fake messages on behalf of one of the parties. This demonstrates that MITM attacks directly threaten not only the confidentiality of information but also the integrity of the data and the security of the parties' identities.

At the heart of MITM attacks lies the fact that at least one of the parties cannot be certain whether the person they are communicating with is actually who they claim to be. For this reason, simply encrypting data is not sufficient to establish secure communication. Whilst encryption ensures confidentiality by making it difficult for others to read messages, it is not sufficient on its own to verify the identity of the other party or to determine whether the message has been altered in transit. Consequently, to ensure complete security, the principles of authentication, confidentiality and data integrity must all be applied together [7].

Consequently, the design of a communication architecture resilient to threats such as MITM and eavesdropping requires the adoption of a holistic security model based on the assumption that the communication channel is untrustworthy, rather than a one-dimensional security approach relying solely on data encryption. At this point, the combined use of various security components—such as robust cryptographic algorithms, secure key exchange mechanisms, digital signature schemes, message authentication codes and authentication protocols—becomes crucial. In particular, the analysis of the mathematical foundations of cryptographic methods and their resilience to attacks constitutes a critical area of research for ensuring secure data communication [1, 8].

2. FUNDAMENTAL PRINCIPLES AND MATHEMATICAL FOUNDATIONS OF CRYPTOGRAPHY

2.1. The Pillars of Information Security (CIA Triad)

Information security refers to a multi-dimensional security approach aimed at protecting information assets against threats such as unauthorised access, tampering, loss or disruption. One of the primary objectives of modern cryptographic systems is to support the principles of integrity, confidentiality and availability, which form the fundamental requirements of information security. In the literature, these three fundamental security elements are referred to as the CIA Triad, an acronym formed from the first letters of the English terms 'Confidentiality', 'Integrity' and 'Availability' [1, 9].

- Confidentiality: Confidentiality refers to the principle that information and data should only be accessible to and comprehensible by authorised users or systems. In particular, the acquisition of sensitive or critical data by unauthorised parties poses a serious threat to information security. Cryptographic encryption methods play a fundamental role in ensuring the principle of confidentiality by transforming data in plain text into an encrypted form that can only be decrypted by authorised personnel possessing the appropriate key[10, 11].

- Integrity: Integrity refers to ensuring that, throughout the process from the creation of the data to its delivery to the recipient or its storage, the data has not been altered, deleted or had unauthorised additions made to it by unauthorised persons. The protection of a piece of data's confidentiality does not imply that the data in question has not been altered. Therefore, in secure communication systems, it is not only necessary to conceal the data, but also to verify that its original integrity has been preserved. Cryptographic hash functions, message authentication codes (MACs) and digital signature mechanisms are among the primary methods used to ensure data integrity [12, 13].

- Availability: Availability refers to authorised users being able to access the information and system resources they require when needed. For an information system to be considered secure, it is not sufficient for data to be merely confidential and have its integrity preserved; it must also be accessible to authorised users when required. Distributed Denial of Service (DDoS) attacks, which target availability by causing system resources to become overloaded, can be cited as an example. Therefore, safeguarding availability requires the combined application of various technical approaches, such as

redundancy, fault tolerance, resource management and attack prevention mechanisms [1, 2].

These three components of the CIA Triad should not be assessed independently of one another but should be treated as complementary elements of information security. For example, encrypting data so that it cannot be accessed by unauthorised persons ensures confidentiality, whilst verifying that it has not been altered during transmission fulfils the principle of integrity; and ensuring that authorised users can access the data in question when they need it fulfils the principle of availability. Consequently, in the design of a secure information system, these three security objectives must be assessed collectively and the system must be supported by security mechanisms appropriate to its threat model [1, 10].

In addition to the CIA Triad, the concepts of authentication and non-repudiation are also considered important security objectives, particularly in cryptographic communication protocols. Authentication aims to verify that the identity claimed by a party participating in the communication is indeed their own. This mechanism plays a critical role, particularly in preventing man-in-the-middle (MITM) attacks. Non-repudiation, on the other hand, is a security feature that ensures a party cannot reasonably deny having carried out a transaction or sent a message after the fact. Digital signature mechanisms, in particular, are used as an important cryptographic tool in ensuring the properties of authentication and non-repudiation [10, 8].

Within this framework, whilst confidentiality, integrity and availability form the fundamental triad of information security, authentication and non-repudiation can be regarded as key security objectives that complement this triad. Addressing these security objectives collectively ensures that modern cryptographic systems move beyond merely encrypting data to create trustworthy and verifiable communication environments [1, 10].

2.2. Kerckhoffs's Principle and Key Management

According to Kerckhoffs' Principle, the fundamental paradigm of cryptography: "The security of a cryptographic system must depend not on the secrecy of the algorithm used, but solely on the secrecy of the key." The fact that algorithms are publicly available allows the system to be tested by the global academic community and to mature through cryptanalytic processes. In this context, key management—the sole component of security—is the most critical discipline, encompassing the processes of secure key generation, distribution, storage and destruction [10, 14].

3. SYMMETRIC CRYPTOGRAPHIC ALGORITHMS

3.1. Block Encryption and the AES Standard

Symmetric cryptography refers to an encryption approach in which the same secret key is used both to encrypt data and to decrypt it. In this type of encryption to ensure secure communication, the parties must either share the secret key to be used in advance via a secure channel or generate it using a secure key exchange method. Symmetric algorithms are predominantly used in modern information security applications, particularly for the rapid and cost-effective processing of large volumes of data [14, 15].

Symmetric encryption is generally divided into two categories block ciphers and stream ciphers. Block ciphers divide the plaintext into blocks of a specific size and then begin processing. Each segment is encrypted using a secret key and various mathematical operations. Techniques such as substitution, permutation and bit-level operations are used in combination during this process. The primary aim here is to completely conceal the statistical information of the original text, whilst making it difficult for attackers to extract any meaningful information from the encrypted data [14, 15].

AES (Advanced Encryption Standard) is undoubtedly one of the most widely used symmetric encryption standards today, having successfully passed security tests. Approved by NIST (National Institute of Standards and Technology) in 2001 under the FIPS 197 standard for use in federal systems, AES is actually based on the algorithm designed by the Belgian cryptographers Joan Daemen and Vincent Rijmen, which was known as ‘Rijndael’ during the competition process [15].

AES always operates on 128-bit data blocks and supports keys of 128, 192, or 256 bits. The number of rounds is determined by the selected key length: AES-128 uses 10, AES-192 uses 12, and AES-256 uses 14. Longer keys increase the search space available to an attacker attempting exhaustive key search, although overall security also depends on how the algorithm is used [15].

The security level of AES does not depend solely on the key length used, but is also closely related to the correct configuration of the algorithm and the selection of an appropriate operating mode. In particular, the direct and incorrect use of block cipher algorithms can lead to various attacks, irrespective of the algorithm’s mathematical security. For this reason, AES should be considered not merely as a standalone encryption algorithm, but in conjunction with an operating mode suitable for the usage scenario and secure key management mechanisms [15, 16].

Consequently, AES has become one of the fundamental components of modern cryptographic systems due to its high performance, support for different key lengths, and its ability to maintain strong security properties despite years of academic and cryptanalytic scrutiny. However, the use of a strong block cipher alone is not sufficient to establish a secure communication system. The secure generation and management of keys, the selection of an appropriate encryption mode, and the support of communication through integrity and authentication mechanisms are also of critical importance in determining the system's overall security level [15].

3.2. The Mathematical Operation of AES

AES has an iterative block cipher structure, and some of the algorithm's fundamental operations are carried out over the finite Galois field GF(2^8), which has 256 elements. As each data byte is represented by 8 bits in AES, the 256 distinct byte values are treated as elements of this finite field. In particular, GF(2^8) arithmetic plays a significant role in the mathematical underpinnings of the SubBytes and MixColumns operations. In this field, addition corresponds to the XOR operation, whilst multiplication is performed using a specific irreducible polynomial [15].

The number of rounds applied in AES varies depending on the key length used. Ten rounds are applied for a 128-bit key, twelve for a 192-bit key and fourteen for a 256-bit key. In each round, the operations SubBytes, ShiftRows, MixColumns and AddRoundKey are essentially carried out. However, the MixColumns operation is not applied in the final round of AES [15].

- SubBytes (Byte Substitution): Each byte in the state matrix is substituted with another byte value using an AES-specific non-linear S-box. The non-linear structure provided by the S-box plays a significant role in AES's resistance to attacks such as differential and linear cryptanalysis [15].

- ShiftRows (Row Shifting): The rows in the state matrix are shifted circularly by varying amounts. This operation prevents bytes from remaining solely within their own column, thereby contributing to the spreading of data across the block and increasing diffusion [15].

- MixColumns (Column Mixing): Each column is transformed by multiplying it by a matrix defined over the finite field GF(2^8). This linear transformation enhances diffusion by increasing the interaction between bytes within the same column. The MixColumns operation is not applied in the final round of AES [15].

- AddRoundKey (Add Round Key): The round key generated for the relevant round is XORed with the state matrix. This ensures that the transformations in the encryption process are linked to the secret key [15].

3.3. Operational Modes and Performance

Various operating modes have been developed to optimise data integrity and security in block ciphers:

- CBC (Cipher Block Chaining): Each block is subjected to an XOR operation with the previous ciphertext block before being encrypted [16].
- CTR (Counter): It encrypts a counter value and combines it with the plaintext; this is the mode best suited to parallel processing [16].
- GCM (Galois/Counter Mode): This is an authenticated encryption (AEAD) mode that provides both confidentiality and integrity verification. TLS 1.3 supports AEAD algorithms such as AES-GCM and ChaCha20-Poly1305; therefore, rather than stating that GCM is the 'most common' or the sole choice in all cases, it is more accurate to note that its use depends on the protocol and application configuration [8, 16]

4. CRYPTOGRAPHIC HASH FUNCTIONS

4.1. Characteristic Properties of Hash Functions

For cryptographic hash functions to be considered secure, it is not only necessary for them to produce an output of fixed length, but also for them to possess certain security properties against various types of attack. In a secure hash function, preimage resistance, second preimage resistance and collision resistance stand out as fundamental security properties. In addition, the avalanche effect is an important design property that describes the nature of the change in the hash value caused by a small change in the input [12, 14, 17]

- One-way property (Preimage Resistance): This refers to the fact that, given a hash value (h), it is computationally impractical to find any input (m) that produces this value. In other words, whilst the forward computation of the hash function is straightforward, deriving the original input solely from the hash value must be extremely difficult [14].
- Second Preimage Resistance: This refers to the fact that, given a specific input (m), it is computationally difficult to find a

different second input $m' \neq m$ such that $(H(m') = H(m))$. In other words, an attacker should not be able to construct another message that produces the same digest as a specific message they possess [14].

- Collision Resistance: This refers to the fact that it is computationally impractical to find a pair of inputs $m_1 \neq m_2$ such that $H(m_1) = H(m_2)$. As the output space of hash functions is finite, the existence of collisions is theoretically inevitable. Collision resistance aims not so much to prevent the existence of such collisions as to make it computationally difficult for an attacker to find a suitable collision within a reasonable time [14].
- The avalanche effect refers to the phenomenon whereby a minute change in the input data alters a significant portion of the hash value. In an ideal system, a single-bit change should result in the output changing by, on average, half its value. This ensures the integrity of the cryptographic structure by eliminating any predictability between inputs and outputs. [14].

It thus becomes clear that the primary purpose of cryptographic hash functions is not merely to shorten data. The key issue is that this transformation is irreversible, and it is practically almost impossible to find different inputs that would produce the same hash value. For this reason, hash functions serve as an indispensable security component in many fields, ranging from data integrity protection to digital signatures, and from secure password storage to authentication systems[14].

4.2. Current Standards

The security properties and resistance to attacks of cryptographic hash functions have been reassessed in light of cryptanalytic research carried out over time. During this process, serious security vulnerabilities have been identified in some widely used legacy hash algorithms, such as MD5 and SHA-1, and it has become apparent that their resistance to collision attacks, in particular, does not meet today's security requirements. It is therefore recommended that stronger and more up-to-date standards be used in the design of new systems, rather than MD5 and SHA-1 [19, 20].

Two major standardized families are SHA-2 and SHA-3, the latter being derived from Keccak. Each family provides four digest sizes, from 224 to 512 bits, identified by the suffix in the function name (for example, SHA-256 and SHA3-256). These functions are used in integrity verification, digital signatures, and cryptographic protocols [12, 18]. SHA-3 is structurally distinct from SHA-2 because it uses a sponge construction [18]. In password storage applications, it is preferable to use specialised algorithms designed for this purpose, where the computational cost can be increased in a controlled manner [14].

Consequently, the SHA-2 and SHA-3 families hold an important position as reliable hash functions in current cryptographic applications. The different structures and properties of these algorithms allow the selection of an appropriate hash function depending on the application's security requirements [12, 18].

5. ASYMMETRIC CRYPTOGRAPHIC ALGORITHMS

5.1. Principles of Public-Key Cryptography

Asymmetric cryptography, also known as public-key cryptography, is a cryptographic approach developed to provide a solution to the problems of secure key distribution and key sharing encountered in symmetric cryptography [14, 10]. Whilst in symmetric cryptography both parties must possess the same secret key for secure communication to take place, in asymmetric cryptography each user employs a key pair consisting of two distinct keys that are mathematically related to one another.

This key pair consists of two components: the Public Key and the Private Key. As the name suggests, the public key can be shared with other users and does not need to be kept secret. In contrast, the private key must be safeguarded solely by the owner of the key pair and kept secret to prevent unauthorised access. Whilst the mathematical relationship between the public and private keys forms the basis of asymmetric cryptography, the fact that the private key cannot be derived from the public key by any practical computational method is a key element of the system's security [14, 10].

In asymmetric cryptography, the purpose for which the keys are used may vary depending on the operation being performed and the algorithm employed. For example, data that a user wishes only they themselves to be able to read can be encrypted using the recipient's public key, and this data can be decrypted using the recipient's private key. This eliminates the need for the parties to share a common secret key via a secure channel before communication takes place [14].

Another important area of application for public-key cryptography is authentication and digital signatures. In this framework, the private key is regarded as a cryptographic tool that can only be used by its owner, whilst the corresponding public key can be used to verify the signature. This makes it possible to verify that a message was created by the owner of a specific private key and to check whether the message has been altered subsequently [14].

However, in asymmetric cryptography, security does not stem directly from the existence of public and private keys, but rather from the mathematical relationship between these keys and the computational security of the mathematical problem employed. Methods such as RSA, Elliptic Curve Cryptography (ECC) and Diffie-Hellman on Elliptic Curves (ECDH) can be cited as examples of different applications of public-key cryptography. For this reason, public-key cryptography has become one of the fundamental components of modern information security applications, such as secure key exchange, authentication, digital signatures and secure communication. In particular, the combined use of symmetric and asymmetric cryptographic methods offers an effective approach in terms of both performance and security in modern secure communication protocols [14]

5.2. RSA Algorithm

The security of RSA, a public-key algorithm, has traditionally been attributed to the infeasibility of factoring the product of two large prime numbers. The public key may be distributed, while the private key must be protected. RSA key generation establishes the mathematical relationships used for encryption or signature operations [14, 18].

The RSA key generation process essentially consists of the following steps [18].

1. Two large prime numbers are selected.

First, two large prime numbers that are distinct from one another are selected. These numbers are denoted by (p) and (q):

The selected values of (p) and (q) are kept secret for the security of the RSA system. Selecting these two prime numbers so that they are secure and of an appropriate size is of great importance for the overall security of the algorithm [18].

2. The RSA modulus (n) is calculated.

The RSA modulus is obtained by multiplying two chosen prime numbers:

$$n = p \times q \tag{1}$$

Here, (n) is the fundamental parameter used in the generation of RSA keys. Whilst (p) and (q) are kept secret, (n) can be shared as part of the public key. [18]

3. The Euler totient function is calculated.

Since (p) and (q) are prime, the Euler totient function is calculated as follows:

$$\varphi(n) = (p - 1)(q - 1) \quad (2)$$

Here, φ(n) denotes the number of positive integers less than n and coprime to n. In RSA, this value is used in the calculation of the private key component d [18].

4. Select a prime base (e)

Next, a prime base, denoted by (e), is selected. The value of (e) must satisfy the following conditions:

$$1 < e < \varphi(n), \ \gcd(e, \varphi(n)) = 1 \quad (3)$$

GCD denotes the greatest common divisor of two numbers. This condition specifies that the only common divisor of e and φ(n) must be 1.

One of the open exponent values commonly used in practice is given below:

$$e = 65{,}537 \quad (4)$$

5. The special exponent (d) is calculated

In the next stage, the special exponent, denoted by (d), is calculated. (d) is determined such that (d) × (e) is the modular multiplicative inverse of (e) modulo (φ(n)). The mathematical notation for this is [18]:

$$d \times e \equiv 1 \ (\text{mod } \varphi(n)) \quad (5)$$

This congruence expresses the defining condition for d:

Equivalently, multiplying d by e must leave a remainder of 1 when divided by φ(n).

The symbols "≡" and "mod" describe congruence in modular arithmetic, rather than ordinary equality or division.

6. Form the public and private keys

Once the required values have been calculated, the RSA key pair is represented as follows.

$$(e, n) \quad (6)$$

The private key is represented by:

$$(d, n) \quad (7)$$

The public key (e, n) can be shared. The private key (d, n) must be kept confidential; p and q must also be protected because knowledge of the factors would compromise the system [10, 21].

5.2.1. RSA Encryption and Decryption

RSA's basic mathematical operations can be expressed using a message represented as an integer m, where $0 \leq m < n$. Encryption with the public exponent is defined as follows [21]:

$$c \equiv m^e \pmod{n} \tag{8}$$

The symbols in the encryption expression denote:

(m): plaintext message,

(e): the exponent value of the public key,

(n): the RSA modulus,

(c): the resulting ciphertext.

Decryption uses the private exponent d:

$$m \equiv c^d \pmod{n} \tag{9}$$

The recipient recovers m from ciphertext c using the private key. This relationship works because the public and private exponents are constructed to satisfy the required modular arithmetic conditions involving $\varphi(n)$.

The equations above describe textbook RSA, which should not be used directly in deployed systems. Practical implementations require standardized encoding and padding. RSA-OAEP is used for encryption, while RSA-PSS is a signature scheme. Secure operation therefore depends on correct parameter generation, suitable padding, and careful implementation as well as the underlying mathematics [21].

5.3. Elliptic Curve Cryptography (ECC)

Elliptic curve cryptography (ECC) denotes a class of public-key schemes that operate on the group of points of an elliptic curve over a finite field. Their security rests on the presumed intractability of the elliptic-curve discrete logarithm problem (ECDLP) for suitably selected parameters [14, 22].

One of the most significant advantages of ECC over RSA is its ability to provide similar levels of security with smaller key sizes. For example, it is

generally accepted that an ECC key of approximately 256 bits can provide a similar level of security to an RSA key of approximately 3072 bits [10, 19].

This enables ECC-based systems to require less memory and storage space, allows cryptographic operations to be performed more efficiently, and facilitates the use of smaller data sizes during communication.

These features make ECC a particularly important cryptographic option for Internet of Things (IoT) devices, mobile devices and embedded systems, which are limited in terms of processing power, memory, energy consumption and communication capacity. Furthermore, ECC is used in various applications such as key exchange, digital signatures and secure communication protocols [19, 10].

Despite the advantages offered by ECC, the security of the system does not depend solely on key length. It is also crucial for the overall security that the elliptic curve used possesses secure parameters, that keys are generated securely, and that the cryptographic protocols employed are implemented correctly [19, 10].

6. SECURE COMMUNICATION PROTOCOLS

6.1. Hybrid Cryptosystems

Hybrid cryptosystems assign different tasks to symmetric and asymmetric techniques. Public-key mechanisms can authenticate participants or help establish a session secret, while symmetric encryption protects the larger volume of application data efficiently. In contemporary protocols, authenticated-encryption schemes such as AES-GCM may be used for the data channel [8, 10].

This division of work combines public-key methods for establishing trust or session keys with the throughput advantages of symmetric encryption. The resulting architecture can be efficient, provided that authentication, key establishment, and the symmetric mode are configured securely [8].

6.2. TLS 1.3 and the Handshake Mechanism

Transport Layer Security version 1.3 (TLS 1.3) protects data exchanged over connections such as HTTPS [8]. During the handshake, the peers negotiate cryptographic parameters and establish traffic keys; the server is authenticated, and client authentication can also be used where configured. TLS 1.3 reduces handshake latency compared with earlier versions, with a typical full handshake requiring one round trip (1-RTT).

TLS 1.3 supports forward secrecy through ephemeral Diffie–Hellman key exchange. As a result, later compromise of a server's long-term authentication private key does not, by itself, reveal the ephemeral session secrets from previously completed sessions [8].

6.3. Public Key Infrastructure (PKI) and Certificates

Public Key Infrastructure (PKI) comprises the policies, roles, services, and technical components used to bind public keys to identities in a verifiable way. Its components commonly include digital certificates, certificate authorities (CAs), registration and validation processes, and operating policies. A certificate conforming to X.509 can contain the subject's identity information, public key, validity period, and the signature of the issuing authority. Clients validate the certificate contents and signature to establish trust in a presented public key. [23, 10].

PKI commonly organizes trust through a certificate chain. A root CA anchors the chain, and intermediate CAs may issue certificates for servers, organizations, or individuals. A client validates certificate signatures and the chain according to its trust store and relevant policy. In TLS/HTTPS, certificate validation helps confirm that the public key transmitted by the server is associated with the expected website identity; encryption without this check would not by itself prevent impersonation [23].

7. COMPARATİVE ANALYSIS OF ALGORITHMS

Table 1 summarizes AES, RSA, and ECC by their mathematical basis, typical performance, representative key sizes, key-management model, and common applications.

Table 1. Comparison of symmetric and asymmetric cryptographic methods

Parameter / Feature	Symmetric (AES)	Asymmetric (RSA)	Asymmetric (ECC)
Mathematical foundation	Substitution-permutation network; GF(2^8) arithmetic	The integer factorization problem	Elliptic curve discrete logarithm problem (ECDLP)
Computational performance	Generally high	Generally lower; depends on the type of operation	Generally more efficient than RSA; depends on the curve and the operation
Example key sizes and approximate security level	128 - 192 - 256-bit keys; approximately 128 - 192 - 256 security bits	3072 / 7680 / 15360-bit modulus; approximately 128 - 192 - 256 security bits	Approximately 256 - 384 - 512+-bit curve size; approximately 128 - 192 - 256 security bits
Key management	$N(N-1)/2$ keys when secret keys are shared pairwise among N users	One public/private key pair per user	One public/private key pair per user
Primary use cases	Bulk data encryption	Public-key operations and digital signatures	Key exchange, digital signatures, TLS, and constrained devices

Source: Prepared by the authors based on [10, 11, 15, 21, 22]. Note: Key sizes are approximate security-level comparisons; symmetric and asymmetric key lengths are not directly equivalent.

It is evident that, due to its high processing speed, AES offers advantages in the encryption of large volumes of data, whilst RSA and ECC are more suited to tasks such as key management, authentication and digital signatures. Although RSA offers a robust security structure, it requires longer keys to achieve comparable levels of security. ECC, on the other hand, provides significant advantages, particularly for mobile devices, IoT systems and resource-constrained environments, as it can achieve comparable levels of security with smaller key sizes. For this reason, in modern cryptographic systems, these algorithms are generally used not as alternatives to one another, but as complementary components performing different tasks. Whilst symmetric algorithms provide high performance in data encryption operations, asymmetric algorithms are used in secure key management and authentication processes; thus, a balanced solution in terms of both security and performance is achieved within a hybrid structure [10, 11, 15, 21, 22].

8. POST-QUANTUM CRYPTOGRAPHY (PQC) AND ITS FUTURE PERSPECTIVES

8.1. Quantum Threat

The development of quantum computers poses a significant threat to some of the cryptographic algorithms currently in use. The fact that certain mathematical problems, which are computationally difficult to solve on classical computers, can be solved much more quickly on quantum computers with sufficient capacity raises significant concerns regarding the future security of public-key cryptography in particular [24, 25].

In this context, one of the most significant threats is Shor's algorithm. Shor's algorithm makes it possible to solve problems such as the factorisation of large numbers and discrete logarithms in polynomial time on a quantum computer of sufficient size. This directly affects asymmetric cryptographic systems such as RSA and ECC, whose security relies on the computational difficulty of these problems. Consequently, it is predicted that should quantum computers of sufficient capacity emerge, these algorithms—which are currently considered secure—will be unable to maintain their reliability in the long term [24, 25].

In symmetric cryptography, however, the situation is different. The Grover Algorithm provides quantum acceleration for unstructured search problems, reducing the complexity of key-search attacks by approximately a square root factor compared to the classical approach. Consequently, the security level of, for example, a 128-bit symmetric key against quantum attacks may be reduced to approximately that of a 64-bit key. This effect does not mean that symmetric algorithms such as AES become completely ineffective; rather, it necessitates the use of longer keys. For this reason, AES-256 is regarded as one of the options providing a higher security margin against the threats that quantum computers could pose [25, 26].

Another significant aspect of the quantum threat is the encrypted data collected today that could be decrypted in the future. Under the approach known as 'Harvest Now, Decrypt Later' (HNDL), attackers may aim to store encrypted communications they have intercepted today, with the intention of decrypting this data in the future once sufficient quantum computing capacity has been achieved. Particularly in the case of sensitive data that must remain confidential for a long time, this situation demonstrates that the transition to quantum-resistant cryptography is important not only in terms of future threats but also in terms of data security today [27].

In light of these developments, research into Post-Quantum Cryptography (PQC) has gained momentum. The aim is to develop new cryptographic

algorithms that are resilient to the computational advantages that quantum computers may possess, whilst remaining applicable on classical computers. The objective is thus to establish alternative mechanisms to counter the security risks that traditional public-key systems, such as RSA and ECC, may pose in the future [24, 25].

8.2. NIST PQC Standardization Process

In light of the threats that quantum computers could pose to traditional public-key systems such as RSA and ECC, the US National Institute of Standards and Technology (NIST) launched a comprehensive standardisation process in 2016 aimed at identifying post-quantum cryptographic algorithms. As part of this process, algorithms developed by researchers from various countries and the cryptography community were evaluated in terms of security, performance and feasibility [25].

As a result of this work, NIST published the first three post-quantum cryptography standards in August 2024. These are the ML-KEM standard under FIPS 203, the ML-DSA standard under FIPS 204, and the SLH-DSA standard under FIPS 205. Whilst ML-KEM is used for key encapsulation and secure key generation, ML-DSA and SLH-DSA are designed for digital signature operations [28, 29, 30].

- Key encapsulation: CRYSTALS-Kyber was standardized as ML-KEM in FIPS 203. The mechanism allows parties to establish a shared secret and is based on module-lattice problems [28].

- Digital signatures: CRYSTALS-Dilithium was standardized as ML-DSA in FIPS 204. It provides algorithms for generating and verifying digital signatures and is based on module-lattice constructions [29].

- Digital signatures: SPHINCS+ was standardized as SLH-DSA in FIPS 205. It is a stateless hash-based signature scheme and therefore offers a design approach distinct from the lattice-based signature standard [30].

It presents a non-lattice-based digital signature approach based on hash functions and establishes a security foundation distinct from other standards [26].

NIST is continuing its work on developing the FIPS 206 standard for the Falcon algorithm, known as FN-DSA. Furthermore, in 2025, HQC was selected for standardisation with the aim of diversifying post-quantum key encapsulation options; the 2025 report published by NIST states that the

standard for HQC will be prepared at a later date. Therefore, FN-DSA should not be presented as a completed standard, nor should HQC be presented as a published FIPS standard [25].

8.3. "Harvest Now, Decrypt Later" Attacks

"Harvest Now, Decrypt Later" (HNDL) is an attack strategy in which attackers aim to seize encrypted data—which is currently considered secure—with the intention of decrypting it in the future once sufficient quantum computing capacity has been achieved. An encryption mechanism that appears sufficiently strong cryptographically today may become vulnerable in the future when faced with the computational power that quantum computers may eventually deliver. NIST also emphasises that this threat must be taken into account today, particularly with regard to data that needs to remain confidential for a long period. In this attack model, the attacker does not need to break the encryption today. Instead, network traffic, emails, corporate documents or other sensitive data are collected and stored in their encrypted form. Should a quantum computer with sufficient capacity be developed in the future, the aim may be to decrypt data collected in the past by targeting public-key algorithms—such as RSA and ECC—that are vulnerable to quantum computers. Consequently, the fundamental risk of the attack depends not only on the data's current security but also on its ability to remain secure for the entire period during which it must remain confidential [22].The HNDL threat is particularly significant for government and defense information, financial records, health data, intellectual property, and strategic corporate information that require long-term confidentiality. For example, if data compromised today—which is supposed to remain confidential for a decade—can be decrypted by quantum computers in the future, protecting that data with today's encryption methods will not be sufficient in the long term. For this reason, NIST recommends that efforts to transition to post-quantum cryptography begin without waiting for the emergence of quantum computers. This threat demonstrates that the transition to Post-Quantum Cryptography (PQC) standards is important not only for protecting future systems but also for the security of data created today that must be protected for a long time. In this context, it is important for organizations to identify the cryptographic algorithms they use, detect systems that are vulnerable to quantum computers, and prepare for the transition to appropriate PQC or hybrid solutions [1, 22].

9. CONCLUSION

Modern cryptography is based on a holistic framework in which symmetric and asymmetric cryptographic methods are used in conjunction to meet the requirements of confidentiality, integrity, authentication and non-repudiation—the fundamental elements of information security. Whilst symmetric algorithms such as AES enable the secure encryption of large

volumes of data thanks to their high processing speeds, asymmetric algorithms such as RSA and ECC play significant roles in areas such as secure key management, authentication and digital signatures. Technologies such as TLS 1.3 and PKI, in turn, bring these cryptographic mechanisms together to facilitate the establishment of a secure communication infrastructure. However, advances in quantum computing technologies pose a significant threat to the long-term security of traditional public-key systems such as RSA and ECC. In particular, the threat posed by Shor's Algorithm to the mathematical problems on which these systems are based, and the 'Hook, Line and Sinker' (HLS) attack model, necessitate that post-quantum security research be addressed today [8, 10, 24, 27].

The adoption of post-quantum cryptographic standards is therefore becoming an increasingly important component of long-term security planning. NIST's ML-KEM, ML-DSA, and SLH-DSA standards provide standardized options for key establishment and digital signatures in a post-quantum setting [28, 29, 30]. A realistic transition will likely require phased adoption and interoperability planning, including the evaluation of hybrid approaches alongside established protocols such as TLS 1.3.

Ultimately, cryptographic security is a system property rather than a result of algorithm selection alone. Key management, protocol design, implementation quality, configuration, and ongoing reassessment all influence the protection achieved in practice. Preparing for post-quantum threats therefore involves a sustained review of existing infrastructure, data lifetimes, and migration priorities, not merely replacing one algorithm with another [10, 1].

REFERENCES

[1] Pascoe, C., Quinn, S., & Scarfone, K. (2024). The NIST Cybersecurity Framework (CSF) 2.0 (NIST CSWP 29). National Institute of Standards and Technology. https://doi.org/10.6028/NIST.CSWP.29

[2] Yeniman Yıldırım, E. (2018). Bilişim sistemlerine yönelik siber saldırılar ve siber güvenliğin sağlanması. Mesleki Bilimler Dergisi, 7(2), 24–33. https://izlik.org/JA72ZG64MK

[3] Gülburun, S., & Dener, M. (2022). Bulut bilişim güvenliğindeki zorluklar ve güncel çalışmalar üzerine bir inceleme. Bilişim Teknolojileri Dergisi, 15(1), 45–53.

[4] Yeşilbaş, E. (2016). Cebirsel kriptoloji yöntemleri ve bazı uygulamaları [Yüksek lisans tezi, Recep Tayyip Erdoğan Üniversitesi].

[5] Macit, H. B. (2025). Bilgisayar ağlarında siber tehditler, önleme yöntemleri ve teknolojiler. In Bilişim sistemlerinden yapay zekâya: Dijital dünyanın temelleri ve uygulamaları (pp. 219–245). Eğitim Yayınevi.

[6] Dicle, S. Z. (2022). Ortadaki adam saldırısı (MITM). Avrupa Bilim ve Teknoloji Dergisi, (42), 100–107. https://doi.org/10.31590/ejosat.1187984

[7] Avaroğlu, E. (2022). Bilgi güvenliğinin temel yapı taşı: Kriptoloji. Düşünce Dünyasında Türkiz, 8(43), 53–65.
[8] Rescorla, E. (2018). The Transport Layer Security (TLS) protocol version 1.3 (RFC 8446). Internet Engineering Task Force. https://doi.org/10.17487/RFC8446
[9] Cochran, K. A. (2024). The CIA triad: Safeguarding data in the digital realm. In Cybersecurity essentials: Practical tools for today's digital defenders (pp. 17–32). Apress.
[10] Barker, E. (2020). Recommendation for key management: Part 1—General (NIST Special Publication 800-57 Part 1, Rev. 5). National Institute of Standards and Technology. https://doi.org/10.6028/NIST.SP.800-57pt1r5
[11] Kaya, A., & Türkoğlu, İ. (2023). Simetrik ve asimetrik şifreleme algoritmalarının performans karşılaştırılması. Fırat Üniversitesi Mühendislik Bilimleri Dergisi, 35(2), 891–900. https://doi.org/10.35234/fumbd.1296228
[12] National Institute of Standards and Technology. (2015a). Secure hash standard (SHS) (FIPS PUB 180-4). U.S. Department of Commerce. https://doi.org/10.6028/NIST.FIPS.180-4
[13] Demirbilek Ayyıldız, S. (2021). Güvenli elektronik seçim için sır paylaşım şemalarının uygulanması [Yüksek lisans tezi, Trakya Üniversitesi].
[14] Katz, J., & Lindell, Y. (2020). Introduction to modern cryptography (3rd ed.). CRC Press.
[15] National Institute of Standards and Technology. (2023). Advanced Encryption Standard (AES) (FIPS PUB 197, upd. 1). U.S. Department of Commerce. https://doi.org/10.6028/NIST.FIPS.197-upd1
[16] Dworkin, M. (2007). Recommendation for block cipher modes of operation: Galois/Counter Mode (GCM) and GMAC (NIST Special Publication 800-38D). National Institute of Standards and Technology. https://doi.org/10.6028/NIST.SP.800-38D
[17] Dworkin, M. (2001). Recommendation for block cipher modes of operation: Methods and techniques (NIST Special Publication 800-38A). National Institute of Standards and Technology. https://doi.org/10.6028/NIST.SP.800-38A
[18] National Institute of Standards and Technology. (2015b). SHA-3 standard: Permutation-based hash and extendable-output functions (FIPS PUB 202). U.S. Department of Commerce. https://doi.org/10.6028/NIST.FIPS.202
[19] National Institute of Standards and Technology. (2022, December 15). NIST's policy on hash functions. https://csrc.nist.gov/Projects/Hash-Functions/NIST-Policy-on-Hash-Functions
[20] Turner, S., & Chen, L. (2011). Updated security considerations for the MD5 message-digest and the HMAC-MD5 algorithms (RFC 6151). Internet Engineering Task Force. https://doi.org/10.17487/RFC6151
[21] Moriarty, K., Kaliski, B., Jonsson, J., & Rusch, A. (2016). PKCS #1: RSA cryptography specifications version 2.2 (RFC 8017). Internet Engineering Task Force. https://doi.org/10.17487/RFC8017
[22] Chen, L., Moody, D., Regenscheid, A., Robinson, A., & Randall, K. (2023). Recommendations for discrete logarithm-based cryptography: Elliptic curve domain parameters (NIST Special Publication 800-186). National Institute of Standards and Technology. https://doi.org/10.6028/NIST.SP.800-186

[23] Cooper, D., Santesson, S., Farrell, S., Boeyen, S., Housley, R., & Polk, W. (2008). Internet X.509 public key infrastructure certificate and certificate revocation list (CRL) profile (RFC 5280). Internet Engineering Task Force. https://doi.org/10.17487/RFC5280

[24] Shor, P. W. (1994). Algorithms for quantum computation: Discrete logarithms and factoring. In Proceedings of the 35th Annual Symposium on Foundations of Computer Science (pp. 124–134). IEEE. https://doi.org/10.1109/SFCS.1994.365700

[25] Alagic, G., Bros, M., Ciadoux, P., Cooper, D., Dang, Q., Dang, T., Kelsey, J., Lichtinger, J., Liu, Y.-K., Miller, C., Moody, D., Peralta, R., Perlner, R., Robinson, A., Silberg, H., Smith-Tone, D., & Waller, N. (2025). Status report on the fourth round of the NIST post-quantum cryptography standardization process (NISTIR 8545). National Institute of Standards and Technology. https://doi.org/10.6028/NIST.IR.8545

[26] Grover, L. K. (1996). A fast quantum mechanical algorithm for database search. In Proceedings of the twenty-eighth annual ACM symposium on Theory of computing (pp. 212–219). Association for Computing Machinery. https://doi.org/10.1145/237814.237866

[27] National Institute of Standards and Technology. (n.d.). What is post-quantum cryptography? https://www.nist.gov/cybersecurity-and-privacy/what-post-quantum-cryptography

[28] National Institute of Standards and Technology. (2024a). Module-lattice-based key-encapsulation mechanism standard (FIPS PUB 203). U.S. Department of Commerce. https://doi.org/10.6028/NIST.FIPS.203

[29] National Institute of Standards and Technology. (2024b). Module-lattice-based digital signature standard (FIPS PUB 204). U.S. Department of Commerce. https://doi.org/10.6028/NIST.FIPS.204

[30] National Institute of Standards and Technology. (2024c). Stateless hash-based digital signature standard (FIPS PUB 205). U.S. Department of Commerce. https://doi.org/10.6028/NIST.FIPS.205

Breaking Boundaries in Natural Language Processing: The Road from RNNs to Transformers

Abdo IBRAHIM
Ammar Mohammed Ali AL-QADASI
Bilal YOUSEFI

1- Dept. of Computer Engineering, Karabuk University, Karabuk, Türkiye, ORCID:0009-0008-9254-510X
2- Dept. of Computer Engineering, Karabuk University, Karabuk, Türkiye, ORCID: 0000-0002-1147-4696
3- Dept. of Computer Engineering, Karabuk University, Karabuk, Türkiye, ORCID: 0000-0002-8397-3654

ABSTRACT

The evolution of Natural Language Processing (NLP) architectures has led to important advancements in the ability to understand and generate the human language. This paper explores the transformative journey from RNNs to Transformers architectures and highlights their strengths, limitations and the factors that drove their adoption. In the early models of NLP including RNNs and their enhanced versions such as Long Short-Term Memory (LSTM) networks and Gated Recurrent Units (GRUs) were successful in sequential data processing they faced some challenges such as the vanishing gradient problem, the need for long time to be trained, and the difficulty to capture long-term dependencies. The introduction of the Attention Mechanism solved these issues and presented the foundation for Transformers models which revolutionized the field with their parallel processing capabilities and self-attention mechanisms. Transformer-based models like BERT and GPT have achieved high performance across many NLP tasks from machine translation to question answering. Despite the success of these models, some challenges such as computational resource demands, ethical concerns, and fine-tuning complexities remain in the domain. A comparative analysis of RNNs and Transformers is provided in this paper to emphasize their impact on modern NLP applications and to outline the future research directions to enhance scalability, efficiency, and the ethical deployment of these models. The transition from RNNs to Transformers marks a significant shift, enhances NLP, and placing it at the forefront of AI-driven innovation and real-word applications.

Keywords - NLP, RNNs, Transformers, Attention Mechanisms, Self-Attention, LLMs.

I. INTRODUCTION

Natural Language Processing (NLP) is one of the fastest evolving field and represents an intersection of computer science, linguistics, and artificial intelligence. It focuses on making the machines understand, interpret, and generate the spoken human language [1]. Starting from foundational applications like language translation and sentiment analysis to advanced ones such as conversational AI and automated content generation, NLP played an important role in transforming the way of our interaction with technology [2], [3]. Over the years, significant improvements in NLP architectures have propelled the field forward which gives models the ability to handle the complex language tasks with improved accuracy and efficiency [4]. But why have researchers pursued new NLP techniques? The Early models of NLP heavily relied on rule-based systems which were built using handcrafted linguistic features and rules. These types of systems required efforts and needed experts to encode grammatical structures and semantic rules explicitly. While these techniques were effective for specific and

narrow tasks, they lacked scalability and adaptability to diverse and complex patterns which made them unsuitable for handling the complex and ambiguity inherent in natural language [5].

The introduction of statistical methods was an important shift from deterministic rule-based systems to probabilistic approaches. N-grams, Hidden Markov Models (HMMS), and Conditional Random Field (CRFs) are techniques that became popular for tasks like named entity recognition and machine translation [6]. These models utilized the statistical patterns in large text corpora to improve language understanding and prediction. However, statistical methods often suffered with capturing long-term dependencies and required significant computational resources for training large-scale models. These early methods formed the base for neural network-based approaches like Recurrent Neural Networks (RNNs) which marked a significant milestone [7], [8]. However, the field has been revolutionized by the introduction of Transformer-based architectures which offer superior performance and scalability [9].

This review paper aims to explore the journey from RNNs to Transformers and examines the evolution of NLP architectures. We will discuss the strengths and limitations of RNNs and the revolutionary features of Transformers and the reasons behind their widespread adoption. By comparing these two approaches, this paper seeks to highlight the transformative impact of architectural advancements on NLP and their implications for future research and applications [10].

II. Evolution of Neural Network Architectures in NLP

II.I. Early Neural Approaches

Feedforward neural networks were the first neural networks applied in NLP and formed the foundation for further advancements. The introduction of word embedding such as Word2Vec [11] and GloVe [12] was one of the most important developments during this period. These types of embedding techniques were designed to represent words as dense vectors in a continuous space to capture semantic relationships between words based on their contexts. The dense representation marked a crucial shift from traditional sparse representation like one-hot encoding [13], which allows models to be more specific in capturing the meaning and improving the performance in tasks like sentiment analysis [14], language modeling [15], and machine translation [16]. However, these types of neural networks had one major limitation, they are not good in modeling sequential data because they process inputs independently without any inherent structure to capture temporal dependencies that exist in words' sequence. This became obvious when dealing with tasks that require understanding the order of the words or the relationships between them, such as speech recognition [17]. However,

more advance models were developed by researchers to handle the sequential nature of languages.

II.II. RECURRENT NEURAL NETWORKS (RNNS)

As a solution for the difficulty of processing the sequential data, Recurrent Neural Networks (RNNs) were introduced. The difference between feedforward networks and RRNs is that RRNs maintain a hidden state that is updated at each time step which allows the model to carry information from one word to the next which means capturing dependencies across the sequence [18]. This made RRNs a good choice for NLP tasks like sentiment analysis [19], machine translation [16], and text generation [20] because the meaning of a word depends on the context provided by previous words in the sequence.

Although RNNs have many advantages but they also have significant limitations. The primary one is the vanishing gradient problem, which happens when gradient become very small during training. This problem makes it difficult for networks to learn long-term dependencies [21] and becomes obvious when training RNNs on long sequences where the influence of the earlier words fades and the model faces difficulties to learn the context required for accurate predictions. Furthermore, RNNs are slow to train because they process sequences one step at a time and this limits their ability to scale effectively [9]. Figure 1 shows the architecture of RNN and its different layers.

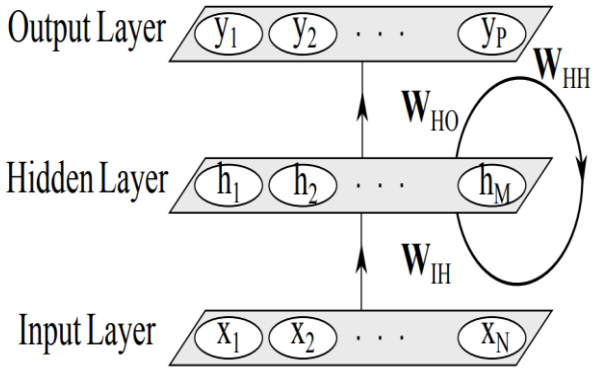

Figure 1. The RNN model architecture. Source: [7].

II.III. LSTMs, GRUs, and Advanced Recurrent Architectures

Long Short-Term Memory (LSTM) networks [22] and Gated Recurrent Units (GRUs) [23] were introduced to overcome the limitations of traditional RNNs. Both architectures solve the vanishing gradient by using gates that control the flow of information through the network. Three gates are used by LSTMs: the input gate, the forget gate, and the output gate and this allow the model to selectively discard or retain information at each time step. GRUs are simpler than LSTMs, just two gates are used: the update gate and the reset gate which also control the flow of the information but with fewer parameters.

These architectures improved the ability of RNNs to capture long-range dependencies and resulted in notable improvements in performance on a range of NLP tasks. LSTMs and GRUs became the most used architectures for many sequence-based applications like machine translation [16], speech recognition [24], and language modeling [8]. However, these architectures suffer from limitations in terms of parallelization and computational efficiency because of their sequential nature especially when processing long sequences and training large models on extensive datasets [9].

II.IV. Attention Mechanisms and the Road to Transformers

The discovering of the Attention Mechanism [25] led to address the limitations of RNNs, LSTMs, and GRUs. Especially when dealing with long sequences. In the traditional RNNs models, a vector with fixed-length context is passed from the encoder to the decoder and this is not practical and can lead to loss of important information for long sequences. However, Attention Mechanisms allow the model to focus on different parts of the input sequence at each step of the decoding process which provides a dynamic and flexible way to capture relevant information from the entire sequence [26]. These mechanisms became a crucial part of sequence-to-sequence models, and they improved performance in tasks like machine translation [16], where the relationships between words at different positions in the sequence is significant. Attention mechanism allows the model to selectively focus on words which are the most relevant for generating the output at each decoding step regardless of their distance from the current word and this dynamic ability represented by focusing on different parts of the input sentence made the attention particularly effective for handling long-range dependencies and more complex tasks. The seminal Transformer architecture [9] improved the effect of the attention mechanism by eliminating the use of recurrent layers altogether because the transformer uses self-attention mechanism that computes attention weights for all words in a sequence simultaneously. Using this mechanism enables the model to learn relationships between words in parallel. This innovation had a big impact in significant improvements in performance and computational efficiency because transformers can be trained more quickly and handle

much larger datasets than RNNs models. Transformers became the base of many state-of-the-art models in NLP such as GPT [27] and BERT [10] which are used now in many fields like language understanding, question answering, and text generation. In transformers, the self-attention mechanism allows for both local and global dependencies to be captured across the whole sequence which make them more effective in representing long-range dependencies than RNNs. Additionally, the scale operation is better because of the parallelizable nature of transformers which led to more robust models that can be trained faster.

III. Transition to Transformers

Due to the limitations in the earlier methods that were used in NLP, this led to the need to use new approaches that can enhance and solve the problems of the older ones. RNNs and their improved versions like LSTMs were revolutionary for sequential data processing but they struggled with long-range dependencies because of vanishing gradient which makes it difficult to capture relationships between the words that are distant in a sequence [22]. Additionally, their sequential nature limited the efficiency of computation since words had to be processed at once which prevent parallelization. These problems were the reasons for a more efficient and scalable architecture which is the Transformer model [9], [28].

III.I. Self-Attention and the Transformer Model

Vaswani et al. introduced the Transformer architecture in their seminal paper "Attention Is All You Need" [9] which marked a significant shift in NLP. The Transformer relies on the self-attention mechanism that gives the model the ability to weigh the importance of different words in a sentence and the correlation among each other regardless of their position. The transformer consists of two main parts, the encoder and the decoder where both of these components are built using the self-attention layers and feed-forward neural networks. Transformers process input sequences in parallel which makes them more efficient than RNNs. The Transformers have a Multi-Head Attention [29] which allows the model to process multiple parts of the input at the same time and capture various aspects of relationships between words and they have a Positional Encoding [30] which is responsible for the word order since the transformers do not process data sequentially. The transformers have several advantages over RNNs like parallelization which enables parallel processing of the input and improves the computational speed, Long-Range Dependencies which allows the model to capture relationsh ips between distant words through the attention mechanism, and the Scalability where transformers can scale well with large datasets and computational resources which enables training of massive models. The

complete architecture of the Vanilla version of Transformer can be seen in figure 2.

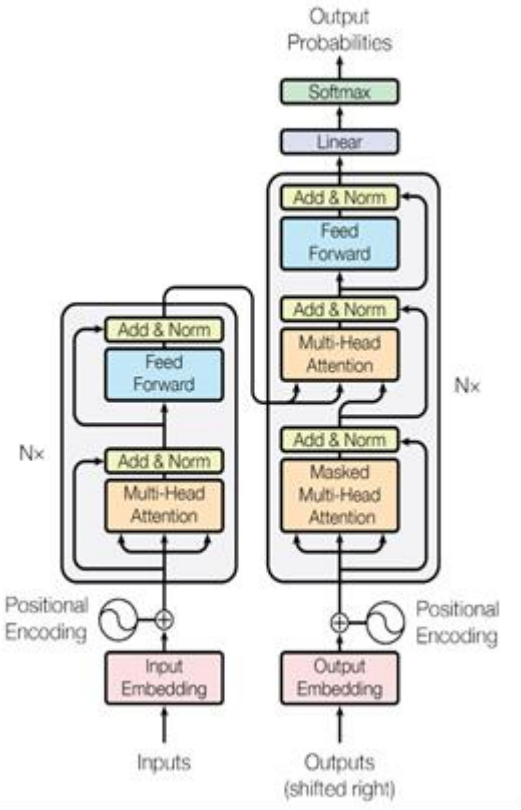

Figure 2. The Transformer model architecture. Source: [9].

III.II. BERT AND GPT

BERT and GPT are specialized architectures that are driven by the success of the transformer model, and they advanced the field of NLP further. The recent advancements have focused on making these models more scalable and enhancing their efficiency. Research seeks to explore more sophisticated applications such as RoBERTa [31], XLNet [32], ELECTRA [33], and ALBERT [34].

III.II.I. BERT: BIDIRECTIONAL CONTEXT AND PRETRAINING

The bidirectional encoder representation of transformers model (BERT) introduced a new approach for NLP by enhancing the bidirectional context during the training operation. BERT uses a masked language modeling technique to predict the words that are masked based on both the left and the

right context unlike the autoregressive models that predict the words sequentially. This bidirectional approach along with pretraining on large-scale corpora led to the improvements of the performance on downstream tasks like question answering, text classification, and named entity recognition [10].

III.II.II. GPT: Autoregressive Models and Transfer Learning

The Generative Pretrained Transformer (GPT) series uses an autoregressive approach where tokens are predicted in a sequential way. These models allow a large mode to be trained on massive amount of data and fine-tuning it for specific tasks. With the innovation of this approach, notable improvements appeared especially in generative tasks such as text completion, summarization, and conversation [27], [35].

III.III. Emergence of Large Language Models (LLMs)

The continues advancements in transformers resulted in the development of Large Language Models (LLM) such as GPT-3, GPT-4 [36] and beyond. The ability of this models to leverage billions of parameters which trained on huge datasets enables them to be multi-tasked models with minimal supervision. These models become the central for most generative tasks like Text Generation which is used for producing coherent and contextually accurate text for tasks like summarization and creative working, Conversational Agents which are responsible for powering the chatbots and virtual assistants that engage in the human-like conversations, and Translation and Question Answering. These applications emphasize the versatility and the important impact of LLMs across industries from content creation to healthcare and education [37].

III.IV. Challenges with Transformers

Despite the success of Transformers and LLMs, they face several challenges that should be pointed to ensure the sustainability and ethical deployment of these models. These challenges are:

- the difficulty to obtain substantial computational resources which limits accessibility for smaller organizations and researchers. There are some ongoing efforts to reduce the model size and the training costs through techniques like model distillation and quantization [38].
- The possibility that these models will be trained on datasets scraped from internet which often contain biases. These biases can affect on the outputs of the model and raise the ethical concerns especially in real world applications [39].

III.IV.I. Scalability and Fine-Tuning

Transformers models have been improved with the scaling process, but it also brings challenges related to fine-tuning and deployment. They have pushed benchmark performances to new heights, but they often require task-specific fine-tuning to achieve optimal results. Fine-tuning remains crucial for aligning models with specific domains and data distributions [40], [41], [42]. It is important to address these challenges to unlock the full potential of Transformers but ensuring they remain accessible, ethical, and efficient.

IV. Comparison: RNNs vs. Transformers

IV. I. Performance and Scalability

Transformers generally are better than RNNs in terms of both performance and scalability because they process the entire sequence at the same time using their self-attention mechanisms which allows them to handle larger datasets and longer sequences efficiently and makes them crucial for tasks that involve understanding extensive context [9], [10]. Unlike RNNs, which process the data sequentially and suffer from the vanishing gradient problem [22], [21].

IV. II. Performance and Scalability

RNNs and their improved versions like LSTMs and GRUs are effective in where the sequence order and the temporal dynamics are essential. They are traditionally used for tasks like speech recognition and simple text classification where the attention is toward the contextual link between immediate elements [17], [43]. Transformers succeed in the NLP tasks that require integration of broader contextual information because of their architecture that enables them to deal with complex tasks due to their ability to evaluate the importance of each part of the input [44], [45] such as machine translation, contextual question answering, and text summarization.

IV. III. Performance and Scalability

RNNs have diminished over time after they were crucial in the earlier stages of NLP development because of their limitations with long sequences and large datasets [46], [47]. In contrast, Transformers have gained the popularity and have become the preferred architecture in both research and industry and that due to their robust performance across the various NLP benchmarks and tasks. Their scalability and efficiency have enabled to develop improve and new models like GPT and BERT and have pushed the boundaries of NLP development [31], [35]. Table 1 outlines the differences between RNNs (such as Vanilla RNNs, LSTMs, and GRUs) and

Transformer models (including GPT, BERT, RoBERTa, XLNet, ELECTRA, and ALBERT) across various metrics such as performance, scalability, task suitability, and industry adoption.

Table 1: Comparative Overview of RNN and Transformer Models in

Features	RNN Models	Transformer Models
Model Types	Vanilla RNNs- LSTMs- GRUs	GPT- BERT- RoBERTa- XLNet- ELECTRA- ALBERT
Performance	Limited by sequential data processing and issues like the vanishing gradient problem.	Superior in handling long sequences and parallel processing.
Scalability	Challenges in scaling due to sequential processing.	Highly scalable, facilitates training of massive models efficiently.
NLP Task Suitability	Suitable for simpler, sequential tasks.	Excel in complex tasks requiring global context understanding.
Adoption	Declining in cutting-edge applications, still used in specific scenarios.	Dominates modern NLP research and applications.

V. CONCLUSION

The transition from Recurrent Neural Network (RNNs) to Transformers in Natural Language Processing (NLP) represents an important change in its complex tasks. Although RNNs have the ability to handle the sequential data and short-term dependencies they were limited by the challenges like vanishing gradient problem and scalability issues. In the other side, The Transformers have revolutionized the NLP landscape by utilizing the self-attention mechanisms which are responsible for processing the entire sequence simultaneously and that led to enhance the efficiency and the scalability. Enhanced models of Transformers such as BERT and GPT have appeared, and these models have succeeded to achieve unprecedented performance across a range of NLP applications. They have set new benchmarks and have pushed the boundaries of automated language understanding. As NLP continues to advance, there is an expanded focus to include the ethical use of AI and to reduce the computational demands which will ensure the future developments in Transformer technology and will enhance both the capability and accessibility of AI systems in processing the human languages.

VI. REFERENCES

[1] "(PDF) Speech and Language Processing: An Introduction to Natural Language Processing, Computational Linguistics, and Speech Recognition," ResearchGate. Accessed: Dec. 14, 2024. [Online]. Available: https://www.researchgate.net/publication/200111340_Speech_and_Language_Proce

ssing_An_Introduction_to_Natural_Language_Processing_Computational_Linguistics_and_Speech_Recognition
[2] E. Cambria and B. White, “Jumping NLP Curves: A Review of Natural Language Processing Research [Review Article],” IEEE Computational Intelligence Magazine, vol. 9, no. 2, pp. 48–57, May 2014, doi: 10.1109/MCI.2014.2307227.
[3] T. Young, D. Hazarika, S. Poria, and E. Cambria, “Recent Trends in Deep Learning Based Natural Language Processing [Review Article],” IEEE Computational Intelligence Magazine, vol. 13, no. 3, pp. 55–75, Aug. 2018, doi: 10.1109/MCI.2018.2840738.
[4] “(PDF) Advances in natural language processing,” ResearchGate, Oct. 2024, doi: 10.1126/science.aaa8685.
[5] L. Chiticariu, Y. Li, and F. R. Reiss, “Rule-Based Information Extraction is Dead! Long Live Rule-Based Information Extraction Systems!,” in Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing, Seattle, Washington, USA: Association for Computational Linguistics, 2013, pp. 827–832. doi: 10.18653/v1/D13-1079.
[6] J. Lafferty, A. McCallum, and F. Pereira, “Conditional Random Fields: Probabilistic Models for Segmenting and Labeling Sequence Data,” presented at the International Conference on Machine Learning, Jun. 2001. Accessed: Dec. 15, 2024. [Online]. Available: https://www.semanticscholar.org/paper/Conditional-Random-Fields%3A-Probabilistic-Models-for-Lafferty-McCallum/f4ba954b0412773d047dc41231c733de0c1f4926
[7] H. Salehinejad, S. Sankar, J. Barfett, E. Colak, and S. Valaee, “Recent Advances in Recurrent Neural Networks,” Feb. 22, 2018, arXiv: arXiv:1801.01078. doi: 10.48550/arXiv.1801.01078.
[8] “(PDF) Recurrent neural network based language model,” ResearchGate. Accessed: Dec. 15, 2024. [Online]. Available: https://www.researchgate.net/publication/311469848_Recurrent_neural_network_based_language_model
[9] A. Vaswani et al., “Attention Is All You Need,” Aug. 02, 2023, arXiv: arXiv:1706.03762. doi: 10.48550/arXiv.1706.03762.
[10] J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova, “BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding,” May 24, 2019, arXiv: arXiv:1810.04805. doi: 10.48550/arXiv.1810.04805.
[11] T. Mikolov, K. Chen, G. Corrado, and J. Dean, “Efficient Estimation of Word Representations in Vector Space,” Sep. 07, 2013, arXiv: arXiv:1301.3781. doi: 10.48550/arXiv.1301.3781.
[12] “(PDF) Glove: Global Vectors for Word Representation,” in ResearchGate, doi: 10.3115/v1/D14-1162.
[13] Y. Goldberg, “A Primer on Neural Network Models for Natural Language Processing,” Oct. 02, 2015, arXiv: arXiv:1510.00726. doi: 10.48550/arXiv.1510.00726.
[14] Y. Kim, “Convolutional Neural Networks for Sentence Classification,” Sep. 03, 2014, arXiv: arXiv:1408.5882. doi: 10.48550/arXiv.1408.5882.
[15] L. Qin et al., “Large Language Models Meet NLP: A Survey,” May 21, 2024, arXiv: arXiv:2405.12819. doi: 10.48550/arXiv.2405.12819.
[16] D. Bahdanau, K. Cho, and Y. Bengio, “Neural Machine Translation by Jointly Learning to Align and Translate,” May 19, 2016, arXiv: arXiv:1409.0473. doi: 10.48550/arXiv.1409.0473.

[17] “Speech Recognition with Deep Recurrent Neural Networks | Request PDF,” ResearchGate, Dec. 2024, doi: 10.1109/ICASSP.2013.6638947.
[18] M.-T. Luong, H. Pham, and C. D. Manning, “Effective Approaches to Attention-based Neural Machine Translation,” Sep. 20, 2015, arXiv: arXiv:1508.04025. doi: 10.48550/arXiv.1508.04025.
[19] E. Cambria, B. Schuller, Y. Xia, and C. Havasi, “New Avenues in Opinion Mining and Sentiment Analysis,” IEEE Intelligent Systems, vol. 28, no. 2, pp. 15–21, Mar. 2013, doi: 10.1109/MIS.2013.30.
[20] A. Graves, “Generating Sequences With Recurrent Neural Networks,” Jun. 05, 2014, arXiv: arXiv:1308.0850. doi: 10.48550/arXiv.1308.0850.
[21] R. Pascanu, T. Mikolov, and Y. Bengio, “On the difficulty of training Recurrent Neural Networks,” Feb. 16, 2013, arXiv: arXiv:1211.5063. doi: 10.48550/arXiv.1211.5063.
[22] “(PDF) Long Short-term Memory,” ResearchGate, Dec. 2024, doi: 10.1162/neco.1997.9.8.1735.
[23] J. Chung, C. Gulcehre, K. Cho, and Y. Bengio, “Empirical Evaluation of Gated Recurrent Neural Networks on Sequence Modeling,” Dec. 11, 2014, arXiv: arXiv:1412.3555. doi: 10.48550/arXiv.1412.3555.
[24] A. Graves and N. Jaitly, “Towards End-To-End Speech Recognition with Recurrent Neural Networks,” in Proceedings of the 31st International Conference on Machine Learning, PMLR, Jun. 2014, pp. 1764–1772. Accessed: Dec. 15, 2024. [Online]. Available: https://proceedings.mlr.press/v32/graves14.html
[25] Z. Niu, G. Zhong, and H. Yu, “A review on the attention mechanism of deep learning,” Neurocomputing, vol. 452, pp. 48–62, Sep. 2021, doi: 10.1016/j.neucom.2021.03.091.
[26] M.-T. Luong, H. Pham, and C. D. Manning, “Effective Approaches to Attention-based Neural Machine Translation,” Sep. 20, 2015, arXiv: arXiv:1508.04025. doi: 10.48550/arXiv.1508.04025.
[27] A. Radford and K. Narasimhan, “Improving Language Understanding by Generative Pre-Training,” 2018. Accessed: Dec. 15, 2024. [Online]. Available: https://www.semanticscholar.org/paper/Improving-Language-Understanding-by-Generative-Radford-Narasimhan/cd18800a0fe0b668a1cc19f2ec95b5003d0a5035
[28] T. Lin, Y. Wang, X. Liu, and X. Qiu, “A survey of transformers,” AI Open, vol. 3, pp. 111–132, Jan. 2022, doi: 10.1016/j.aiopen.2022.10.001.
[29] A. Galassi, M. Lippi, and P. Torroni, “Attention in Natural Language Processing,” IEEE Transactions on Neural Networks and Learning Systems, vol. 32, no. 10, pp. 4291–4308, Oct. 2021, doi: 10.1109/TNNLS.2020.3019893.
[30] P. Dufter, M. Schmitt, and H. Schütze, “Position Information in Transformers: An Overview,” Computational Linguistics, vol. 48, no. 3, pp. 733–763, Sep. 2022, doi: 10.1162/coli_a_00445.
[31] Y. Liu et al., “RoBERTa: A Robustly Optimized BERT Pretraining Approach,” Jul. 26, 2019, arXiv: arXiv:1907.11692. doi: 10.48550/arXiv.1907.11692.
[32] Z. Yang, Z. Dai, Y. Yang, J. Carbonell, R. Salakhutdinov, and Q. V. Le, “XLNet: Generalized Autoregressive Pretraining for Language Understanding,” Jan. 02, 2020, arXiv: arXiv:1906.08237. doi: 10.48550/arXiv.1906.08237.
[33] K. Clark, M.-T. Luong, Q. V. Le, and C. D. Manning, “ELECTRA: Pre-training Text Encoders as Discriminators Rather Than Generators,” Mar. 23, 2020, arXiv: arXiv:2003.10555. doi: 10.48550/arXiv.2003.10555.

[34] Z. Lan, M. Chen, S. Goodman, K. Gimpel, P. Sharma, and R. Soricut, “ALBERT: A Lite BERT for Self-supervised Learning of Language Representations,” Feb. 09, 2020, arXiv: arXiv:1909.11942. doi: 10.48550/arXiv.1909.11942.
[35] A. Radford, J. Wu, R. Child, D. Luan, D. Amodei, and I. Sutskever, “Language Models are Unsupervised Multitask Learners,” 2019. Accessed: Oct. 31, 2024. [Online]. Available: https://www.semanticscholar.org/paper/Language-Models-are-Unsupervised-Multitask-Learners-Radford-Wu/9405cc0d6169988371b2755e573cc28650d14dfe
[36] J. A. Baktash and M. Dawodi, “Gpt-4: A Review on Advancements and Opportunities in Natural Language Processing,” May 04, 2023, arXiv: arXiv:2305.03195. doi: 10.48550/arXiv.2305.03195.
[37] S. Minaee et al., “Large Language Models: A Survey,” Feb. 20, 2024, arXiv: arXiv:2402.06196. doi: 10.48550/arXiv.2402.06196.
[38] G. Hinton, O. Vinyals, and J. Dean, “Distilling the Knowledge in a Neural Network,” Mar. 09, 2015, arXiv: arXiv:1503.02531. doi: 10.48550/arXiv.1503.02531.
[39] E. M. Bender, T. Gebru, A. McMillan-Major, and S. Shmitchell, “On the Dangers of Stochastic Parrots: Can Language Models Be Too Big? 🦜,” in Proceedings of the 2021 ACM Conference on Fairness, Accountability, and Transparency, in FAccT ’21. New York, NY, USA: Association for Computing Machinery, Mar. 2021, pp. 610–623. doi: 10.1145/3442188.3445922.
[40] Q. Xie, Z. Dai, E. Hovy, M.-T. Luong, and Q. V. Le, “Unsupervised Data Augmentation for Consistency Training,” Nov. 05, 2020, arXiv: arXiv:1904.12848. doi: 10.48550/arXiv.1904.12848.
[41] J. Kaplan et al., “Scaling Laws for Neural Language Models,” Jan. 23, 2020, arXiv: arXiv:2001.08361. doi: 10.48550/arXiv.2001.08361.
[42] K. Clark, U. Khandelwal, O. Levy, and C. D. Manning, “What Does BERT Look At? An Analysis of BERT’s Attention,” Jun. 11, 2019, arXiv: arXiv:1906.04341. doi: 10.48550/arXiv.1906.04341.
[43] K. Cho et al., “Learning Phrase Representations using RNN Encoder-Decoder for Statistical Machine Translation,” Sep. 03, 2014, arXiv: arXiv:1406.1078. doi: 10.48550/arXiv.1406.1078.
[44] T. B. Brown et al., “Language Models are Few-Shot Learners,” Jul. 22, 2020, arXiv: arXiv:2005.14165. doi: 10.48550/arXiv.2005.14165.
[45] C. Raffel et al., “Exploring the Limits of Transfer Learning with a Unified Text-to-Text Transformer,” Sep. 19, 2023, arXiv: arXiv:1910.10683. doi: 10.48550/arXiv.1910.10683.
[47] Y. Bengio, P. Simard, and P. Frasconi, “Learning long-term dependencies with gradient descent is difficult,” IEEE Trans. Neural Netw., vol. 5, no. 2, pp. 157–166, Mar. 1994, doi: 10.1109/72.279181.

A Machine Learning-Based Proactive Framework for Wildfire Risk Prediction and Early Warning Using Historical Fire Records

Mohanad Ehab Al-Hajj
AHMED NAJIB AHMED ABDO
ABDULRAHMAN MOHAMMED AHMED ALHALQI
Ammar Mohammed Ali Al-Qadasi*

Dept. of Computer Engineering, Karabuk University, Türkiye * abdoibrahim19981998@gmail.com

ABSTRACT

This paper presents a proactive wildfire risk prediction model developed for the Karabük region in Türkiye, focusing on automated weather-based risk assessment. The methodology utilized May–October seasonal weather records from 2012 to 2025, leveraging Open-Meteo daily aggregates as inputs and a processed Fire Weather Index (FWI) target for supervised learning. The research is centered on two primary objectives. First, to predict the FWI and identify high-risk days using machine learning as an efficient alternative to traditional calculation methods. This was achieved through a two-stage pipeline: a HistGradientBoostingRegressor to estimate continuous FWI values, followed by a RandomForestClassifier to categorize risk levels. Second, to flag high-risk days early by utilizing a clear, data-driven threshold derived directly from historical fire records (FWI $\geq$ 35). The results demonstrate that this machine learning approach can effectively and rapidly forecast fire weather conditions and proactively alert emergency managers to critical risk days. The study serves as a robust academic prototype for predictive early warning, providing a scalable foundation for regional wildfire decision-support.

Keywords – wildfire prediction, Fire Weather Index (FWI), machine learning, early warning system, risk classification..

I. INTRODUCTION

Forest fires are among the most severe natural hazards affecting Türkiye, causing the destruction of thousands of hectares annually and threatening human lives, wildlife, critical infrastructure, and the national economy [1]. The combination of hotter summers, prolonged drought periods, and human activities has significantly increased both the frequency and severity of these wildfires [2, 3].

Traditional wildfire monitoring and early warning practices in Türkiye rely heavily on fixed lookout towers, ground patrols, and reports from local communities. Although these methods are useful, they suffer from fundamental limitations: they are highly labor-intensive, dependent on human vigilance, and often detect fires only after they have grown to a dangerous size [4]. In contrast, there is an urgent need to transition from a reactive firefighting approach to a proactive, risk-driven monitoring strategy [5]. The early and reliable development of risk prediction systems can significantly reduce burned areas, economic losses, and ecological damage [5], enabling relevant authorities to direct their limited resources toward the most vulnerable periods and locations.

Despite the increasing impact of wildfires, current early warning practices still suffer from operational gaps, most notably that fire risk is not always quantified in a continuous and operational manner to support daily monitoring decisions for the target region [1]. To address this gap, this

research paper focuses on developing a proactive prediction model primarily centered on two specific objectives: the first is to predict the Fire Weather Index (FWI) and high-risk days using Machine Learning techniques instead of relying on traditional calculation methods. The second objective is to flag high-risk days early by utilizing a clear threshold derived from historical fire records.

To achieve these objectives, this study leverages a rich dataset covering 14 fire seasons (from 2012 to 2025). The main contribution of this paper lies in presenting a prediction pipeline based on a two-stage prediction model for daily risk estimation: the first stage utilizes a HistGradientBoostingRegressor, followed by a second stage that employs a RandomForestClassifier as a safety mechanism to ensure classification accuracy and identify days that exceed the historical risk threshold.

II. Literature Review

Feedforward This section reviews the most relevant recent studies aligned with the objectives of this research, focusing on the use of machine learning techniques for wildfire risk prediction, estimation of the Fire Weather Index (FWI), and identification of high-risk days based on historical and environmental data.

A. Wildfire Risk Prediction and Fire Weather Index Studies

Recent studies have sought to improve the accuracy of wildfire risk prediction by transitioning from traditional models to advanced machine learning approaches. For instance, Kondylatos et al. [7] designed deep learning models based on ConvLSTM networks to forecast short-term wildfire danger using spatio-temporal environmental variables. The ConvLSTM configuration demonstrated clear improvements over traditional methods, achieving AUROC values of approximately 0.93, while highlighting the most influential predictors using interpretability tools.

In another approach, Shmuel and Heifetz [8] introduced a machine learning-based Fire Weather Index (MLFWI), trained on global ignition-related data using gradient-boosted trees (XGBoost). This index achieved very high performance compared to traditional indices, with ROC–AUC values close to 0.99 and accuracy reaching 0.96, reflecting its ability to capture non-linear interactions between drivers such as heat, drought, and wind.

In the context of local fire occurrence prediction, Zaidi [9] studied wildfire occurrence in Algerian forests using several machine learning classifiers based on weather data and FWI-related inputs. The Artificial Neural Network (ANN) model reported accuracy of approximately 0.967 and an F1-score of 0.971, demonstrating that high predictive performance is achievable

even with limited local data, which supports the concept of defining critical thresholds for predicting high-risk days.

Li et al. [10] combined local sensing with neural networks to estimate wildfire risk indices, focusing on microclimate variables. They reported strong agreement with reference indices, with a correlation coefficient of approximately 0.89, highlighting the ability of neural models to capture fine-scale conditions that may not appear in coarse-grained weather products, thereby improving index estimation accuracy.

To enhance model interpretability, Sengupta and Woodford [11] proposed an explainable machine learning workflow for wildfire occurrence prediction. They evaluated multiple models and used SHAP-based analysis to interpret feature importance, reporting very strong classification performance (F1 up to approximately 0.99). This approach provides a systematic framework for feature selection and interpreting "why risk is high," which helps justify threshold choices based on historical records.

B. Limitations of Previous Studies

Despite the significant progress in using machine learning for wildfire risk prediction, previous studies still face several challenges. The most prominent limitation is the difficulty of transferring models trained on specific geographic regions (such as the Eastern Mediterranean) to other areas without retraining [7, 9]. Additionally, advanced models such as XGBoost and ConvLSTM require large, high-quality datasets and substantial computational resources [8, 10]. Some studies focus on predicting the index (FWI) rather than directly observed fire events, making the operational linkage indirect in certain cases [10]. Finally, many models rely primarily on meteorological and FWI data, sometimes overlooking fuel and topography factors, or lack validation across multiple years and regions [8, 11].

C. Comparative Summary of Studies

Table 1 provides a brief comparison of the reviewed prediction studies, showing the methodologies used, main contributions, and achieved performance levels, which form the theoretical and technical foundation for the proposed model in this paper.

Table 1: Comparative Summary of Wildfire Risk Prediction and Fire Weather Index Studies

Study	Year	Method	Main Contribution	Achieved Performance
Kondylatos et al. [7]	2022	ConvLSTM	Daily wildfire danger prediction from spatio-temporal inputs	AUROC ≈ 0.93; F1 ≈ 0.8
Shmuel & Heifetz [8]	2023	XGBoost (MLFWI)	Machine learning-based fire weather index	AUC ≈ 0.99; Acc ≈ 0.96
Zaidi [9]	2023	ANN	Local wildfire occurrence prediction based on weather and FWI data	Acc ≈ 0.967; F1 ≈ 0.971
Li et al. [10]	2023	MLP (Neural Nets)	Risk index estimation from fine-grained environmental and climate measurements	MSE = 1.116; R ≈ 0.89
Sengupta & Woodford [11]	2025	XAI + HPO	Explainable models for wildfire occurrence with feature importance analysis	F1 up to ~ 0.99; RMSE ≈ 1.9

III. METHODOLOGY

A. Methodology Overview

The methodology of this research is built upon developing a proactive prediction model for wildfire risk estimation based on meteorological variables, leveraging machine learning techniques rather than traditional calculation methods. The model aims to provide daily predictions of the Fire Weather Index (FWI) for the Karabük region and to flag high-risk days based on a clearly defined historical threshold. The system does not compute the traditional Canadian FWI equations at runtime; instead, it relies on a trained machine learning model to estimate this index directly from available weather variables and engineered temporal features.

B. Study Region and Data Sources

The Karabük region in Türkiye (latitude 41.2061, longitude 32.6204) was selected as the pilot study area. Meteorological variables were collected from the Open-Meteo platform. The Fire Weather Index (FWI) serves as the target variable in the supervised learning process, derived from the official fire danger data provided by the CEMS/Copernicus system [19]. This target is used to train the model on how to relate daily weather patterns and recent drying conditions to the official risk index.

C. Data Collection Period and Seasonal Filtering

The processed dataset covers seasonal records from May 1, 2012, to October 31, 2025, comprising a total of 2,576 daily records. The data was restricted to the months of May through October for methodological reasons, as including low-risk winter months could reduce the model's ability to capture the critical weather patterns associated with the fire season, ensuring

that the model remains focused on the period when operational preparedness is essential.

D. Daily Weather Aggregation and Model Input Alignment

The model inputs are based on daily aggregate values, including: maximum temperature, minimum relative humidity, maximum wind speed, precipitation sum, mean cloud cover, shortwave radiation sum, and reference evapotranspiration, in addition to mean soil moisture (0–7 cm). This approach was designed to ensure consistency between training and operational data, as Stage 1 was trained on these aggregated daily variables.

E. Data Cleaning and Alignment

The data was organized as chronologically ordered daily records, with column names standardized to match the model feature schema. The high-risk classification target was derived from the official FWI value, where a day is classified as high risk if FWI $\geq$ 35. To ensure model integrity, temporal and historical features (rolling features) were computed using shifted time series to represent only past conditions, thereby preventing target leakage.

F. Feature Engineering

The model employs a locked schema of 34 input features, designed to approximate the environmental behavior represented by the official FWI. These features are categorized as follows:

Raw features (8 features): Represent immediate weather conditions (temperature, humidity, wind, precipitation, solar radiation, and soil moisture).

Seasonal and cyclical features: Enable the model to learn seasonal patterns without treating days as linear values.

Derived features: Such as vapor pressure deficit (VPD), fuel drying rate, and squared wind index, to represent moisture stress and spread potential.

Rolling history features: Capture accumulated precipitation, persistent heat, and low-humidity periods over 3, 7, and 30 days.

Exponentially weighted memory features (EWMA): Assign greater weight to recent weather while preserving short-term historical influence.

Dryness persistence feature: Measures the duration of recent dry conditions.

G. Stage 1: Wildfire Risk Regression Model

Stage 1 represents the core numerical prediction model, employing a HistGradientBoostingRegressor to predict the continuous FWI value from the 34 extracted features. Regression is appropriate for this stage since FWI is a continuous intensity index, and predicting a continuous value preserves information about the severity of weather conditions and indicates how close a day is to the high-risk boundary. The model was implemented with missing

value handling and sample weighting to assign greater importance to medium and high FWI cases, given their critical operational significance.

H. Stage 2: High-Risk Safety Classifier

To achieve the second objective of identifying high-risk days early using a clear threshold, a safety-oriented classification layer (Stage 2) was built on top of the regression outputs. This classifier uses a RandomForestClassifier, trained on the outputs of Stage 1 along with three supporting features: relative humidity, wind speed, and fuel drying rate.

The goal of this stage is binary classification based on the historical threshold (FWI ≥ 35). If the predicted FWI ≥ 35, the day is directly classified as high risk. However, if the value falls within the grey zone (28 to below 35), the classifier layer can promote the day to "high risk" if the risk probability exceeds the configured probability threshold (0.10). This two-stage structure ensures that the continuous risk intensity is preserved in Stage 1, while Stage 2 adds a decision-support layer focused on avoiding missed high-risk cases, thereby achieving early and effective alerting.

İ. Methodology Summary

This paper presents a regional two-stage decision-support methodology for the Karabük region. The processed FWI target provides the supervised training signal, while weather variables and engineered temporal features offer practical operational inputs. Stage 1 estimates the continuous FWI using machine learning, and Stage 2 adds a safety layer to identify high-risk days based on a clear historical threshold (FWI ≥ 35), achieving the objectives of accurate prediction and early alerting without relying on traditional calculation methods.

IV. RESULTS AND DISCUSSION

A. Stage 1 Regression Results

The Stage 1 model was evaluated as a continuous regression model for estimating the Fire Weather Index (FWI). The task of this model is to estimate the daily index value based on a locked set of 34 engineered features. The adopted model is the HistGradientBoostingRegressor, and its output (predicted_fwi) is the numerical risk estimate that is passed to the Stage 2 decision layer.

Table 2: Stage 1 Regression Performance for Continuous FWI Prediction

Evaluation Setting	RMSE	MAE	R^2	Interpretation
Out-of-fold validation	4.111	2.918	0.835	Temporal validation result across training folds; shows strong fit while respecting chronological evaluation.
2025 holdout test	7.259	5.098	0.819	Later-season holdout result; error increases, but the model still captures most variation in the FWI target.

Table 2 reports the verified performance metrics for Stage 1. The out-of-fold validation results showed values of RMSE = 4.111, MAE = 2.918, and R^2 = 0.835. For the held-out test data for the 2025 season, the results were RMSE = 7.259, MAE = 5.098, and R^2 = 0.819. This slight increase in error is expected when testing the model on a chronologically later season; however, the R^2 value still indicates that the model captures the majority of the variance in the FWI target value.

Fig.1 shows that the selected regression approach follows the general trend of the observed FWI values. This result should not be interpreted as an exact reproduction of fire risk, but rather as a decision-support estimate that preserves useful information about how close a day is to the high-risk boundary.

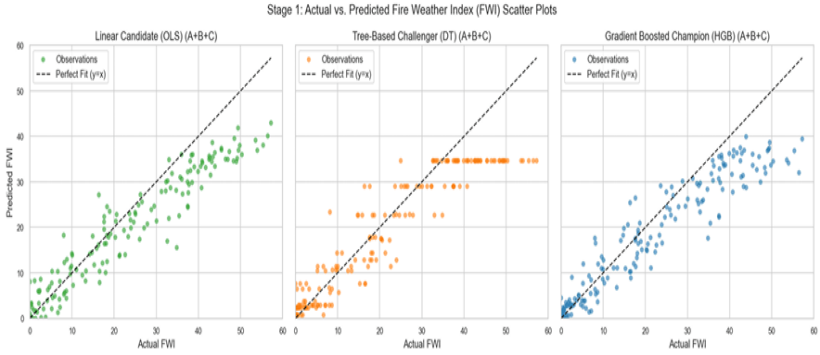

Fig.1: Stage 1 actual versus predicted FWI comparison for candidate regression models.

The residual analysis shown in Fig.2 is critical, as wildfire risk decisions are sensitive to larger errors at high FWI values. The model is useful for daily risk estimation, but the residual pattern also supports the need for a second safety-oriented classification layer.

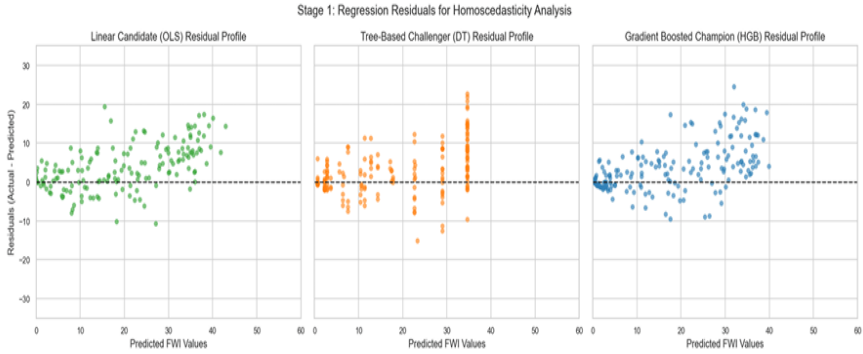

Fig.2: Stage 1 residual analysis showing prediction error behaviour across FWI values.

Fig.3 summarizes the benchmark evidence for Stage 1. The combined use of RMSE, MAE, and R^2 is necessary, as a single metric cannot describe both the average error and the explained variance simultaneously.

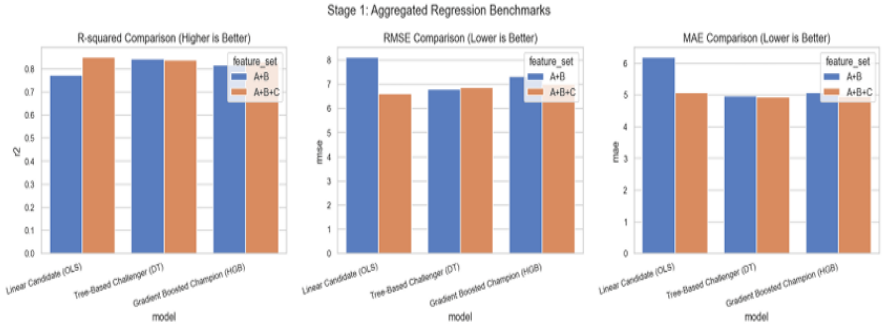

Fig.3: Regression benchmark comparison using R2, RMSE, and MAE.

B. Stage 2 High-Risk Classification Results

Stage 2 was evaluated as a classification layer for high-risk days. This classifier uses the RandomForestClassifier algorithm with four inputs: (predicted_fwi), (rh), (ws), and (fuel_drying_rate). The classification target is (target_ge_35), where a day is classified as high risk when FWI ≥ 35. The final stored probability threshold is 0.10.

Table 3 shows the final classifier values. At the 0.10 threshold, the Stage 2 classifier achieved Precision = 0.849, Recall = 0.900, F1-score = 0.874, and overall Accuracy = 0.929, with 45 true positives, 8 false positives, 5 false negatives, and 126 true negatives.

Table 3: Final Stage 2 high-risk classification result at probability threshold 0.10

Threshold	Precision	Recall	F1-score	Accuracy	TP	FP	FN	TN
0.10	0.849	0.900	0.874	0.929	45	8	5	126

Fig.4 is more informative than relying solely on overall Accuracy, as high-risk days are less frequent than normal days. In this application, the Recall metric is particularly important; a false negative may reduce attention on a genuinely dangerous day. However, Precision remains important because excessive false alarms may reduce operator trust and increase the monitoring burden.

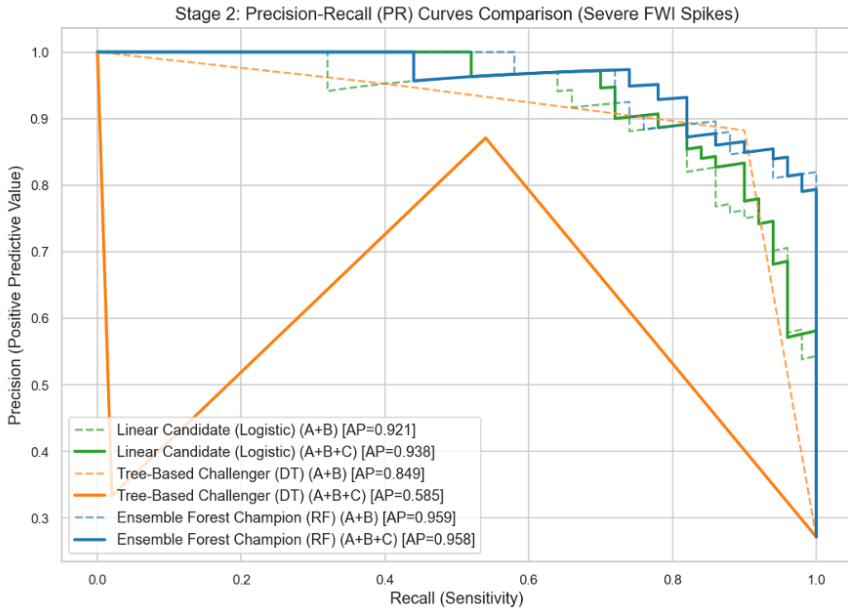

Fig.4: Stage 2 precision–recall comparison for imbalanced high-risk classification.

The confusion matrix in Fig.5 uses the final report values at the 0.10 threshold. This distinction highlights the importance of not mixing ablation results with the final deployed threshold, ensuring an accurate representation of classifier performance.

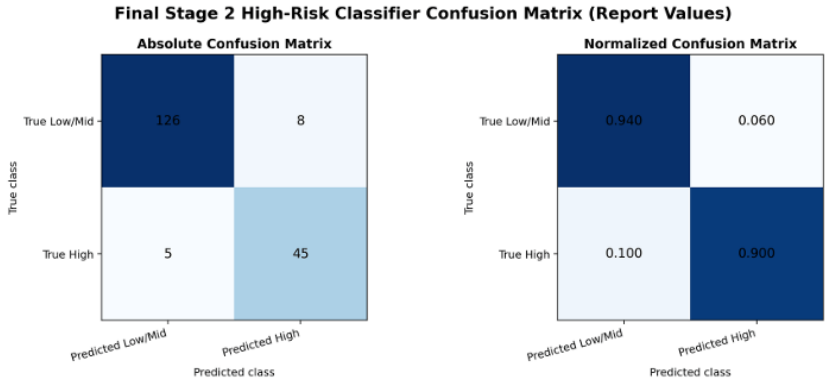

Fig.5: Final Stage 2 high-risk classifier confusion matrix using the report threshold of 0.10

Fig.6 and Fig.7 show that the final decision layer is not a black-box replacement for the regression model, but rather an integrated safety layer that uses Stage 1 outputs along with humidity, wind, and fuel drying rate information to reduce missed high-risk days near the decision boundary.

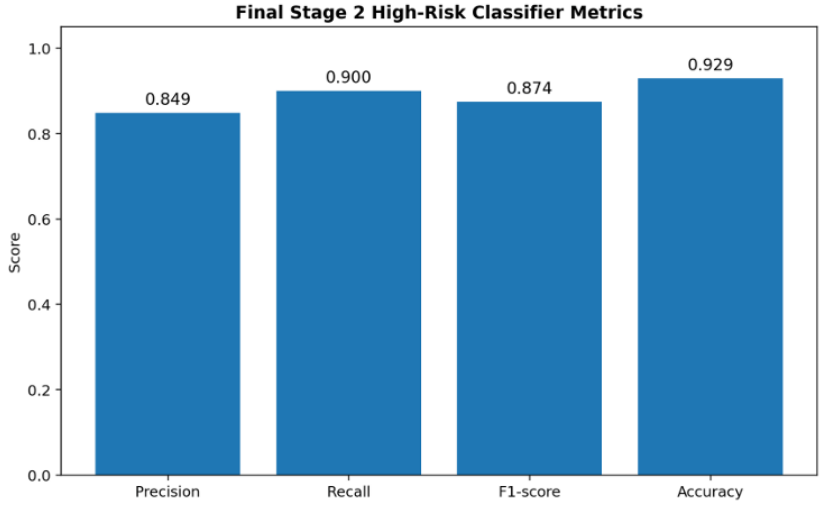

Fig.6: Final Stage 2 classifier metrics at threshold 0.10

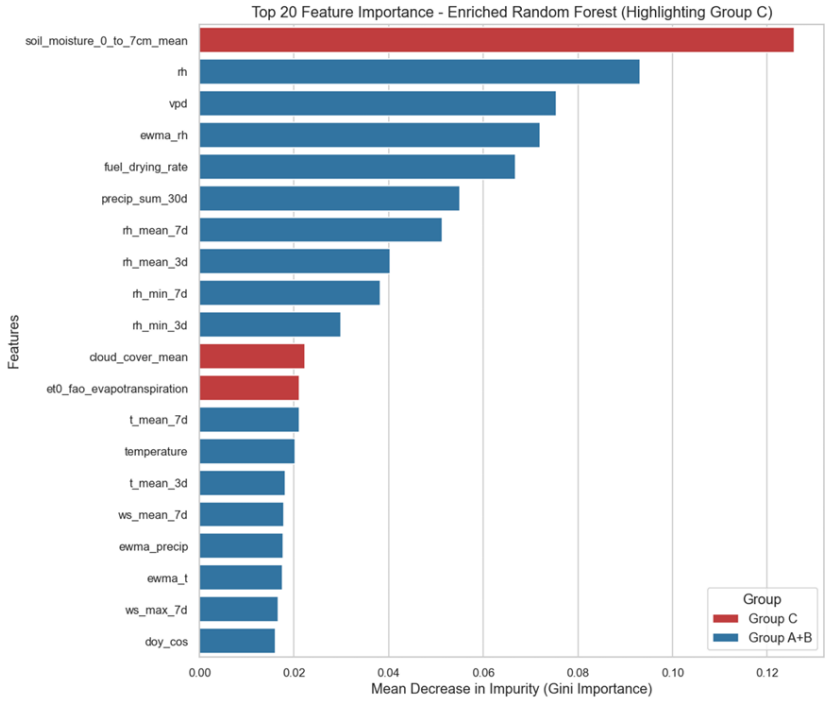

Fig.7: Feature importance evidence for the enriched Random Forest classifier, highlighting the contribution of the support features

C. Limitations

The Stage 1 regression model still contains a margin of error in prediction. Although the cross-validation and held-out data results show useful performance, unusual weather patterns and days with very high FWI values can produce larger errors. Therefore, the predicted FWI value should be interpreted as a decision-support estimate, not as a replacement for official fire danger products.

On the other hand, the Stage 2 classifier improves decision support for high-risk days, but it does not completely eliminate false positives or false negatives. False positives can increase the monitoring burden, while false negatives are more dangerous as they may reduce attention during hazardous conditions.

D. Discussion

The results answer the research questions by demonstrating the system's effectiveness in daily wildfire risk prediction for the Karabük region. The two-stage design provides a key advantage by separating risk severity from high-risk day decision support. The continuous FWI estimation helps operators understand the risk level, while the Stage 2 classifier focuses on reducing missed high-risk days near the decision boundary. This approach is

more informative than a simple binary model, as it preserves both the estimated risk level and the operational decision.

V. CONCLUSION

This research developed a proactive prediction model for wildfire risk, achieving two primary objectives. First, the study successfully replaced traditional Fire Weather Index (FWI) calculation methods with an efficient machine learning framework. The proposed architecture utilized a two-stage approach: a HistGradientBoostingRegressor to accurately predict continuous FWI values from historical weather data, followed by a RandomForestClassifier to identify high-risk days. This methodology enables automated, rapid forecasting, overcoming the complexities and limitations of conventional physical models. Second, the objective of early warning was achieved by establishing a clear alert threshold derived directly from historical fire records (FWI $\geq$ 35). This criterion enabled the system to proactively flag and classify high-risk days, providing decision-makers and emergency managers with a critical time window to implement prevention and preparedness measures before fire outbreaks occur. Looking ahead, future work will focus on enhancing model accuracy by integrating additional environmental and topographical variables, as well as dynamically calibrating the alert threshold based on operational feedback. These enhancements aim to ensure higher reliability and improved generalizability for deployment in operational early warning systems.

REFERENCES

[1] E. Çolak and F. Sunar, "Evaluation of forest fire risk in the Mediterranean Turkish forests: A case study of Menderes region, Izmir," *International Journal of Disaster Risk Reduction*, vol. 45, p. 101479, 2020, doi: 10.1016/j.ijdrr.2020.101479.

[2] M. Göltaş, H. Ayberk, and O. Küçük, "Forest fire occurrence modeling in Southwest Turkey using MaxEnt machine learning technique," *iForest – Biogeosciences and Forestry*, vol. 17, no. 1, pp. 10–18, 2024, doi: 10.3832/ifor4321-016.

[3] B. Ekberzade, T. Görüm, F. Karabacak, S. S. Akay, et al., "Up in flames: the human factor behind a megafire in Mediterranean Türkiye," *npj Natural Hazards*, vol. 2, p. 65, 2025, doi: 10.1038/s44304-025-00120-4.

[4] M. Bugarić, D. Krstinić, L. Šerić, and D. Stipaničev, "Current trends in wildfire detection, monitoring and surveillance," *Fire*, vol. 8, no. 9, p. 356, 2025, doi: 10.3390/fire8090356.

[5] A. Mohapatra and T. Trinh, "Early wildfire detection technologies in practice—A review," *Sustainability*, vol. 14, no. 19, p. 12270, 2022, doi: 10.3390/su141912270.

[6] D. Rjoub, A. Alsharoa, and A. Masadeh, “Unmanned-aircraft-system-assisted early wildfire detection with air quality sensors,” *Electronics*, vol. 12, no. 5, p. 1239, 2023, doi: 10.3390/electronics12051239.
[7] S. Kondylatos, I. Prapas, M. Ronco, I. Papoutsis, et al., “Wildfire danger prediction and understanding with deep learning,” *Geophysical Research Letters*, vol. 49, no. 17, p. e2022GL099368, 2022, doi: 10.1029/2022GL099368.
[8] A. Shmuel and E. Heifetz, “Developing novel machine-learning-based fire weather indices,” *Machine Learning: Science and Technology*, vol. 4, no. 1, p. 015029, 2023, doi: 10.1088/2632-2153/acc008.
[9] A. Zaidi, “Predicting wildfires in Algerian forests using machine learning models,” *Heliyon*, vol. 9, no. 7, p. e18064, 2023, doi: 10.1016/j.heliyon.2023.e18064.
[10] L. Li, A. Sali, N. K. Noordin, A. Ismail, and F. Hashim, “Prediction of peatlands forest fires in Malaysia using machine learning,” *Forests*, vol. 14, no. 7, p. 1472, 2023, doi: 10.3390/f14071472.
[11] A. Sengupta and B. J. Woodford, “Recent advances in explainable Machine Learning models for wildfire prediction,” *Applied Computing and Geosciences*, vol. 27, p. 100266, 2025, doi: 10.1016/j.acags.2025.100266.