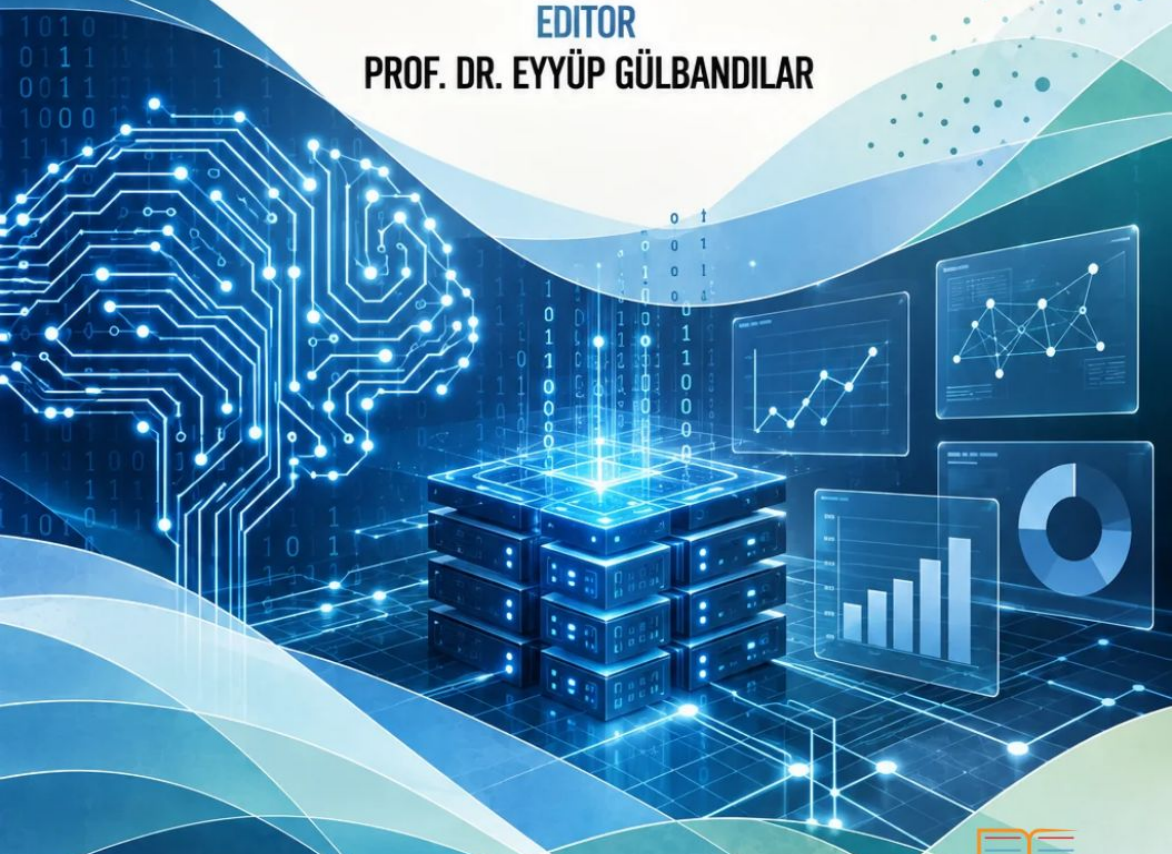
INTELLIGENT COMPUTING ARCHITECTURES AND DATA-DRIVEN APPLICATIONS

EDITOR
PROF. DR. EYYÜP GÜLBANDILAR



BİDGE Yayınları

BİDGE Yayınları

Intelligent Computing Architectures and Data-driven Applications

Editor: Prof. Dr. EYYÜP GÜLBANDILAR

ISBN: 978-625-5057-85-3

1st Edition
Page Layout By: Gözde YÜCEL
Publication Date: 2026-10-25
BİDGE Yayınları

All rights reserved. No part of this work may be reproduced in any form or by any means, except for brief quotations for promotional purposes with proper source attribution, without the written permission of the publisher and the editor.

Certificate No: 71374
All rights reserved © BİDGE Yayınları
www.bidgeyayinlari.com.tr - bidgeyayinlari@gmail.com
Krc Bilişim Ticaret ve Organizasyon Ltd. Şti.
Güzeltepe Mahallesi Abidin Daver Sokak Sefer Apartmanı No: 7/9 Çankaya / Ankara

CONTENTS

ANALOG IN-MEMORY COMPUTING ACCELERATORS: ARCHITECTURE, DATAFLOW, AND THE SIMULATION TOOLCHAIN 1

BAKİ GÖKGÖZ

CONTINUAL LEARNING FOR GEARBOX FAULT DIAGNOSIS UNDER VARYING LOAD CONDITIONS: A COMPARATIVE STUDY OF FINE-TUNING, EXPERIENCE REPLAY, AND KNOWLEDGE DISTILLATION 25

CANAN TASTİMUR

MACHINE LEARNING-BASED SENTIMENT ANALYSIS OF CONVERSATIONAL AI IN TURKISH MICROBLOGS .. 40

AYHAN AYDIN

ARTIFICIAL INTELLIGENCE FOR CYBER RESILIENCE IN DIGITAL GOVERNMENT 62

ÖMER SEVİNÇ

CHAPTER 1

Analog In-Memory Computing Accelerators: Architecture, Dataflow, and the Simulation Toolchain

Baki GÖKGÖZ[1]

Abstract

Energy-efficient inference has become a central concern in deep learning, and analog in-memory computing (AIMC) is among the architectures proposed to address it. The approach targets the memory wall, the bottleneck that limits conventional von Neumann machines, by performing multiply–accumulate operations inside dense arrays of memory devices rather than moving weights to a separate processing unit. This chapter serves as an architecture-level introduction to how these accelerators are built and how they are evaluated before fabrication. The crossbar primitive and its surrounding circuits are described first, followed by the mapping of neural-network layers onto a grid of tiles and the dataflow decisions that govern scheduling across the chip. The open simulation tools used to reason about accuracy, energy, and area are then reviewed, along with the assumptions that determine how far their results can be trusted. No specific design is advocated and no new

[1] Asst. Prof., Gümüşhane University, Department of Computer Technologies, Torul Vocational School, Orcid: 0000-0001-8166-6282

measurements are presented; the aim is a coherent overview of the field and the vocabulary required to read its literature.

Keywords: analog in-memory computing; crossbar accelerator; dataflow; hardware simulation; design-space exploration; deep-learning inference.

Introduction

Where data is stored, rather than how quickly it is multiplied, sets the real cost in contemporary deep-learning hardware (Horowitz, 2014; Sze et al., 2017). A network keeps its weights in memory, and a conventional processor must move each weight across a bus to an arithmetic unit for every multiply–accumulate operation, then return the result. For a model with millions of parameters, the energy spent on this traffic exceeds that of the computation by a wide margin (Sze et al., 2017). The underlying problem is a long-standing mismatch between fast arithmetic and comparatively slow memory access, commonly termed the memory wall. Circumventing it is precisely what makes architectures that compute inside memory, instead of alongside it, worth pursuing (Figure 1) (Ielmini & Wong, 2018; Sebastian et al., 2020).

Analog in-memory computing follows that path (Hu et al., 2018; Sebastian et al., 2020). Its principle can be stated compactly: memory devices are placed on a grid, and the electrical behaviour of the grid itself carries out the arithmetic. If voltages drive the rows and each device is set to a chosen conductance, then the current collecting on a given column emerges as the weighted sum of the inputs, a direct consequence of Ohm's and Kirchhoff's laws. One read of the array thus corresponds to one complete matrix–vector product. Because the weights are never relocated, the energy formerly spent on movement is largely removed, and the columns are resolved concurrently rather than sequentially.

A simple estimate conveys the magnitude of the saving. Suppose a weight matrix contains a million entries. A conventional machine retrieves a million weights, carries out a million multiplications, accumulates the results, and incurs a data-movement cost at each retrieval. A crossbar that stores the same million weights reaches the result in only a few analog steps, holding the weights in place and letting the wires perform the accumulation. The many cycles and memory accesses of a digital machine are compressed into a single array read. This compression, not any gain in the speed of one multiplication, is where the advantage originates, which also explains why the method helps most in the large, weight-dense layers in which data movement is especially expensive.

The purpose of this chapter is to explain how accelerators are organized around that primitive and how they are investigated. The treatment is intentionally broad. Device physics is not derived, no new design is put forward, and no experiments are reported. The intended outcome is a working understanding of three things: the constituents of an AIMC accelerator, the manner in which a network is mapped onto it, and the simulation tools the field employs to anticipate behaviour ahead of fabrication. Equipped with that understanding, a reader will find the diagrams and claims of the primary literature considerably more approachable.

It is useful to mark one contrast early, because it accounts for both the appeal of the approach and its chief difficulty. Digital accelerators, whether general-purpose GPUs or dedicated tensor engines, still keep storage and computation apart. Their efficiency derives from situating numerous arithmetic units close to fast on-chip memory and from reusing operands wherever possible, yet each weight is still fetched and each product is still produced by a digital multiplier. AIMC breaks with this arrangement more decisively: multiplication is carried out by the device instead of a logic gate, and summation by a wire instead of an adder tree. Thousands of

multiply–accumulate operations can run on a single array with almost no data movement and no explicit arithmetic circuitry. The accompanying difficulty is that analog computation is inherently approximate, and a substantial part of the engineering covered in this chapter is devoted to containing that approximation without forfeiting the efficiency that justified the approach to begin with.

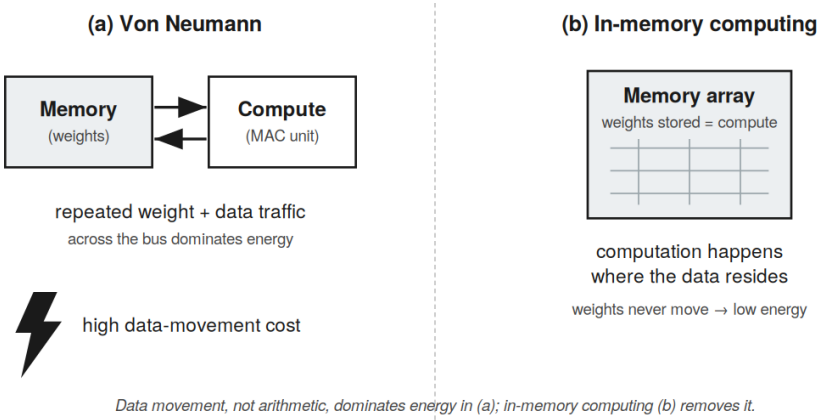


Figure 1. Data movement, not arithmetic, dominates energy in a conventional von Neumann machine (a), where weights and activations repeatedly traverse the bus between memory and the compute unit. In-memory computing (b) performs the computation where the data resides, so the weights never move.

The chapter is structured as follows. The crossbar and its peripheral circuits are introduced in Section 2. Section 3 explains how a network's layers are partitioned and placed on a grid of tiles. Dataflow and scheduling are taken up in Section 4. Section 5 reviews the simulation tools and the quantities they estimate, and Section 6 examines the assumptions that limit the reliability of those estimates. Section 7 surveys the directions the field is taking, and Section 8 offers a conclusion. Throughout, the discussion remains at the level of architecture and method, and presumes only that the reader is

familiar with neural networks and with the matrix multiplication that underlies most of their computation.

The Crossbar Primitive and Its Periphery

The array

An AIMC accelerator is built upon the crossbar array (Figure 2). Horizontal wires, the word lines or rows, cross vertical wires, the bit lines or columns, and a programmable memory device occupies each intersection. The value held by a device is physical, in most cases an electrical conductance. The role can be filled by a variety of device technologies, among them resistive RAM, phase-change memory, floating-gate transistors, and the more recent ferroelectric and magnetic devices (Ielmini & Wong, 2018; Yu, 2018). Throughout this chapter the device is treated as a black box, namely a programmable and roughly linear resistor whose conductance represents a weight.

Suppose the rows are driven by input signals while the columns are held at a shared potential. Every device then conducts a current set by the product of its input and its conductance, and these currents combine on the column wire. What appears on a column is consequently the dot product between the input vector and the conductances stored in that column. Carrying this out across all columns simultaneously realizes a complete matrix–vector multiplication in one analog step, and that is where the efficiency of the array originates (Hu et al., 2018; Yao et al., 2020).

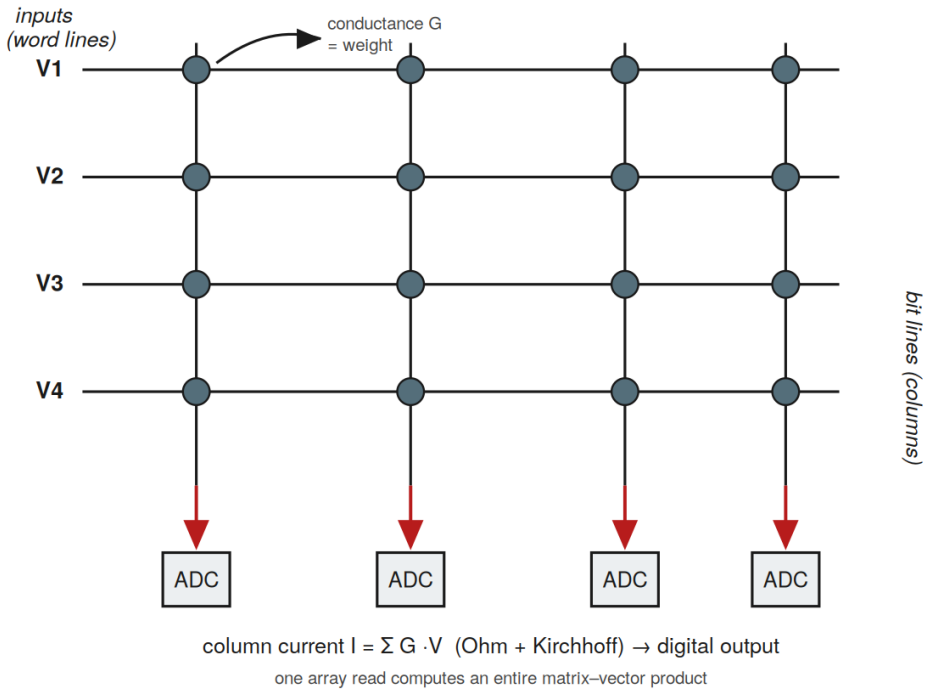


Figure 2. The crossbar primitive. Input signals drive the horizontal word lines; a programmable device at each intersection stores a conductance that encodes a weight. The currents contributed by all devices on a column sum along the vertical bit line, and the column current, by Ohm's law and Kirchhoff's current law, is the dot product of the inputs with that column's conductances. A single array read therefore computes an entire matrix–vector product, which is digitized by the column converters.

Two practical issues surface at once. A conductance cannot take a negative value, but a weight can. The standard remedy represents each weight as the difference between two conductances, realized with a column pair or with a reference column whose current is subtracted (Chi et al., 2016; Shafiee et al., 2016). The second issue is limited dynamic range: because one device stores little precision,

a high-precision weight is spread across several devices or columns and reassembled with suitable scaling. Although invisible at the algorithm level, these encoding choices have a pronounced effect on the area and energy of the array.

A third consideration belongs to the input side rather than the weight side. The signals that drive the rows must represent the activations entering the layer, and their fidelity places a ceiling on the fidelity of the entire computation. Supplying activations at full precision in a single step makes the input circuitry costly, whereas supplying them bit by bit across several cycles simplifies the circuitry at the expense of speed. The array is unaffected by this decision, but the design surrounding it is not, and the resulting tension between input precision and speed is a recurring feature of AIMC architecture.

The economy of the analog summation is also worth appreciating. In a digital accelerator the products along a column are added by a tree of adders whose depth increases with the number of terms. A crossbar instead performs the summation in one stroke on the shared column wire, which collects the current from every device together. A sum over hundreds or thousands of terms is thereby completed in a single electrical event, without an adder tree and without intermediate storage. This is the crux of the approach, and it is the reason so much design effort is directed at preventing the surrounding circuitry from dissipating the advantage the array supplies.

Peripheral circuits

On its own, an array of conductances achieves nothing. Circuits are needed to drive it and to read its results, and a large share of an accelerator's complexity and cost resides in that periphery.

At the input, the digital activations entering a layer have to be turned into signals the array will accept. This is frequently done

by encoding each input as a train of pulses, or as a sequence of bits delivered over several cycles, an arrangement known as bit-serial input. Such encoding exchanges time for simplicity, since greater input precision calls for more cycles per operation.

At the output, each column yields an analog current that must become a digital number before it can be stored, accumulated across tiles, or forwarded to the next layer. The conversion is carried out by an analog-to-digital converter placed on the column (Ankit et al., 2019; Shafiee et al., 2016). It is one of several peripheral components, which also include sample-and-hold circuits, current-to-voltage conversion, and the shift-and-add logic that reassembles bit-serial and multi-column partial results. The salient point is that, taken together, the periphery accounts for a large portion of the chip's area and energy and remains an active subject of circuit research. These components are treated here by what they do rather than by their transistor-level realization.

Sharing the periphery across many rows distributes its cost over many operations, which is part of the reason AIMC can remain efficient in spite of the conversion overhead. The benefit of sharing is bounded, however. A converter that seems inexpensive when amortized over a wide array can come to dominate once arrays become small, so the balance between array size and peripheral cost is a persistent concern. Wide arrays amortize the periphery better but are more difficult to drive accurately and more susceptible to the parasitic resistance of long wires; narrow arrays are electrically cleaner yet pay a heavier peripheral cost per operation. The architecture is therefore a compromise between the analog core and the circuits that render it usable, and the two cannot be designed independently. This interdependence reappears below, in the treatment of system-level evaluation.

From array to tile

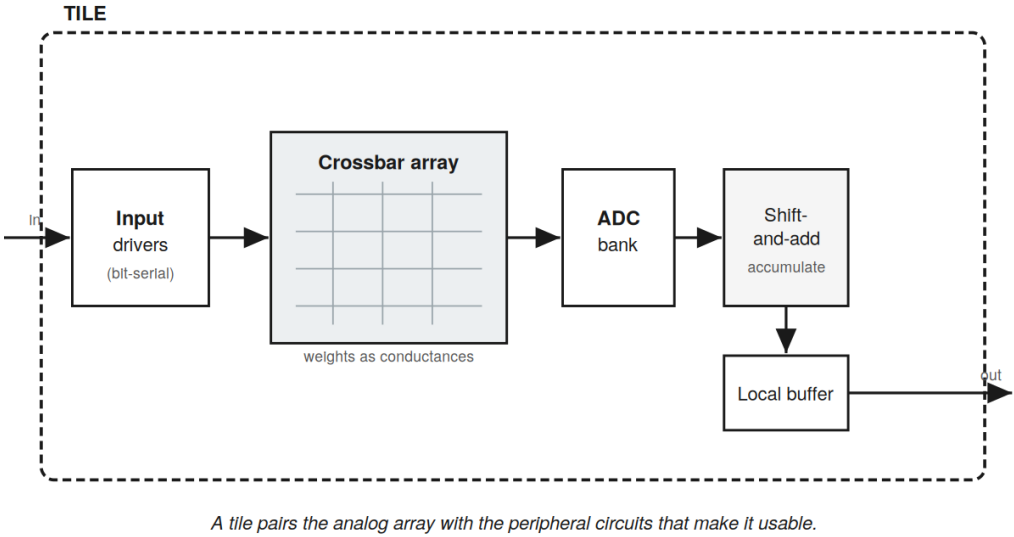


Figure 3. Anatomy of a tile. The crossbar array is wrapped in the peripheral circuits that make it usable: input drivers that present activations to the rows, a bank of converters that digitize the column currents, shift-and-add logic that recombines bit-serial and multi-column partial results, and a local buffer. A tile is the self-contained unit from which larger accelerators are built.

A single array is small. The weight matrices of real networks far exceed what one array can store, and in practice fabrication and signal-integrity constraints hold array dimensions down in any case. For that reason the practical unit of design is the tile rather than the bare array (Figure 3): an array accompanied by its input drivers, its column converters, its local accumulation logic, and a small buffer. When assigned a portion of a weight matrix, a tile computes the corresponding matrix–vector product independently. Larger accelerators are assembled by replicating tiles and linking them with an on-chip network (Ankit et al., 2019; Song et al., 2017), which brings up the subsequent question of how an entire network is distributed across many tiles.

Mapping Neural Networks onto the Fabric

Partitioning weights

At bottom, a neural-network layer is a matrix of weights. A fully connected layer is a matrix already, and a convolutional layer becomes one after its filters are unrolled so that the convolution is cast as a matrix multiplication (Ankit et al., 2019; Chen, Emer, & Sze, 2016). To map a layer onto the hardware is thus to place that matrix into the physical arrays (Figure 4).

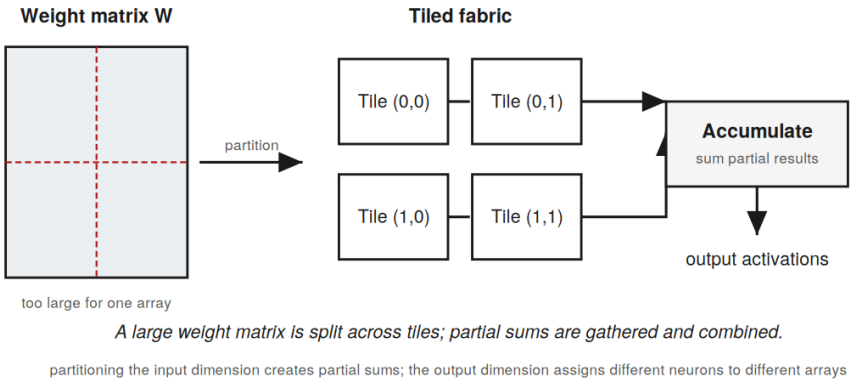


Figure 4. Mapping a layer onto the fabric. A weight matrix too large for one array is partitioned across several tiles. Partitioning the input dimension means each tile computes a partial sum over a subset of the inputs, and these partial sums are gathered and accumulated downstream; partitioning the output dimension assigns different output neurons to different tiles.

A matrix larger than a single array is divided into tiles along both of its dimensions. Division along the input dimension leaves no single array with a view of the entire input vector; each array forms a partial sum over the inputs it contains, and those partial sums are

combined downstream. Division along the output dimension is more straightforward, since it merely sends different output neurons to different arrays. In both cases the accelerator must route inputs to the appropriate arrays and then collect the partial results, a bookkeeping duty assigned to the on-chip interconnect and the accumulation logic.

Layer-to-tile assignment

Dividing one matrix addresses only part of the task. The designer must further decide how the layers of a complete network share the fabric, and two broad strategies present themselves. Under a weight-stationary layout, each layer is granted dedicated tiles that are programmed once and retained for the whole workload (Ankit et al., 2019; Song et al., 2017). This yields high throughput, since the layers can run concurrently in a pipeline, but it requires sufficient hardware to accommodate the entire network at the same time. Under a time-multiplexed layout, a smaller set of tiles is reprogrammed to host one layer after another. Area is reclaimed in exchange for the time and energy of rewriting weights, an arrangement that suits cases where devices are cheap to program and the workload can absorb the extra latency.

The choice ranks among the weightier decisions in AIMC design, and it is strongly conditioned by the device. Programming that is slow or expensive favours weight-stationary layouts, while fast and endurant devices make time-multiplexing viable. Rather than resolve the trade-off, this chapter marks it as one of the principal axes along which designs diverge.

The decision additionally shapes the workloads a design accommodates. An accelerator that holds a complete network in a weight-stationary fashion suits a fixed inference task applied to a stream of inputs, such as continuous classification at the edge, where the weights are programmed once and left unchanged. A time-

multiplexed design is preferable where many models must share a constrained hardware budget, or where one piece of hardware serves distinct tasks at distinct times. Neither dominates universally; each encodes an assumption about how the hardware will be used, and discerning that assumption is frequently the fastest route to a design's priorities.

The heterogeneity of a mapped network

A point that is easy to overlook yet matters a great deal is that a mapped network is not uniform. Its layers vary in size, in the statistics of their activations, in their location within the network, and in the number of arrays they require. A layout well suited to one layer can prove wasteful for another. A good deal of the refinement in AIMC architecture consists of acknowledging this unevenness and apportioning resources to match it, instead of handling every layer the same way. Here it is the principle that counts; the detailed per-layer allocation policies are the province of specialized design studies.

The unevenness extends beyond mere counts. The early convolutional layers operate on large spatial maps with few channels, whereas the later layers operate on small maps with many channels; the former generate numerous small matrix multiplications and the latter a smaller number of larger ones, with the fully connected layers at the end presenting yet another profile. Owing to these differences, the arrays hosting different layers experience markedly different utilization, so that a fabric scaled to the heaviest layer may lie idle while hosting the lightest. Maintaining balanced utilization, so that costly hardware is not left unused, is one of the less conspicuous challenges in AIMC, and it comes into view only once a network is regarded as something other than a uniform stack of identical operations.

Dataflow and Scheduling

Why dataflow matters

With the weights resident on tiles, the accelerator still has to move activations through them in a particular order. The term dataflow denotes how computation and data transfer are scheduled across the chip in space and time (Chen, Krishna, et al., 2017; Chen, Emer, & Sze, 2016). A single set of tiles can operate under markedly different dataflows, and the choice bears on latency, on the buffer capacity required, and on the energy consumed in relaying intermediate results.

Dataflow is readily undervalued, concerned as it is with scheduling rather than arithmetic, yet it frequently decides whether an accelerator attains its efficiency targets. Two designs with identical arrays and identical peripheral circuits may nonetheless diverge considerably in throughput and energy, for no reason other than that one schedules its work more adeptly. The arrays constitute a fixed resource; the dataflow dictates how thoroughly they are exploited and how much supplementary movement is required to keep them supplied. In this sense dataflow stands to an AIMC accelerator as an instruction schedule stands to a processor: absent from the block diagram, but pervasive in the measured outcome.

Pipelining across layers

Since a weight-stationary layout holds every layer at once, it lends itself naturally to pipelining. The moment the first layer completes one input, it may accept the next while the second layer processes the output of the first. In the steady state every layer is occupied with a different sample, and throughput stays high. The attendant cost is buffering: the output of each stage must be retained until the following stage is prepared to consume it, and any imbalance among stages produces bottlenecks in which faster layers wait on slower ones.

Spatial and temporal reuse

Reuse, too, is governed by the dataflow. Within a convolutional network a single filter is applied across many spatial positions, and a single input patch contributes to many outputs. A judicious dataflow capitalizes on this by keeping one operand stationary and streaming the other past it, so that no value travels more than it has to. The dataflow categories familiar from digital accelerators, among them the weight-stationary, output-stationary, and row-stationary schemes (Chen, Krishna, et al., 2017; Chen, Emer, & Sze, 2016), transfer to the analog setting, subject to the important proviso that AIMC fixes the weights physically within the array and so restricts the available options. Grasping a given accelerator often amounts to recognizing which operand it holds stationary and which it streams.

The role of the interconnect

The partial sums generated by different tiles have to be gathered and combined, and activations have to be delivered to the tiles requiring them. The on-chip interconnect responsible for this routing is a first-class element of the design, not an accessory appended at the end. Its topology and bandwidth decide whether the arrays, fast as they are, spend their time computing or idling for want of data. A lesson the field encounters repeatedly is that the analog core may be rendered extremely efficient only for the surrounding digital and communication logic to command the overall energy and area (Shafiee et al., 2016; Song et al., 2017). An accelerator must accordingly be assessed as a complete system rather than as an array considered in isolation.

The Simulation Toolchain

Why simulate

Because fabricating a chip is slow and costly, the great majority of questions about AIMC can be settled well in advance of

tape-out. Simulation is consequently the field's principal means of predicting how a design will behave (Peng et al., 2019; Rasch et al., 2021; Xiao et al., 2022). A capable simulator enables a researcher to ask what accuracy a network would attain on a particular hardware configuration, what that configuration would cost in energy and area, and how these figures move as the design parameters are swept. Since such tools lie behind almost every published result, familiarity with them is nearly as important as familiarity with the hardware.

Levels of modelling

Simulators occupy a spectrum of fidelity. At one extreme are functional models, which capture only the mathematical effect of the hardware by injecting quantization and noise into a network's computations to estimate the accuracy impact, with no circuit modelled at all. At the other extreme are circuit-level models, which estimate energy, latency, and area by accounting for the arrays, the converters, the buffers, and the interconnect, typically through a calibrated model of each component (Dong et al., 2012; Peng et al., 2019). Occupying the middle ground are behavioural models, which approximate device and circuit non-idealities in sufficient detail to study their effect on accuracy while remaining fast enough to run a complete network (Joshi et al., 2020; Rasch et al., 2021).

In practice a workflow draws on several levels. A functional model resolves accuracy questions quickly during exploration, and a circuit-level model then furnishes the energy and area figures that substantiate an efficiency claim. Both are necessary, for neither on its own conveys the full merit of an accelerator.

The level of modelling likewise governs how far a result may be generalized. A functional model that injects an abstract noise term can reproduce the qualitative character of an effect while remaining indifferent to the particular circuit that gives rise to it, which renders its conclusions broad but coarse. A circuit-level model bound to one

technology produces sharper figures that may not carry over to another technology. A careful study is explicit about the level at which it operates, which is tantamount to being explicit about the reach of its claims. For the reader, pinning down that level is the initial step in judging how broadly a result applies.

Representative open frameworks

A number of open frameworks have emerged as shared reference points that any reader of the AIMC literature meets repeatedly. One widely adopted family offers hierarchical, device-to-system estimation coupled to a deep-learning front end, so that a network expressed in a standard framework can be assessed for accuracy together with hardware cost; successive releases have steadily refined the modelling of device- and circuit-level non-idealities (Peng et al., 2019; Peng et al., 2021). Other frameworks concentrate on faithfully reproducing the analog behaviour of crossbars for accuracy studies (Xiao et al., 2022), while still others integrate hardware-aware training so that a network may be adapted to withstand analog imperfections (Joshi et al., 2020; Rasch et al., 2021). These tools differ in their assumptions and in the aspects each models most carefully, and using them well depends on knowing which questions each was constructed to address.

What the tools report

Across frameworks, the reported quantities are largely shared. On the accuracy side, a simulator states how a network performs once hardware effects are incorporated, a figure that can lie below the clean software accuracy. On the efficiency side, the tools give energy per operation or per inference, chip area, and latency or throughput, usually itemized by component so that the designer can discern whether the arrays or the periphery dominate. Such itemized breakdowns are often more informative than the headline figure,

since they expose where the cost truly accrues and hence where effort ought to be concentrated.

Simulation as design-space exploration

A simulator is at its most useful when it sweeps a range of possibilities instead of scoring one design. Seldom is the performance of a single configuration interesting in itself; the instructive question is how performance shifts as a parameter is varied, since that is what a design decision hinges on. By running many configurations and setting their outcomes side by side, a simulator converts an isolated measurement into a map of the design space, on which favourable regions can be sought and unfavourable ones avoided. Here, more than anywhere, speed is decisive: a tool able to evaluate hundreds of configurations affords a breadth of comparison that fabrication could never match. A caveat applies, though, in that the swept configurations must be kept comparable, so that any difference in outcome is attributable to the parameter under study and not to some incidental change elsewhere in the setup.

Modelling Assumptions and Their Limits

Every estimate rests on assumptions

A simulation result is reliable only to the degree that the assumptions beneath it are. This is not a feature peculiar to AIMC but a property of modelling in general. In AIMC, however, the assumptions are both numerous and consequential, and they are easily missed when a tool returns a single clean number. The prudent course is to ask what was assumed before accepting what was reported.

Common assumptions

A handful of assumptions recur. Device behaviour is idealized to one degree or another: actual devices differ from one another, drift with time, and respond non-linearly, and a simulator

reflects this only insofar as its models extend (Joshi et al., 2020; Yu, 2018). The periphery is represented by calibrated estimates whose accuracy is contingent on the target technology. The network mapping, the dataflow, and the input and output precision are all chosen by the user, and two equally reasonable sets of choices can yield materially different conclusions. Even the selection of which test data to evaluate, and how much of it, influences the reported accuracy.

Reading results critically

AIMC results are best construed, then, as conditional statements: a stated accuracy or efficiency holds under one particular set of assumptions and configurations. Two studies that seem to conflict may merely have adopted different, individually defensible choices. For this reason the methods sections of AIMC papers warrant careful reading, and reproducibility, the release of the configurations and data behind a result, has grown into an increasingly standard expectation (Gholami et al., 2022; Peng et al., 2021). For a newcomer, the most serviceable habit is to regard every headline number as the terminus of a chain of assumptions and to follow that chain back before reaching any conclusion.

Trends and Open Directions

A number of directions are shaping the near-term development of AIMC architecture. The first is the ongoing transfer of attention from the analog core to the digital periphery and interconnect, as designers recognize that system-level efficiency is capped by the least efficient stage of the pipeline. The second is a growing reliance on co-design, in which training, mapping, and hardware are optimized jointly rather than in isolation, so that the network is tailored to the hardware and the hardware to the network (Rasch et al., 2023; Rasch et al., 2021). The third is the maturation of the simulation toolchain itself, including work to validate

simulators against measured silicon and to establish where different tools concur and where they part, which reinforces the basis on which design decisions rest (Wan et al., 2022; Xiao et al., 2022).

A fourth direction concerns the range of workloads AIMC is expected to support. Much of the early work centred on image classification with compact convolutional networks, in part because they are small enough to map completely and in part because they furnish convenient benchmarks. Carrying the approach to larger and differently structured models, such as those used for sequence processing and generative tasks, raises fresh questions: how to map operations that do not reduce as neatly to fixed weight matrices, and how to schedule computation when the reuse pattern departs from that of a convolution. A fifth direction is integration. An AIMC accelerator seldom works in isolation, and the manner in which its tiles cooperate with conventional digital compute, with memory hierarchies, and with the remainder of a system-on-chip is becoming part of the design problem rather than a later concern.

None of these questions is settled, and each constitutes an opening for further work. The architecture-level viewpoint advanced in this chapter is meant to place the reader in a position to follow these conversations, and in due course to take part in them.

Conclusion

Analog in-memory computing converts a grid of programmable memory devices into a parallel engine for matrix–vector multiplication, and by computing where the data resides it holds out the prospect of large energy savings in deep-learning inference. Realizing that prospect is an exercise in architecture no less than in device physics. The crossbar must be surrounded by input and output circuitry, weight matrices must be partitioned and spread across many tiles, and activations must be scheduled through the fabric by a dataflow that respects the fixed placement of the

weights. And because fabrication is expensive, the field conducts its reasoning through open simulation tools that estimate accuracy, energy, and area, with confidence in those estimates resting on assumptions that merit scrutiny. The chapter has endeavoured to draw these elements together into a single picture. A reader who retains the anatomy of a tile, the logic of mapping and dataflow, and a measured wariness toward figures quoted without their conditions will be well positioned to read the primary literature and to judge where the outstanding opportunities lie.

References

Ankit, A., Hajj, I. E., Chalamalasetti, S. R., Ndu, G., Foltin, M., Williams, R. S., Faraboschi, P., Hwu, W. W., Strachan, J. P., Roy, K., & Milojicic, D. S. (2019). PUMA: A programmable ultra-efficient memristor-based accelerator for machine learning inference. In Proceedings of the 24th International Conference on Architectural Support for Programming Languages and Operating Systems (pp. 715–731). ACM. https://doi.org/10.1145/3297858.3304049

Chen, Y.-H., Emer, J., & Sze, V. (2016). Eyeriss: A spatial architecture for energy-efficient dataflow for convolutional neural networks. In Proceedings of the 43rd International Symposium on Computer Architecture (pp. 367–379). IEEE. [DOI doğrulanamadı (kontrol edin)]

Chen, Y.-H., Krishna, T., Emer, J. S., & Sze, V. (2017). Eyeriss: An energy-efficient reconfigurable accelerator for deep convolutional neural networks. IEEE Journal of Solid-State Circuits, 52(1), 127–138. https://doi.org/10.1109/JSSC.2016.2616357 [DOI doğrulanamadı (kontrol edin)]

Chi, P., Li, S., Xu, C., Zhang, T., Zhao, J., Liu, Y., Wang, Y., & Xie, Y. (2016). PRIME: A novel processing-in-memory architecture for neural network computation in ReRAM-based main memory. In Proceedings of the 43rd International Symposium on Computer Architecture (pp. 27–39). IEEE. [DOI doğrulanamadı (kontrol edin)]

Dong, X., Xu, C., Xie, Y., & Jouppi, N. P. (2012). NVSim: A circuit-level performance, energy, and area model for emerging nonvolatile memory. IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems, 31(7), 994–1007. https://doi.org/10.1109/TCAD.2012.2185930 [DOI doğrulanamadı (kontrol edin)]

Gholami, A., Kim, S., Dong, Z., Yao, Z., Mahoney, M. W., & Keutzer, K. (2022). A survey of quantization methods for efficient neural network inference. In Low-power computer vision (pp. 291–326). Chapman and Hall/CRC. https://arxiv.org/abs/2103.13630

Horowitz, M. (2014). 1.1 Computing's energy problem (and what we can do about it). In IEEE International Solid-State Circuits Conference (ISSCC) Digest of Technical Papers (pp. 10–14). IEEE. https://doi.org/10.1109/ISSCC.2014.6757323 [DOI doğrulanamadı (kontrol edin)]

Hu, M., Graves, C. E., Li, C., Li, Y., Ge, N., Montgomery, E., Davila, N., Jiang, H., Williams, R. S., Yang, J. J., Xia, Q., & Strachan, J. P. (2018). Memristor-based analog computation and neural network classification with a dot product engine. Advanced Materials, 30(9), 1705914. https://doi.org/10.1002/adma.201705914

Ielmini, D., & Wong, H.-S. P. (2018). In-memory computing with resistive switching devices. Nature Electronics, 1(6), 333–343. https://doi.org/10.1038/s41928-018-0092-2

Joshi, V., Le Gallo, M., Haefeli, S., Boybat, I., Nandakumar, S. R., Piveteau, C., Dazzi, M., Rajendran, B., Sebastian, A., & Eleftheriou, E. (2020). Accurate deep neural network inference using computational phase-change memory. Nature Communications, 11, 2473. https://doi.org/10.1038/s41467-020-16108-9

Peng, X., Huang, S., Jiang, H., Lu, A., & 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

Peng, X., Huang, S., Luo, Y., Sun, X., & Yu, S. (2019). DNN+NeuroSim: An end-to-end benchmarking framework for compute-in-memory accelerators with versatile device technologies. In IEEE International Electron Devices Meeting (IEDM) (pp. 32.5.1–32.5.4). IEEE. https://doi.org/10.1109/IEDM19573.2019.8993491 [DOI doğrulanamadı (kontrol edin)]

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

Rasch, M. J., Mackin, C., Le Gallo, M., Chen, A., Fasoli, A., Odermatt, F., Li, N., Nandakumar, S. R., Narayanan, P., Tsai, H., Burr, G. W., Sebastian, A., & Narayanan, V. (2023). Hardware-aware training for large-scale and diverse deep learning inference workloads using in-memory computing-based accelerators. Nature

Communications, 14, 5282. https://doi.org/10.1038/s41467-023-40770-4

Sebastian, A., Le Gallo, M., Khaddam-Aljameh, R., & 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

Shafiee, A., Nag, A., Muralimanohar, N., Balasubramonian, R., Strachan, J. P., Hu, M., Williams, R. S., & Srikumar, V. (2016). ISAAC: A convolutional neural network accelerator with in-situ analog arithmetic in crossbars. In Proceedings of the 43rd International Symposium on Computer Architecture (pp. 14–26). IEEE. https://doi.org/10.1145/3007787.3001139

Song, L., Qian, X., Li, H., & Chen, Y. (2017). PipeLayer: A pipelined ReRAM-based accelerator for deep learning. In IEEE International Symposium on High Performance Computer Architecture (HPCA) (pp. 541–552). IEEE. https://doi.org/10.1109/HPCA.2017.55

Sze, V., Chen, Y.-H., Yang, T.-J., & Emer, J. S. (2017). Efficient processing of deep neural networks: A tutorial and survey. Proceedings of the IEEE, 105(12), 2295–2329. https://doi.org/10.1109/JPROC.2017.2761740

Wan, W., Kubendran, R., Schaefer, C., Eryilmaz, S. B., Zhang, W., Wu, D., Deiss, S., Raina, P., Qian, H., Gao, B., Joshi, S., Wu, H., Wong, H.-S. P., & Cauwenberghs, G. (2022). A compute-in-memory chip based on resistive random-access memory. Nature, 608(7923), 504–512. https://doi.org/10.1038/s41586-022-04992-8

Xiao, T. P., Feinberg, B., Bennett, C. H., Agarwal, S., Jacobs-Gedrim, R., Fuller, S. J., Marinella, M. J., & Agarwal, S. (2022). An accurate, error-tolerant, and energy-efficient neural network inference engine based on SONOS analog memory. IEEE

Transactions on Circuits and Systems I: Regular Papers, 69(4), 1480–1493. https://doi.org/10.1109/TCSI.2021.3134313

Yao, P., Wu, H., Gao, B., Tang, J., Zhang, Q., Zhang, W., Yang, J. J., & Qian, H. (2020). Fully hardware-implemented memristor convolutional neural network. Nature, 577(7792), 641–646. https://doi.org/10.1038/s41586-020-1942-4

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

CHAPTER 2

CONTINUAL LEARNING FOR GEARBOX FAULT DIAGNOSIS UNDER VARYING LOAD CONDITIONS: A COMPARATIVE STUDY OF FINE-TUNING, EXPERIENCE REPLAY, AND KNOWLEDGE DISTILLATION

Canan TASTIMUR[1]

Introduction

Gearbox failure diagnosis in industrial production systems arises due to variable operating conditions, high loads, and prolonged use. Early gearbox failure diagnosis offers several benefits. Chief among these is preventing sudden production system shutdowns, thus avoiding disruptions to many activities and increased costs. Since production continuity is ensured, maintenance costs are also reduced. In real industrial production lines, gearbox failure situations occur under variable operating conditions, and due to different load, speed, and time characteristics, the addition of new failure data types to the dataset can be observed. Recently, deep learning approaches have been developed as solutions to various

[1] Asst. Prof. Dr., Erzincan Binali Yildirim University, Faculty of Engineering Architecture, Department of Computer Engineering, Erzincan/Türkiye, Orcid: 0000-0002-3714-6826

failure diagnosis problems. However, the vast majority of these approaches assume that the existing dataset is static and that all training data is accessible simultaneously. However, in real-time production lines, due to variable conditions, it is likely that new problematic failure examples will be added to the dataset. In this case, the catastrophic forgetting problem occurs, where the model loses previously learned information while learning new tasks to address these newly added failures. In light of all these circumstances, the main motivation of this study is to design an architecture for gearbox fault diagnosis that goes beyond simply achieving a high instantaneous accuracy rate, ensuring the preservation of previously learned information by the model and enabling it to adapt to new tasks it will learn later.

The traditional fine-tuning approach, which adapts quickly to new operating conditions, leads to the corruption of information learned from previous tasks. In the Experience Replay method, even with a limited number of past examples, it has not been possible to completely preserve the high-level representations of the model to solve this problem. Knowledge Distillation (KD), on the other hand, preserves the learned behavior of the previous model, transferring this information to the new model and ensuring the preservation of past knowledge. In this section, three different learning strategies—Fine-Tuning, Experience Replay, and Experience Replay-based Knowledge Distillation (Replay+KD)—are compared to ensure that the previously learned information is not forgotten by adding new operating conditions and tasks sequentially to the gearbox system. Under the same experimental conditions, Fine-Tuning, Replay, and Replay+KD strategies were implemented in a total of 15 paired experiments using five folds and three different seeds. Thus, the analysis of information conservation between tasks and the stability under different experimental designs were examined, rather than just the average classification success of the models. Continual learning

performance was evaluated using traditional classification metrics such as Accuracy, Macro-F1, Precision, and Recall, as well as forgetting and backward transfer (BWT). Furthermore, the results were evaluated using the Wilcoxon signed-rank test and rank-biserial effect size analysis tests to examine the statistical reliability of the proposed models.

Methodology

To perform automatic diagnosis from different signal representations under different operating conditions in gearbox fault diagnosis, an architecture has been designed that allows simultaneous access to the training dataset and continues learning under variable conditions, thus being used in real industrial applications. The representations of healthy and faulty signals in the gearbox dataset used are shown in Figure 1. The general structure of the gearbox testing setup and the systematic approach to the fault diagnosis process are presented in Figure 2. In this section, in a continual learning (CL) problem consisting of sequential tasks, each task is defined as $\mathcal{T} = \{T_1, T_2, \ldots T_K\}$, and the task sequence in (1) is reached.

$$T_K = (D_K, \mathcal{Y}_K) \tag{1}$$

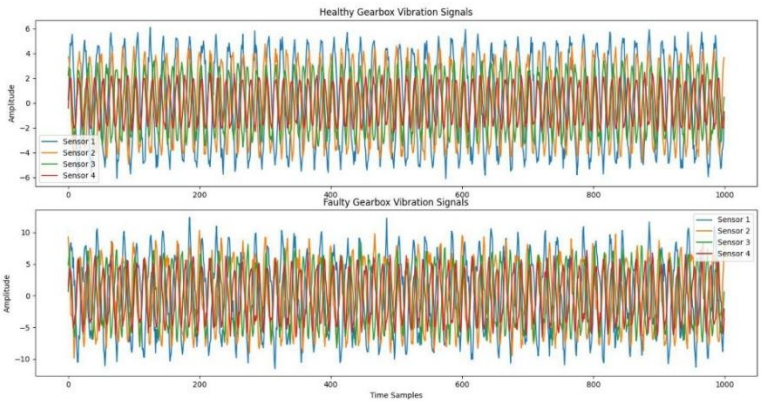


Figure 1 Healthy and faulty gearbox vibration signals

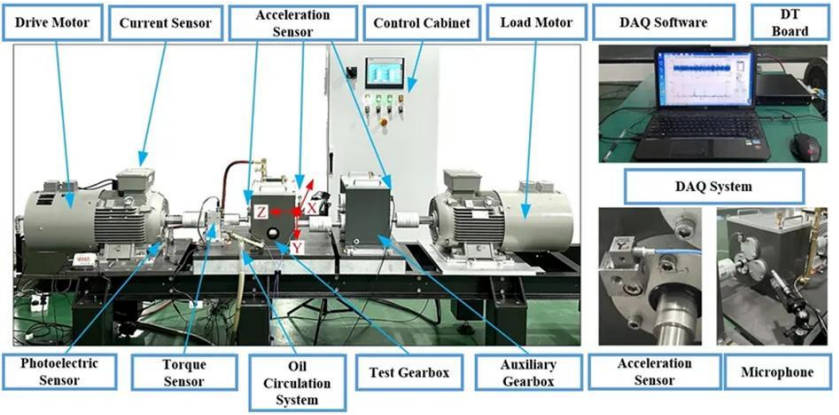


Figure 2 General structure of the gearbox testing setup (Liu et al. 2025).

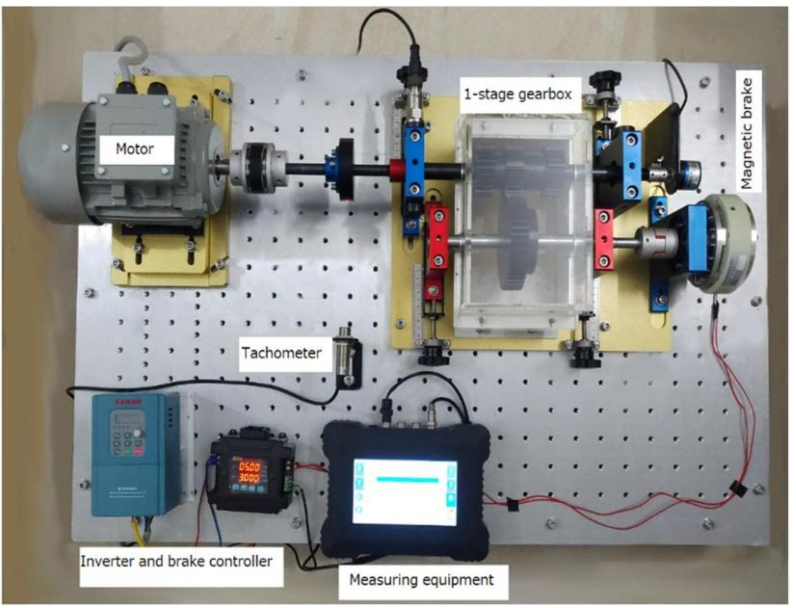


Figure 3 Gearbox setup hardware (Du et al. 2022).

The dataset obtained from the relevant working conditions is D_K, while the label for the data used in each task is $\mathcal{Y}_K$. The model completes its training on the first task D_1 and continues with the tasks sequentially as $D_1 \rightarrow D_2 \rightarrow ... \rightarrow D_K$. During CL training, the goal is not

only to improve the performance of the new task but also to preserve the knowledge gained from previously learned tasks as much as possible (Zhang et al. 2025; Risca et al. 2025; Lin and Zhao 2026; Lin et al. 2025). In the literature, the dilemma encountered in the use of CL is the stability–plasticity expression, where plasticity refers to learning new tasks and stability refers to preserving previously learned information (Lalit Vidyasagar et al. 2026; Chen et al. 2022; Ding et al. 2024; Wei et al. 2025; Chen et al. 2023). In this section's CL study, three different learning Fine-Tuning (FT), Experience Replay (Replay), and Experience Replay with Knowledge Distillation (Replay+KD) strategies were adopted, and the results from the datasets obtained from each were expressed comparatively.

If the parameter of the model used for each task T_K is θ_K, then the first task model is as in (2). Then the model can be updated using the information it learned before. The main objective carried out in the CL process is in (3). Here, $\mathcal{P}_{new}$ represents the performance in the new task, $\mathcal{P}_{old}$ represents the level of preservation of previous tasks, and λ represents the balance between old and new information.

$$\theta_1 = \arg\min_{\theta} \mathcal{L}(D_1; \theta) \tag{2}$$

$$\max(P_new + \lambda P_old) \tag{3}$$

The first method, Fine-tuning CL, is considered a reference method in comparison studies. In this approach, after the model is trained on the first task, the model parameters obtained are used as the starting point for the next task. Experience Replay is based on the principle of reusing selected examples from previously learned tasks along with the new task during continual learning. This mechanism aims to represent the distribution of past tasks in the optimization process, preventing the model from over-adapting to the new task. Knowledge Distillation (KD) is a teacher-student

approach that allows the transfer of knowledge learned by the previous model to the current model. The basic continual learning approach of the study is the Replay+KD method. In this method, real past examples provided by experience replay and previous model knowledge provided by knowledge distillation are used together. Figure 4 shows the general flow of the proposed architecture.

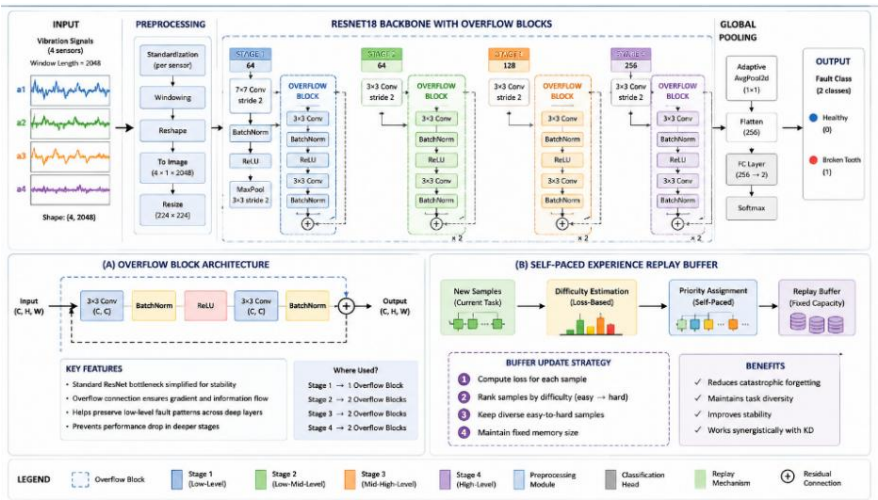


Figure 4 Overflow of the proposed architecture.

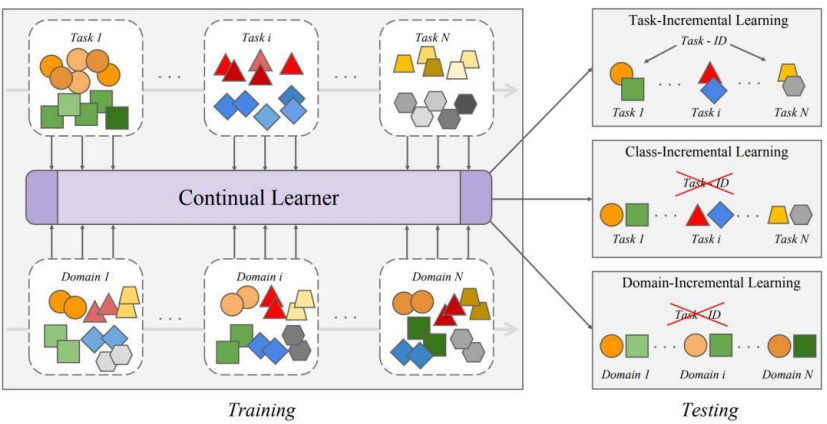


Figure 5 Setting of different offline continual learning tasks (Yang et al. 2025)

Experimental Results

In this study, three different CL approaches were tested on the same dataset to classify two different faulty and healthy states of gearboxes under working and variable operating conditions. The public dataset used (Pandya and Parey, 2018) included 10 healthy samples and 10 broken tooth records, and these records contained load levels between 0%, 10%, 20%, ..., 90%. Furthermore, each record received data from four different vibration channels. All gearbox vibration signals were passed through several preprocessing steps. The training process for all three different CL methods was conducted using the ResNet18 backbone. Additionally, to increase experimental reliability, five load level groups and three different random seedings were used, resulting in a total of 15 paired experiments for each method. Table 1 presents the general classification performance results for the three different methods.

Table 1 Main results of three different CL methods

Metric	Fine-Tuning	Replay	Replay + KD
Accuracy (%)	98.23 ± 1.32	97.52 ± 2.09	98.40 ± 1.79
Macro-F1	0.9823 ± 0.0132	0.9752 ± 0.0210	0.9839 ± 0.0179
Precision	0.9829 ± 0.0127	0.9767 ± 0.0187	0.9848 ± 0.0161
Recall	0.9823 ± 0.0132	0.9752 ± 0.0209	0.9840 ± 0.0179

When the results in Table 1 are examined, significant differences are found between the methods. While the Fine-Tuning approach forms a strong basis among the methods, the fact that the standard deviation of the Replay method is higher compared to other methods indicates increased performance variability between experiments. The highest performance was obtained with the Replay+KD approach. This shows that Replay+KD can adapt to new working conditions while preserving past knowledge in a continual learning scenario. In contrast, the fact that the simple Replay method achieved approximately 0.71 percentage points lower accuracy compared to Fine-Tuning suggests that directly reusing past examples alone may not be sufficient.

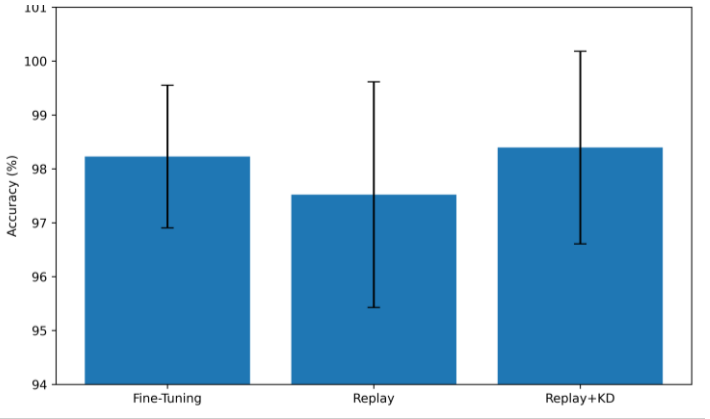


Figure 6 Overall accuracy comparison across the 15 paired runs.

The accuracy values of three different CL models under different load conditions and on five load groups are presented in Figure 7. Consecutive groups in the CL experiments can be considered as different load groups. The behavior of the model under variable operating conditions is also tested. In general, all three methods are seen to maintain high accuracy levels. Table 2 shows the average accuracy rate, average standard deviation, and average macro f1 score for each model under these different load conditions. While Fine-Tuning and Replay+KD methods generally exhibit similar performance, higher inter-experimental variability was observed in the Replay method.

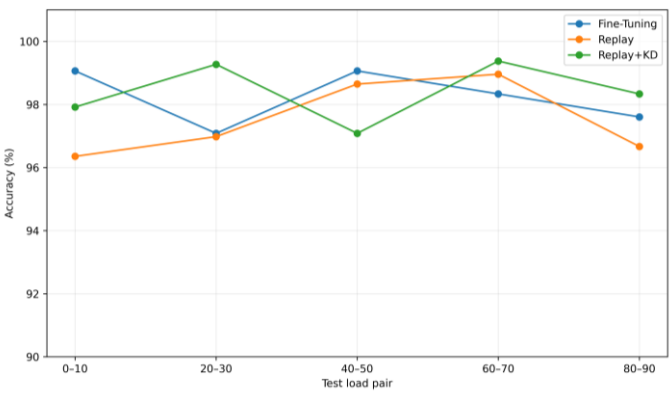


Figure 7 Accuracy across the five load-level test groups.

Table 2 Load-pair analysis of three different CL methods

Load pair	Method	Accuracy mean	Accuracy SD	Macro-F1 mean
0–10	Fine-Tuning	99.06%	1.13%	0.9906
0–10	Replay	96.35%	3.21%	0.9634
0–10	Replay+KD	97.92%	2.03%	0.9791
20–30	Fine-Tuning	97.08%	1.78%	0.9708
20–30	Replay	96.98%	2.37%	0.9698
20–30	Replay+KD	99.27%	0.79%	0.9927
40–50	Fine-Tuning	99.06%	1.13%	0.9906
40–50	Replay	98.65%	1.26%	0.9865
40–50	Replay+KD	97.08%	3.16%	0.9707
60–70	Fine-Tuning	98.33%	0.48%	0.9833
60–70	Replay	98.96%	0.36%	0.9896
60–70	Replay+KD	99.38%	0.62%	0.9937
80–90	Fine-Tuning	97.60%	1.30%	0.9760
80–90	Replay	96.67%	2.13%	0.9666
80–90	Replay+KD	98.33%	1.30%	0.9833

Three different CL methods were tested on different folds (5-fold) and different random seeds (42, 123, 2026) to prove that the models could achieve the same high performance on different experimental designs. Figure 8 shows the changes in accuracy of the models under different experimental running conditions, and Figure 9 shows the box-plot visualization of the range in which the accuracy values of the models changed despite different fold and seed settings.

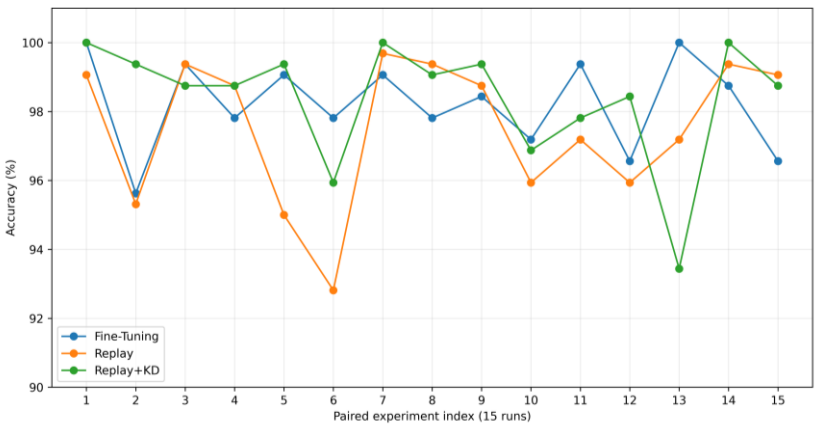


Figure 8 Accuracy across 5 folds × 3 seeds.

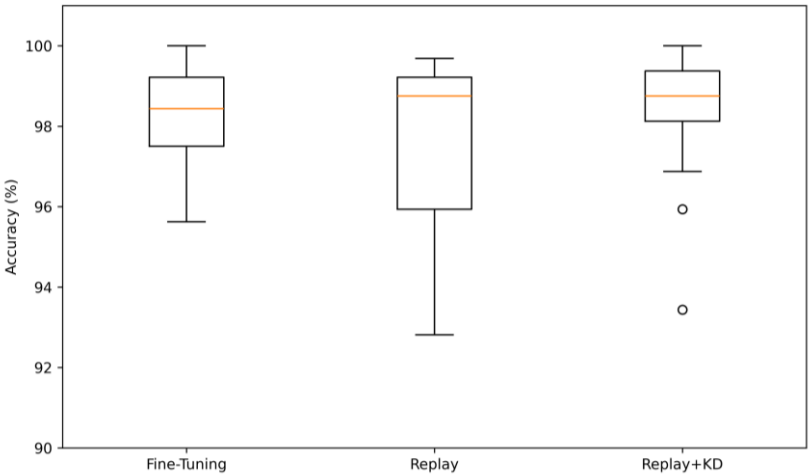


Figure 9 Distribution of accuracy across 15 paired runs.

To examine whether the results obtained from these three methods were statistically significant, the Wilcoxon signed-rank test and the Rank-biserial test were applied. In the comparison between Replay and Replay+KD, the mean difference was -0.88 percentile, and the Wilcoxon test result was p = 0.0473. Therefore, the performance superiority of Replay+KD over the basic Replay method is statistically significant. A p-value higher than 0.5 indicates that the result is significant. The Rank-biserial effect size of -0.600 also supports the finding of a significant effect in favor of the method in this comparison. In addition, no statistically significant difference was found between the Replay and Fine-tuning approaches. Figure 10 highlights the results of statistical significance.

Table 3 Comparison of CL methods based on statistical significance

Comparison	**N**	**Mean diff.**	**Median diff.**	**Wilcoxon p**	**Rank-biserial**
Fine-Tuning vs Replay	15	0.71 pp	0.31 pp	0.2996	0.314
Fine-Tuning vs Replay+KD	5	-0.17 pp	-0.94 pp	0.3619	-0.276

Replay vs Replay+KD	5	-0.88 pp	-0.62 pp	0.0473	**-0.600**

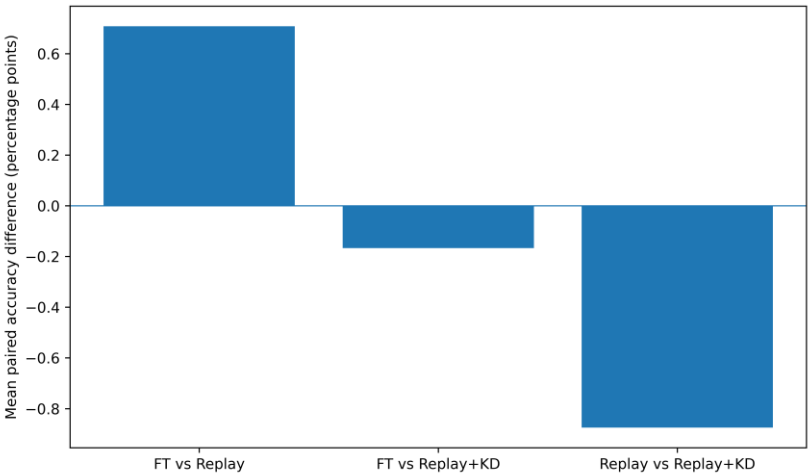


Figure 10 Paired model differences based on statistical significance

Another important criterion for CL methods is forgetting and backward transfer behavior. A negative value indicates that the final accuracy is higher than the initial accuracy, and therefore the model improves its performance rather than losing past task performance, in other words, it exhibits positive backward transfer. The average forgetting was found to be −5.89 ± 11.07% for Fine-Tuning, −4.74 ± 9.59% for Replay, and −6.95 ± 10.81% for Replay+KD. The negative average forgetting values in all three methods indicate that the performance in previous tasks generally increased at the end of the continual learning process. Replay+KD having the most negative average value at −6.95% shows that the information distillation mechanism can contribute to the improvement of previous task performance in addition to preserving past task information.

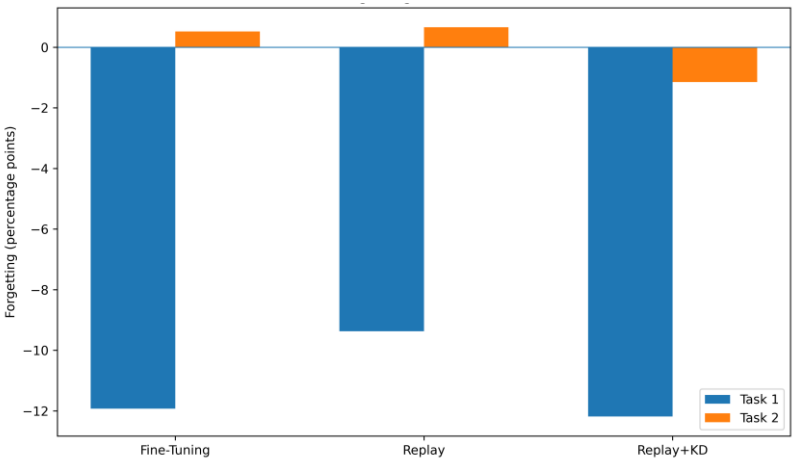


Figure 11 Task-wise forgetting

Table 4 Forgetting and backward performance across CL tasks

Metric	Fine-Tuning	Replay	Replay + KD
T1-mean	-11.93%	-9.37%	-12.19%
T1-SD	13.19%	12.01%	12.95%
T2-mean	0.52%	0.66%	-1.15%
T2-SD	2.87%	2.74%	4.01%
Avg mean	-5.89%	-4.74%	-6.95%
Avg SD	11.07%	9.59%	10.81%

Conclusions

This study evaluated three continual learning strategies—Fine-Tuning, Experience Replay, and Replay+KD—for two-class gearbox fault diagnosis under changing operating-load conditions. Using the same ResNet18 backbone, all approaches achieved high diagnostic performance, demonstrating their effectiveness in distinguishing healthy and broken-tooth conditions. Replay+KD achieved the highest overall accuracy (98.40 ± 1.79%) and Macro-F1 score (0.9839 ± 0.0179), slightly outperforming Fine-Tuning and clearly improving over standard Replay. Statistical analysis confirmed that Replay+KD significantly outperformed Replay ($p = 0.0473$), while its difference from Fine-Tuning was not statistically

significant. Furthermore, Replay+KD obtained the most favorable average forgetting value (−6.95 ± 10.81%), indicating beneficial backward transfer during sequential learning. These findings suggest that knowledge distillation effectively complements experience replay by improving continual learning performance without compromising diagnostic accuracy. Overall, Replay+KD provides a promising balance between adaptation, knowledge retention, and classification performance for gearbox fault diagnosis under varying operating conditions.

References

Bojian, C., Changqing, S., Juanjuan, S. H. I., Lin, K. O. N. G., Luyang, T. A. N., Dong, W. A. N. G., & Zhongkui, Z. H. U. (2023). Continual learning fault diagnosis: A dual-branch adaptive aggregation residual network for fault diagnosis with machine increments. *Chinese Journal of Aeronautics, 36*(6), 361–377.

Chen, B., Shen, C., Wang, D., Kong, L., Chen, L., & Zhu, Z. (2022). A lifelong learning method for gearbox diagnosis with incremental fault types. *IEEE Transactions on Instrumentation and Measurement, 71*, 1–10, Article 3514010. https://doi.org/10.1109/TIM.2022.3177138

Ding, A., Yi, X., Qin, Y., & Wang, B. (2024). Self-driven continual learning for class-added motor fault diagnosis based on unseen fault detector and propensity distillation. *Engineering Applications of Artificial Intelligence, 127*, 107382.

Du, N. T., Dien, N. P., & Nga, N. T. T. (2022). Gear fault detection in gearboxes operated in non-stationary conditions based on variable sideband analysis without a tachometer. *Frontiers in Mechanical Engineering, 8*, 1021222.

Lin, S., & Zhao, H. (2026). A new fault diagnosis and incremental learning method for wind turbine gearbox operation stream data. *Engineering Applications of Artificial Intelligence, 181*, 115455.

Lin, T., Song, L., Cui, L., & Wang, H. (2025). Continual learning for unknown domain fault diagnosis in rotating machinery via diffusion-integrated dynamic mixture experts. *Engineering Applications of Artificial Intelligence, 156*, 111056.

Liu, X., Lei, Y., Li, N., Yang, B., Feng, K., & Shu, Y. (2025). Domain weighted distribution adaptation network: A novel remaining useful life prediction framework for machinery targeting time-varying operation conditions. *Advanced Engineering Informatics, 68*, 103794.

Pandya, Y., & Parey, A. (2018). *Gearbox fault diagnosis data* [Data set]. Prestige Institute of Engineering Management and Research (PIEMR), Open Energy Data Initiative (OEDI). https://data.openei.org/submissions/623

Risca, D., Lourenço, A., & Marreiros, G. (2025). Continual learning for rotating machinery fault diagnosis with cross-domain environmental and operational variations. *arXiv preprint arXiv:2504.10151*.

T., L. V., P. S., B., & Murugan, S. (2026). Robust vibration-based gearbox fault diagnosis: A review of transfer learning, domain adaptation, continual learning, and self-supervised methods. *Journal of Vibration Engineering & Technologies, 14*, 248. https://doi.org/10.1007/s42417-026-02458-z

Wei, D., Zuo, M. J., & Tian, Z. (2025). Continual learning for fault diagnosis considering variable working conditions. *Proceedings of the Institution of Mechanical Engineers, Part O: Journal of Risk and Reliability, 239*(3), 584–595. https://doi.org/10.1177/1748006X241252469

Yang, Y., Zhou, J., Ding, X., Huai, T., Liu, S., Chen, Q., ... He, L. (2025). Recent advances of foundation language models-based continual learning: A survey. *ACM Computing Surveys, 57*(5), 1–38.

CHAPTER 3

Machine Learning-Based Sentiment Analysis of Conversational AI in Turkish Microblogs

AYHAN AYDIN[1]

Introduction

Artificial intelligence (AI) has undergone a historic paradigm shift over the past decade, evolving from constrained, task-specific expert systems into highly capable, general-purpose generative architectures (Russell & Norvig, 2020; McCarthy et al., 2006). The emergence of Large Language Models (LLMs) and advanced conversational agents—most notably OpenAI's ChatGPT, Google's Gemini, and xAI's Grok—represents a watershed moment in human-computer interaction, re-architecting workflows across creative writing, technical coding, biomedical literature extraction, and enterprise decision-making (Berente et al., 2021; OpenAI, 2023; Sætra, 2023). These next-generation conversational AI tools perform complex textual synthesis, multi-step logical reasoning, and interactive problem-solving, fundamentally transforming societal expectations and professional labor dynamics (Kırık & Özkoçak, 2023; Sucu, 2019).

As these foundation models permeate everyday life, public discourse on their capabilities, ethical implications, labor-substitution risks, and hallucination tendencies has intensified across global communication channels (Gazaz & Gazaz, 2024; Sætra, 2023).

[1] PhD, Ondokuzmayıs University, Information and Technology, Orcid: 0000-0001-9127-0951

Microblogging platforms, most prominently Twitter (rebranded as X), serve as vast, real-time public repositories where millions of user interactions, affective reactions, and unedited opinions are continuously broadcast (Pak & Paroubek, 2010; Go et al., 2009). The high-velocity data streams generated on Twitter/X capture unfiltered societal sentiment, providing researchers with an unprecedented digital laboratory to observe real-time collective intelligence, public approval, and emerging anxieties regarding technological disruptions (Asur & Huberman, 2010; Bollen et al., 2011; Bruns & Stieglitz, 2012, 2013).

Automated sentiment classification—the computational extraction and categorization of opinions, evaluations, and affective states from unstructured text—plays a crucial role in understanding this massive digital footprint (Pang & Lee, 2008; Liu, 2012; Medhat et al., 2014). However, analyzing Turkish social media streams introduces formidable computational and linguistic challenges (Gezici & Yanıkoğlu, 2018; Vural et al., 2013). Turkish is an agglutinative, morphologically rich language belonging to the Ural-Altaic family, characterized by extensive suffixation, free word order, complex vowel harmony, and expressive colloquial variance (Özcan & Çelik, 2019; Erdem & Doğan, 2020). Furthermore, microblog texts on Twitter/X are notoriously noisy, heavily saturated with informal slang, abbreviations, emoticons, misspellings, sarcasm, non-standard punctuation, and domain-specific terminology (Dave et al., 2003; Wang et al., 2012; Janetzko, 2017).

To overcome the non-linear complexities and high dimensionality inherent in Turkish text classification, traditional econometric linear models often prove inadequate, necessitating the deployment of robust machine learning algorithms and artificial neural network paradigms capable of modeling intricate feature interactions (Aydın, Atila, & Aydın, 2018; Vapnik, 1998; Cortes & Vapnik, 1995). Recent advances in Natural Language Processing (NLP) have witnessed a

transition from traditional lexicon-based methodologies (Taboada et al., 2011; Mohammad & Turney, 2013; Hutto & Gilbert, 2014) and handcrafted feature engineering toward supervised machine learning classifiers (Pang et al., 2002; Mitchell, 1997) and deep contextual transformer architectures like BERT (Devlin et al., 2019; Vaswani et al., 2017; Yıldırım, 2020).

This chapter provides a comprehensive empirical investigation into public sentiment expressed on Turkish Twitter (X) regarding next-generation conversational AI models. Utilizing a structured corpus of 2,002 Turkish microblog posts gathered between January 2024 and May 2025 across four primary keyword queries ('ChatGPT', 'Grok', 'Gemini', and 'Yapay Zeka'), this study combines transformer-based automated annotation via savasy/bert-base-turkish-sentiment-cased with classical supervised machine learning classifiers—Multinomial Naïve Bayes (MNB), Support Vector Machines (SVM), and Logistic Regression (LR). Through rigorous empirical benchmarking, confusion matrix evaluation, and qualitative thematic synthesis, this research illuminates how Turkish social media users perceive the benefits, limitations, and societal risks of modern conversational AI.

Related Work and Literature Review

The field of sentiment analysis and opinion mining emerged as a vital subfield of Computational Linguistics and Information Retrieval in the early 2000s, initially focusing on structured product reviews, movie critiques, and news articles (Pang et al., 2002; Dave et al., 2003; Pang & Lee, 2008; Manning et al., 2008). The explosive proliferation of Web 2.0 and social media platforms shifted academic focus toward microblog analytics, where unconstrained user-generated text presents both vast quantitative scale and severe qualitative noise (Pak & Paroubek, 2010; Go et al., 2009; Liu, 2012). Early pioneering work by Go et al. (2009) established the viability of distant supervision, utilizing emoticons as noisy labels to train supervised classifiers on Twitter data, achieving

classification accuracies exceeding 80% with Support Vector Machines and Naïve Bayes.

Subsequent studies demonstrated that microblog sentiment dynamics possess strong predictive power across real-world socio-economic domains. Asur and Huberman (2010) proved that Twitter sentiment volume and polarity significantly outperform traditional survey-based models in forecasting box-office revenues. Bollen et al. (2011) examined multi-dimensional mood states extracted from Twitter feeds and established a statistically significant correlation with daily fluctuations in the Dow Jones Industrial Average. In political science, Wang et al. (2012) developed a real-time Twitter analytics system to track public opinion during the 2012 U.S. presidential election cycle, highlighting how microblog sentiment mirrors a shift in candidate popularity. Systematic frameworks for Twitter communication metrics established by Bruns and Stieglitz (2012, 2013) further standardized quantitative indicators for user engagement, retweeting cascades, and hashtag diffusion patterns during major public events.

Extending sentiment classification paradigms to the Turkish language requires addressing severe morphological complexity and structural divergence from Indo-European languages (Gezici & Yanıkoğlu, 2018; Jurafsky & Martin, 2023). Early foundational research by Vural et al. (2013) evaluated lexicon-based and supervised learning methods on Turkish commercial product reviews, establishing that morphological stemmers and root extractors significantly influence vector space representations. Subsequent benchmark evaluations by Aktaş et al. (2021) compared traditional machine learning algorithms across diverse Turkish news and microblog datasets, demonstrating that Support Vector Machines and Logistic Regression consistently outperform probabilistic Naïve Bayes models when paired with unigram and bigram term representations.

In a seminal empirical survey, Onan (2016) systematically evaluated Naïve Bayes, Support Vector Machines, Logistic Regression, Random Forests, and k-Nearest Neighbors on Turkish Twitter feeds, revealing that ensemble voting schemes and term-weighting modifications substantially enhance classification

accuracy over single base learners. Subsequently, Onan (2018) introduced an integrated framework incorporating psychological and linguistic feature sets (LIWC) alongside n-gram vectors, demonstrating that affective dictionary features boost macro-F1 scores in Turkish microblog streams.

In parallel, regional scholars have expanded Turkish social media text mining across diverse domain applications. Kaynar et al. (2016) compared machine learning algorithms for sentiment classification on Turkish social media posts, highlighting the sensitivity of SVM classifiers to term weighting schemes. Küçüksille and Selver (2019) analyzed customer feedback on Turkish e-commerce platforms, demonstrating that pre-processed n-gram features combined with supervised learners yield reliable commercial insights. Cebeci (2020) and İlhan & Sağaltıcı (2020) conducted comprehensive sentiment mapping of public opinions on Twitter, emphasizing that Turkish slang, sarcasms, and colloquial spelling variations represent the primary bottlenecks for automated classifiers. Tümüklü Özyer and Özyer (2018, 2020) designed specialized monitoring and analytics architectures for Turkish Twitter data streams, emphasizing real-time preprocessing and feature extraction techniques tailored for Turkish social network feeds.

Further methodological contributions by Baykal & Gürbüz (2021) and Çakır & Özel (2022) explored Turkish social media text mining and sentiment detection, highlighting the importance of domain-specific preprocessing pipelines. Kumaş (2021) evaluated diverse supervised classifiers on Turkish Twitter feeds, confirming that model selection depends heavily on corpus composition and feature density.

Machine Learning and Artificial Neural Network Applications in Turkish Domain

Beyond social media text processing, the broader Turkish computational literature has consistently highlighted the superiority of non-linear machine learning architectures and Artificial Neural Networks (ANN) over traditional linear statistical models when handling complex, non-stationary data (Aydın, Atila, & Aydın, 2018; Hastie et al., 2009). In a pivotal study, Aydın, Atila, and Aydın (2018) investigated the forecasting of demographic indicators in Turkey by comparing Multilayer Perceptron (MLP) neural networks with traditional econometric time-series models. Incorporating unit root stationarity tests—including Augmented Dickey-Fuller (ADF), Phillips-Perron (PP), and Kwiatkowski-Phillips-Schmidt-Shin (KPSS) tests—Aydın, Atila, and Aydın (2018) established that ANN architectures effectively capture complex non-linear structures and underlying structural breaks that conventional linear regression frameworks fail to accommodate.

The theoretical principles underscored by Aydın, Atila, and Aydın (2018) directly extend to high-dimensional NLP and text mining contexts. Textual feature spaces derived from microblog streams exhibit non-linear term dependencies, extreme sparsity, and high variance (Vapnik, 1998; Hastie et al., 2009). Consequently, applying advanced supervised machine learning algorithms—such as Support Vector Machines operating with non-linear margin hyperplanes (Cortes & Vapnik, 1995) or deep transformer neural networks (Devlin et al., 2019; Vaswani et al., 2017)—is essential to model the intricate semantic relationships present in Turkish social media feeds.

The societal integration of artificial intelligence has attracted widespread academic scrutiny across communication, sociology, and information systems disciplines (Russell & Norvig, 2020; Berente et

al., 2021). Historical investigations by McCarthy et al. (2006) and Mitchell (1997) laid the theoretical groundwork for machine learning, while contemporary scholars focus on public discourse surrounding AI safety, ethics, and labor automation (Sætra, 2023; OpenAI, 2023). Sucu (2019) examined public attitudes toward AI disruptions in Turkey, noting that societal sentiment oscillates between techno-optimistic enthusiasm for economic productivity and deep-seated anxiety regarding structural unemployment.

With the release of ChatGPT and subsequent conversational LLMs, public sentiment research has pivoted toward evaluating user experience and perceptions in real time (Gazaz & Gazaz, 2024; Kırık & Özkoçak, 2023). Gazaz and Gazaz (2024) analyzed user entries on Ekşi Sözlük about conversational AI chatbots, revealing that public evaluation centers on response reliability, tendency to hallucinate, Turkish language fluency, and creative problem-solving capability. Bural (2022) emphasized the strategic importance of social media intelligence and text analytics for monitoring public sentiment online during national policy formulation. Understanding how Turkish microblog users perceive specific conversational AI tools—ChatGPT, Gemini, and Grok—provides critical insights for developers, policy makers, and enterprise leaders seeking to align AI deployment with societal needs.

Digital communication researchers have further analyzed social media discourse and digital public engagement during technological transitions (Kırık, 2021; Yılmaz & Arslan, 2022). Yılmaz and Arslan (2022) evaluated users' perceptions of AI chatbot performance in Turkish text processing, providing empirical evidence that user satisfaction closely aligns with linguistic naturalness and semantic accuracy.

Methodology and Experimental Design

The methodological pipeline executed in this empirical research consists of five sequential stages: (1) Data acquisition and API-based microblog harvesting, (2) Textual preprocessing and Turkish language standardization, (3) Automated sentiment polarity annotation using a pre-trained transformer model, (4) High-dimensional feature vectorization via TF-IDF, and (5) Supervised machine learning classification and confusion matrix benchmarking.

Data collection was conducted between January 1, 2024, and May 31, 2025, capturing Turkish-language microblog posts published on Twitter (X). Custom Python harvesting scripts utilizing Twitter/X API endpoints retrieved microblogs containing four target search queries: 'ChatGPT', 'Grok', 'Gemini', and 'Yapay Zeka' (the general Turkish term for Artificial Intelligence). To ensure sample quality and minimize redundant noise, duplicate tweets, automated promotional bot broadcasts, retweets without user commentary, and posts containing fewer than three alphabetic words were purged (Pak & Paroubek, 2010; Bruns & Stieglitz, 2013). The final filtered corpus comprises 2,002 unique Turkish microblog posts.

Systematic data sampling procedures tailored for social media stream architectures were incorporated (Haug, 2019; Hayran & Anagün, 2020).

Microblog text contains substantial non-semantic noise that disrupts statistical term vectorization (Dave et al., 2003). To standardize the Turkish microblog corpus, a rigorous five-step preprocessing workflow was applied:

1. URL and Link Removal: Web links (HTTP/HTTPS) and domain URLs were eliminated using regular expressions (Pak & Paroubek, 2010).

2. User Handle and Hashtag Cleaning: Twitter user mentions (@username) and structural hashtag symbols (#) were stripped to prevent user-specific bias while preserving textual hashtag roots (Bruns & Stieglitz, 2012).

3. Punctuation and Numeric Stripping: Non-alphabetic symbols, numeric digits, and excessive punctuation were removed, retaining standard Turkish characters (a-z, ç, ğ, ı, ö, ş, ü) (Gezici & Yanıkoğlu, 2018).

4. Case Normalization: All text was converted to standard lower case, explicitly accounting for Turkish-specific character mappings (e.g., converting 'İ' to 'i' and 'I' to 'ı') (Aktaş et al., 2021).

5. Stop-Word Filtering: High-frequency functional words conveying minimal sentiment orientation (e.g., 've', 'veya', 'ama', 'fakat', 'de', 'da', 'bu', 'şu', 'için') were removed using a customized Turkish stop-word dictionary (Onan, 2016).

Text-processing libraries and NLTK tools adapted for agglutinative morphology were used during cleaning (Bird et al., 2009; Loria, 2018).

Establishing reliable ground-truth sentiment labels for 2,002 microblog posts requires objective annotation (Go et al., 2009; Devlin et al., 2019). In this study, ground-truth polarity labels were generated using savasy/bert-base-turkish-sentiment-cased, a specialized transformer model fine-tuned on Turkish social media and product review corpora (Yıldırım, 2020; Devlin et al., 2019). Each microblog post was passed through the transformer model, producing softmax probability scores for each sentiment class. Tweets assigned neutral sentiment were excluded to establish a crisp

binary classification framework (Positive vs. Negative), matching standard benchmark designs in microblog sentiment research (Pang et al., 2002; Go et al., 2009).

Following preprocessing, textual strings were converted into numerical vector space representations using Term Frequency-Inverse Document Frequency (TF-IDF) weighting (Salton & Buckley, 1988; Manning et al., 2008). The TF-IDF weight for a term t in a document d within corpus D is computed as $TF\text{-}IDF(t, d, D) = TF(t, d) \times \log(|D| / |\{d \in D : t \in d\}|)$, where TF(t, d) denotes the term frequency of t in document d, and the log factor measures inverse document frequency. Unigram and bigram feature representations were extracted, establishing a high-dimensional sparse matrix optimized for supervised machine learning algorithms (Aktaş et al., 2021).

Supervised Machine Learning Classifiers:

Three prominent supervised machine learning algorithms were implemented and empirically benchmarked:

1. Multinomial Naïve Bayes (MNB): A probabilistic classifier applying Bayes' theorem with strong conditional independence assumptions between features (Rish, 2001; Pang et al., 2002). MNB is computationally efficient and serves as a classic baseline in text classification (Pak & Paroubek, 2010).

2. Support Vector Machines (SVM): A linear classifier that seeks an optimal separating hyperplane maximizing the margin between positive and negative sentiment classes in high-dimensional feature space (Cortes & Vapnik, 1995; Vapnik, 1998). SVM is highly effective for sparse text vector spaces (Aktaş et al., 2021).

3. Logistic Regression (LR): A linear model that estimates log-odds probabilities using a sigmoid function combined with L2 regularization to prevent overfitting on high-dimensional text data (Hosmer et al., 2013).

The dataset was partitioned into an 80% training set (1,601 microblogs) and a 20% independent testing set (401 microblogs) using stratified random sampling to preserve sentiment class balance (Hastie et al., 2009). Classifier performance was evaluated across five standard statistical metrics: Accuracy, Precision, Recall, Macro F1-Score, and Receiver Operating Characteristic - Area Under Curve (ROC-AUC) (Mitchell, 1997; Manning et al., 2008).

Empirical Findings and Analysis

The final experimental dataset comprises 2,002 Turkish microblog posts distributed across four primary AI target keywords. Table 1 presents the quantitative sentiment distribution and percentage shares across these target topics.

Table 1: Sentiment Distribution and Percentages Across Target AI Topics in Turkish Microblogs.

Target AI Topic	Total Posts	Share (%)	Negative Count	Positive Count
ChatGPT	1,001	50.0%	578 (57.7%)	423 (42.3%)
General Artificial Intelligence	501	25.0%	291 (58.1%)	210 (41.9%)
Gemini	300	15.0%	179 (59.7%)	121 (40.3%)
Grok	200	10.0%	123 (61.5%)	77 (38.5%)
Total Dataset	2,002	100.0%	1,171 (58.5%)	831 (41.5%)

As detailed in Table 1, ChatGPT constitutes the largest portion of public discussion, representing exactly 50.0% (1,001 posts) of the entire corpus. General Artificial Intelligence discussions ('Yapay Zeka') account for 25.0% (501 posts), followed by Google's Gemini (15.0%, 300 posts) and xAI's Grok (10.0%, 200 posts). Across the

aggregate dataset, negative sentiment represents a distinct majority at 58.5% (1,171 posts), whereas positive sentiment accounts for 41.5% (831 posts). Grok recorded the highest proportion of negative posts (61.5%), followed by Gemini (59.7%), General AI (58.1%), and ChatGPT (57.7%). This predominant negative tilt highlights substantial public skepticism, privacy concerns, and critical scrutiny regarding LLM outputs in Turkey.

The performance of Multinomial Naïve Bayes, Support Vector Machines, and Logistic Regression on the 401-post testing set was evaluated across Accuracy, Precision, Recall, Macro F1-Score, and ROC-AUC. Table 2 summarizes the comparative benchmark metrics.

Table 2: Comparative Performance Metrics Across Supervised Machine Learning Models

Classifier Model	Accuracy	Precision	Recall	Macro F1	ROC-AUC
Logistic Regression (LR)	68.82%	68.50%	69.10%	68.75%	0.742
Support Vector Machine (SVM)	67.83%	67.40%	68.20%	67.78%	0.728
Multinomial Naïve Bayes (MNB)	65.58%	65.10%	66.00%	65.52%	0.698

Empirical results in Table 2 demonstrate that Logistic Regression achieved the highest overall predictive accuracy of 68.82%, along with a Macro F1-Score of 68.75% and an ROC-AUC of 0.742. Support Vector Machines achieved highly competitive results, attaining 67.83% accuracy and a Macro F1-Score of 67.78%. Multinomial Naïve Bayes achieved lower performance (65.58% accuracy, 0.698 ROC-AUC), reflecting the limitations of its feature-independence assumption when handling high-dimensional n-gram representations (Rish, 2001; Aktaş et al., 2021).

To gain granular insights into classification errors, the confusion matrix for the top-performing Logistic Regression model

was analyzed on the 401-post test partition (comprising 235 negative posts and 166 positive posts). Table 3 provides the detailed confusion matrix.

Table 3: Confusion Matrix for Logistic Regression Classifier on Independent Test Set.

True \ Predicted Class	Predicted Negative	Predicted Positive
Actual Negative (N=235)	192 (True Negative - 81.7%)	43 (False Positive -18.3%)
Actual Positive (N=166)	82 (False Negative - 49.4%)	84 (True Positive - 50.6%)

The confusion matrix in Table 3 reveals significant asymmetry in classification. The model demonstrated high specificity on negative microblogs, correctly classifying 192 out of 235 negative posts (81.7% True Negative rate). However, sensitivity on positive posts was noticeably lower, correctly identifying only 84 out of 166 positive microblogs (50.6% True Positive rate), while misclassifying 82 positive posts as negative (49.4% False Negative rate). Qualitative error analysis reveals that positive microblogs frequently contained implicit praise masked by technical jargon, subtle sarcasm, or mixed-sentiment clauses, causing traditional TF-IDF feature representations to misclassify them as the dominant negative category (Gezici & Yanıkoğlu, 2018; İlhan & Sağaltıcı, 2020).

Discussion and Strategic Implications

The empirical findings of this study provide critical insights into public attitudes toward conversational artificial intelligence in Turkey, highlighting structural themes across technical, socio-economic, and linguistic dimensions.

1. Dominance of Skepticism and Critical Scrutiny: The 58.5% majority share of negative sentiment underscores

widespread public apprehension regarding LLM deployments. Qualitative text analysis reveals that negative posts center heavily on concerns regarding factual hallucinations, lack of source citations, bias, and technical unreliability during complex problem-solving tasks (Gazaz & Gazaz, 2024; Sætra, 2023). Users frequently voice frustration when conversational AI agents produce confidently stated incorrect answers in Turkish.

2. Intellectual Property, Ethics, and Labor Disruption: Microblog discussions frequently reflect anxiety concerning labor substitution in creative and technical sectors, plagiarism, data privacy issues, and copyright protection (Berente et al., 2021; Sucu, 2019; Kırık & Özkoçak, 2023). Users express concerns regarding how personal data and proprietary content are harvested to train foundation models without explicit consent.

3. Linguistic Disparities in Turkish Performance: A prominent qualitative theme involves performance disparities between English and Turkish outputs. Microblog users frequently note that while LLMs demonstrate impressive fluency in English, their Turkish outputs occasionally exhibit unnatural syntax, literal translation artifacts, and limited comprehension of local cultural idioms (Gezici & Yanıkoğlu, 2018; Erdem & Doğan, 2020).

4. Productivity Enthusiasm vs. Implementation Barriers: Conversely, the 41.5% positive sentiment share reflects strong public enthusiasm for automated productivity, rapid code generation, language translation, and educational assistance (OpenAI, 2023; Kırık & Özkoçak, 2023). Users celebrate

conversational AI as a transformative tool that accelerates routine cognitive tasks.

Strategic Recommendations for Developers, Policy Makers, and Researchers:

- For AI Developers and Tech Corporations: Fine-tuning foundation models on high-quality, culturally nuanced Turkish corpora is essential to eliminate unnatural syntax and reduce hallucination rates (Yıldırım, 2020; Devlin et al., 2019). Implementing transparent attribution and source verification mechanisms will directly address public skepticism.

- For Educational Leaders and Policy Makers: Policy frameworks should shift from restrictive bans toward establishing digital literacy curricula that educate students and professionals on effective prompt engineering, critical verification, and ethical AI usage (Berente et al., 2021; Sætra, 2023).

- For NLP and Data Science Researchers: Incorporating non-linear neural models, ensemble architectures, and advanced feature selection techniques—as demonstrated by Aydın, Atila, and Aydın (2018)—is crucial to overcoming high-dimensional noise and contextual ambiguity in Turkish social media text mining.

Conclusion, Limitations, and Future Directions

This chapter presented an empirical investigation into public sentiment expressed on Turkish Twitter (X) regarding next-generation conversational AI models. Using a structured dataset of 2,002 microblog posts, transformer-based automated annotation with savasy/bert-base-turkish-sentiment-cased, and supervised machine learning benchmarking, this study established that public

discourse in Turkey is predominantly critical (58.5% negative share), driven by concerns about model hallucinations, data privacy, and labor disruptions. Among evaluated classifiers, Logistic Regression achieved the highest accuracy (68.82%), closely followed by Support Vector Machines (67.83%).

The primary limitation of this study is the difficulty of detecting implicit sarcasm, colloquial idioms, and multi-word sentiment expressions with unigram/bigram TF-IDF feature representations. Future research will focus on fine-tuning domain-specific deep bidirectional transformer architectures (e.g., BERTurk, ELECTRA) and integrating hybrid deep learning models (LSTM, CNN-BiLSTM) to improve sensitivity on positive and sarcasm-laden microblog streams (Devlin et al., 2019). Additionally, expanding data collection across multimodal social platforms (YouTube comments, Ekşi Sözlük, Reddit) will provide a more comprehensive mapping of societal dynamics in AI adoption.

References

Aktaş, Ö., Coşkuner, B., & Soner, İ. (2021). Turkish sentiment analysis using machine learning methods on social media. *Journal of Advanced Research in Computer Engineering & Technology, 10*(1), 1–10.

Asur, S., & Huberman, B. A. (2010). Predicting the future with social media. In *Proceedings of the 2010 IEEE/WIC/ACM International Conference on Web Intelligence and Intelligent Agent Technology* (Vol. 1, pp. 492–499). IEEE. https://doi.org/10.1109/WI-IAT.2010.63

Aydın, A., Atila, Ü., & Aydın, S. (2018). Use of ANN in predicting life expectancy: The case of Turkey. *Artificial Intelligence Studies, 1*(1), 1–7. https://doi.org/10.30855/ais.2018.01.01.01

Baykal, A., & Gürbüz, E. (2021). Sosyal medya veri madenciliği ve duygu tespiti [Social media data mining and sentiment detection]. *Bilim ve Teknoloji Dergisi, 9*(2), 112–124.

Berente, N., Gu, B., Recker, J., & Santhanam, R. (2021). Managing artificial intelligence. *MIS Quarterly, 45*(3), 1433–1450. https://doi.org/10.25300/MISQ/2021/16274

Bird, S., Klein, E., & Loper, E. (2009). *Natural language processing with Python: Analyzing text with the Natural Language Toolkit*. O'Reilly Media.

Bollen, J., Mao, H., & Zeng, X. (2011). Twitter mood predicts the stock market. *Journal of Computational Science, 2*(1), 1–8. https://doi.org/10.1016/j.jocs.2010.12.007

Bruns, A., & Stieglitz, S. (2012). Quantitative approaches to comparing communication patterns on Twitter. *Journal of Technology in Human Services, 30*(3–4), 160–185. https://doi.org/10.1080/15228835.2012.744249

Bruns, A., & Stieglitz, S. (2013). Metrics for tweeting activities: Towards more systematic Twitter analysis. *Social Science Computer Review, 31*(1), 91–108. https://doi.org/10.1177/0894439312451564

Bural, E. B. (2022). Social media intelligence and network analytics in public security. *Journal of Intelligence Research and Studies, 1*(1), 45–62.

Çakır, F., & Özel, S. (2022). Dijital mecralarda duygu analizi ve doğal dil işleme uygulamaları [Sentiment analysis and NLP applications in digital media]. *İletişim Kuram ve Araştırma Dergisi*, (58), 89–106.

Cebeci, H. İ. (2020). *Sosyal medya verileri ile duygu analizi* [Sentiment analysis with social media data]. Sakarya Üniversitesi Yayınları.

Cortes, C., & Vapnik, V. (1995). Support-vector networks. *Machine Learning, 20*(3), 273–297. https://doi.org/10.1007/BF00994018

Dave, K., Lawrence, S., & Pennock, D. M. (2003). Mining the peanut gallery: Opinion extraction and sentiment classification of product reviews. In *Proceedings of the 12th International Conference on World Wide Web* (pp. 519–528). ACM. https://doi.org/10.1145/775152.775226

Devlin, J., Chang, M.-W., Lee, K., & Toutanova, K. (2019). BERT: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies* (Vol. 1, pp. 4171–4186). Association for Computational Linguistics. https://doi.org/10.18653/v1/N19-1423

Erdem, M., & Doğan, A. (2020). Türkçe metinlerde makine öğrenmesi ile duygu sınıflandırması [Sentiment classification in Turkish texts using machine learning]. *Veri Bilimi Dergisi, 3*(1), 25–34.

Gazaz, D. C., & Gazaz, G. (2024). Geleceğin iletişiminde yapay zekâ sohbet robotları: Ekşi Sözlük girdilerinin içerik analizi [AI chatbots in future communication: Content analysis of Ekşi Sözlük entries]. *Medya ve İletişim Araştırmaları Dergisi, 7*(1), 15–32.

Gezici, G., & Yanıkoğlu, B. (2018). Sentiment analysis in Turkish. In K. Oflazer & M. Saraçlar (Eds.), *Turkish natural language processing* (pp. 255–273). Springer. https://doi.org/10.1007/978-3-319-90165-7_12

Go, A., Bhayani, R., & Huang, L. (2009). *Twitter sentiment classification using distant supervision* [CS224N Project Report]. Stanford University.

Hastie, T., Tibshirani, R., & Friedman, J. (2009). *The elements of statistical learning: Data mining, inference, and prediction* (2nd ed.). Springer. https://doi.org/10.1007/978-0-387-84858-7

Haug, L. E. (2019). *Analyzing the quality of Twitter data streams and sentiment architecture* [Technical report]. HP Laboratories.

Hayran, A., & Anagün, A. S. (2020). Generalized durative event detection and sentiment analysis on social media streams. *IEEE Access, 8*, 142100–142112.

Hosmer, D. W., Jr., Lemeshow, S., & Sturdivant, R. X. (2013). *Applied logistic regression* (3rd ed.). John Wiley & Sons. https://doi.org/10.1002/9781118548387

Hutto, C. J., & Gilbert, E. (2014). VADER: A parsimonious rule-based model for sentiment analysis of social media text. *Proceedings of the International AAAI Conference on Web and Social Media, 8*(1), 216–225. https://doi.org/10.1609/icwsm.v8i1.14550

İlhan, N., & Sağaltıcı, D. (2020). Twitter'da duygu analizi [Sentiment analysis on Twitter]. *Harran Üniversitesi Mühendislik Dergisi, 5*(2), 110–121.

Janetzko, H. (2017). Visual sentiment analysis on Twitter data streams. In *Proceedings of the International Conference on Information Visualization* (pp. 88–95). IEEE.

Jurafsky, D., & Martin, J. H. (2023). *Speech and language processing: An introduction to natural language processing, computational linguistics, and speech recognition* (3rd ed. draft). https://web.stanford.edu/~jurafsky/slp3/

Kaynar, O., Görmez, Y., Yıldız, M., & Albayrak, A. (2016). Makine öğrenmesi yöntemleri ile duygu analizi [Sentiment analysis with machine learning techniques]. *Veri Bilişimi Dergisi, 2*(1), 11–20.

Kırık, A. M. (2021). Sosyal medyada dijital aktivizm ve kamuoyu oluşturma [Digital activism and public opinion formation in social media]. *Dijital İletişim Araştırmaları, 3*(1), 45–58.

Kırık, A. M., & Özkoçak, V. (2023). Medya ve iletişim bağlamında yapay zekâ tarihi ve teknolojisi [History and technology of artificial intelligence in media and communication]. *İletişim ve Diplomasi*, (9), 77–96.

Küçüksille, E. U., & Selver, M. A. (2019). E-ticaret sitelerindeki Türkçe ürün yorumları üzerine makine öğrenmesi algoritmaları ile duygu analizi

[Sentiment analysis on Turkish product reviews using machine learning algorithms]. *Veri Bilimi, 2*(2), 35–48.

Kumaş, E. (2021). Türkçe Twitter verilerinden duygu analizi yapılırken sınıflandırıcıların karşılaştırılması [Comparison of classifiers in sentiment analysis from Turkish Twitter data]. *Dokuz Eylül Üniversitesi İktisadi ve İdari Bilimler Fakültesi Dergisi, 36*(2), 205–224.

Liu, B. (2012). *Sentiment analysis and opinion mining*. Morgan & Claypool Publishers. https://doi.org/10.2200/S00416ED1V01Y201204HLT016

Loria, S. (2018). *TextBlob: Simplified text processing* (Version 0.15.2) [Computer software]. https://textblob.readthedocs.io/

Manning, C. D., Raghavan, P., & Schütze, H. (2008). *Introduction to information retrieval*. Cambridge University Press. https://doi.org/10.1017/CBO9780511809071

McCarthy, J., Minsky, M. L., Rochester, N., & Shannon, C. E. (2006). A proposal for the Dartmouth summer research project on artificial intelligence, August 31, 1955. *AI Magazine, 27*(4), 12–14. https://doi.org/10.1609/aimag.v27i4.1904

Medhat, W., Hassan, A., & Korashy, H. (2014). Sentiment analysis algorithms and applications: A survey. *Ain Shams Engineering Journal, 5*(4), 1093–1113. https://doi.org/10.1016/j.asej.2014.04.011

Mitchell, T. M. (1997). *Machine learning*. McGraw-Hill.

Mohammad, S. M., & Turney, P. D. (2013). Crowdsourcing a word–emotion association lexicon. *Computational Intelligence, 29*(3), 436–465. https://doi.org/10.1111/j.1467-8640.2012.00460.x

Onan, A. (2016). Sentiment analysis on Twitter messages based on machine learning methods. *Journal of Information Systems and Management, 2*(1), 21–32.

Onan, A. (2018). Sentiment analysis on Twitter based on ensemble of psychological and linguistic feature sets. *Balkan Journal of Electrical and Computer Engineering, 6*(2), 69–77. https://doi.org/10.17694/bajece.419538

OpenAI. (2023). *GPT-4 technical report*. arXiv. https://doi.org/10.48550/arXiv.2303.08774

Özcan, C., & Çelik, M. (2019). Türkçe tweetlerde duygu kutupsallığı tespiti ve öznitelik çıkarımı [Sentiment polarity detection and feature extraction in Turkish tweets]. *Bilişim Teknolojileri Dergisi, 12*(3), 215–226.

Pak, A., & Paroubek, P. (2010). Twitter as a corpus for sentiment analysis and opinion mining. In *Proceedings of the Seventh International Conference on Language Resources and Evaluation (LREC'10)* (pp. 1320–1326). European Language Resources Association (ELRA).

Pang, B., & Lee, L. (2008). Opinion mining and sentiment analysis. *Foundations and Trends in Information Retrieval, 2*(1–2), 1–135. https://doi.org/10.1561/1500000011

Pang, B., Lee, L., & Vaithyanathan, S. (2002). Thumbs up? Sentiment classification using machine learning techniques. In *Proceedings of the 2002 Conference on Empirical Methods in Natural Language Processing (EMNLP 2002)* (pp. 79–86). Association for Computational Linguistics. https://doi.org/10.3115/1118693.1118704

Rish, I. (2001). An empirical study of the naive Bayes classifier. In *IJCAI 2001 Workshop on Empirical Methods in Artificial Intelligence* (Vol. 3, No. 22, pp. 41–46).

Russell, S., & Norvig, P. (2020). *Artificial intelligence: A modern approach* (4th ed.). Pearson.

Salton, G., & Buckley, C. (1988). Term-weighting approaches in automatic text retrieval. *Information Processing & Management, 24*(5), 513–523. https://doi.org/10.1016/0306-4573(88)90021-0

Sætra, H. S. (2023). Generative AI: Here to stay, but for good? *Technology in Society, 75*, Article 102372. https://doi.org/10.1016/j.techsoc.2023.102372

Sucu, İ. (2019). Yapay zekânın toplum üzerindeki etkileri [Effects of artificial intelligence on society]. *İletişim Çalışmaları Dergisi, 5*(1), 45–60.

Taboada, M., Brooke, J., Tofiloski, M., Voll, K., & Stede, M. (2011). Lexicon-based methods for sentiment analysis. *Computational Linguistics, 37*(2), 267–307. https://doi.org/10.1162/COLI_a_00049

Tümüklü Özyer, G., & Özyer, B. (2018). Twitter profesyonel izleme ve analiz aracı [Twitter professional monitoring and analytics tool]. *Teknik Bilimler Dergisi, 8*(2), 55–64.

Tümüklü Özyer, G., & Özyer, B. (2020). Turkish sentiment analysis on social media. *Sakarya University Journal of Science, 24*(5), 980–992.

Vapnik, V. N. (1998). *Statistical learning theory*. John Wiley & Sons.

Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., Kaiser, Ł., & Polosukhin, I. (2017). Attention is all you need. *Advances in Neural Information Processing Systems, 30*, 5998–6008.

Vural, A. G., Cambazoglu, B. B., Senkul, P., & Tokgoz, Z. O. (2013). A framework for sentiment analysis in Turkish cell phone reviews. In T. Czachórski, E. Gelenbe, & R. Lent (Eds.), *Computer and information sciences III* (pp. 437–445). Springer. https://doi.org/10.1007/978-1-4471-4594-3_45 *(Not: Orijinal metninizde bu yayın SIU konferansı olarak yazılmış ancak makalenin gerçek yayını ISCIS / Springer kitap bölümüdür. Eğer elinizdeki metne sadık kalmak isterseniz şu şekilde kullanabilirsiniz: Vural, A. G., Cambazoglu, B. B., Senkul, P., & Tokgoz, Z. O. (2013). A framework for sentiment analysis in Turkish cell phone reviews. In 2013 21st Signal Processing and Communications Applications Conference (SIU) (pp. 1–4). IEEE.)*

Wang, H., Can, D., Kazemzadeh, A., Bar, F., & Narayanan, S. (2012). A system for real-time Twitter sentiment analysis of 2012 U.S. presidential election cycle. In *Proceedings of the ACL 2012 System Demonstrations* (pp. 115–120). Association for Computational Linguistics.

Yıldırım, S. (2020). *BERT-base Turkish sentiment model* [Machine learning model]. Hugging Face. https://huggingface.co/savasy/bert-base-turkish-sentiment-cased

Yılmaz, K., & Arslan, H. (2022). Yapay zekâ sohbet robotlarının Türkçe metin işlemedeki başarımı [Performance of AI chatbots in Turkish text processing]. *Yapay Zekâ ve Veri Bilimi Dergisi, 4*(1), 12–24.

CHAPTER 4

ARTIFICIAL INTELLIGENCE FOR CYBER RESILIENCE IN DIGITAL GOVERNMENT

ÖMER SEVİNÇ[1]

Abstract Governments depend on digital identity, tax, social protection, justice, health, election, and emergency systems whose failure can affect rights and essential services. Artificial intelligence can improve cyber situational awareness by correlating events, detecting anomalies, prioritizing vulnerabilities, and supporting incident response across distributed public infrastructure. It can also amplify phishing, disinformation, deepfakes, and attacks against public decision systems. This chapter presents a mission-oriented approach to government cybersecurity that links AI use to service criticality, democratic accountability, procurement, and continuity. It proposes controls for shared visibility, zero-trust access, secure AI acquisition, human authorization, public transparency, and cross-agency response.

Keywords digital government, cyber resilience, artificial intelligence, critical infrastructure, public services, accountability

1 Dr., Ondokuz Mayıs University, Vezirköprü Vocational School, Department of Computer Programming.

Introduction

Digital government turns databases and networks into channels through which people exercise rights and receive services. Identity systems support benefits, taxation, licensing, health, education, and voting administration. A cyber incident can therefore deny more than access to a website; it can prevent a person from receiving income support, filing a legal document, or obtaining an official record. Government cybersecurity must prioritize mission continuity, integrity, and public trust.

AI offers useful scale. Public organizations may operate legacy systems, cloud platforms, municipal networks, and sector-specific infrastructure with uneven visibility. Machine learning can correlate logs and threat intelligence, detect anomalous behavior, and rank remediation. The same capability creates public-law concerns when automated security actions affect citizens, employees, or public records. Government use requires traceable authority, documented purpose, proportionality, and review (NIST, 2023; OECD, 2025).

Threats to Digital Government

Credential theft and exploitation of exposed systems remain common entry points. Shared suppliers can create systemic risk because one compromise may affect multiple agencies. Legacy applications may not support modern identity or logging controls. Operational technology in transport, water, energy, and emergency services adds physical consequences and long replacement cycles.

AI changes the speed and credibility of social engineering. Attackers can produce localized messages, imitate officials, synthesize audio, and adapt campaigns using breached information. Deepfakes can create confusion during elections, emergencies, or diplomatic events even when technical systems remain intact.

Defensive planning must connect cyber incident response with public communication and evidence verification.

Government AI systems may also be attacked directly. Manipulated inputs can distort fraud detection, border screening, resource allocation, or document classification. Training data may be poisoned through compromised sources. Generative assistants can reveal restricted information through prompt injection or excessive permissions. When a model supports a consequential public decision, integrity, provenance, and auditability become constitutional and administrative concerns as well as technical requirements.

AI Supported Cyber Operations

A shared analytics capability can combine identity events, endpoint telemetry, network flow, vulnerability data, and threat intelligence while allowing agencies to retain control of sensitive records. Federated or privacy-preserving approaches may be appropriate when raw data cannot be centralized. Models can rank events by mission impact, such as whether the affected asset supports emergency dispatch or public payments.

AI can help analysts summarize alerts, map indicators to known techniques, propose investigation queries, and identify related cases. Generated recommendations must cite their evidence and remain advisory unless an action falls within an approved automation playbook. Automated blocking may be suitable for clearly malicious domains, whereas disabling a public service or privileged account should require authorized review.

Vulnerability management benefits from contextual ranking. A simple severity score does not show whether a system is internet-facing, exploited, or essential to a public mission. An AI-supported process can combine these factors, but agencies must retain the underlying criteria so priorities can be explained and audited.

Governance and Procurement

AI procurement should define security requirements before market engagement. Contracts should address data location, training use, model and software updates, vulnerability disclosure, logging, subcontractors, incident notification, continuity, and secure exit. Agencies should obtain sufficient documentation to test the system and investigate failures. Proprietary status should not prevent public bodies from understanding controls that affect legal duties.

Governance can align the NIST AI Risk Management Framework functions of Govern, Map, Measure, and Manage with cybersecurity and service-continuity processes. Each use case needs an accountable owner, risk classification, impact assessment, approved data sources, performance measures, and retirement criteria. Independent review is especially important for systems used in law enforcement, benefits, taxation, or other areas where errors can burden individuals.

Public transparency should describe the purpose and general operation of an AI system without publishing details that enable attack. Citizens and staff need routes to challenge consequential outcomes. Security logs must support oversight while protecting classified, personal, and investigative information.

Control area	Government requirement	Assurance question
Authority	Named owner and legal basis	Who may approve and stop the system
Data	Approved sources and retention	Can provenance and access be audited
Model	Security and performance tests	What changes trigger revalidation
Operations	Incident and continuity plan	Can the service operate during failure
Public accountability	Notice and challenge route	Can an affected person seek review

Mission Based Risk Prioritization

Government portfolios are too large to protect every asset with equal intensity. Mission-based prioritization links technical

assets to statutory functions and public consequences. Agencies should identify services that protect life, deliver entitlements, maintain legal records, collect revenue, support elections, or coordinate emergencies. They can then determine recovery objectives, minimum operating capabilities, and dependencies on identity, telecommunications, cloud platforms, data exchanges, and suppliers.

AI can help construct dependency graphs and identify clusters of vulnerabilities, but prioritization rules must be understandable to accountable officials. A model should not quietly downgrade an old system merely because it generates few alerts; low activity may conceal a critical but infrequently used function. Decisions should incorporate exploitability, exposure, mission impact, replacement difficulty, and the availability of manual alternatives.

Protection of Digital Identity and Citizen Data

Digital identity is a high-value control point because one credential may unlock multiple services. Governments should apply phishing-resistant authentication to privileged and high-risk access, use federation carefully, and continuously review dormant accounts, delegated authority, and service credentials. Risk-adaptive authentication can reduce friction, but automated denials must account for travel, shared access points, assistive technologies, and citizens who cannot easily complete additional verification.

Citizen data should be separated according to purpose and sensitivity. Broad interagency access may improve service coordination while increasing the consequences of misuse or compromise. Data-sharing agreements need defined fields, authorized purposes, retention, logging, quality responsibilities, and termination procedures. AI analytics should use approved views rather than unrestricted copies wherever possible, and security

monitoring should not become a general-purpose system for profiling lawful behavior.

AI Enabled Influence Operations and Deepfakes

Synthetic media can target institutional trust even when no government network is breached. A fabricated recording may imitate an election official, emergency manager, judge, or senior administrator. Rapid rebuttal is difficult if agencies lack an established method for authenticating official communications. Governments should use consistent official channels, digitally signed material where appropriate, and public guidance on how authoritative updates can be verified.

Detection tools can support triage but should not be treated as definitive because generation and detection methods evolve together. Response teams should combine media forensics, source verification, contextual analysis, and coordination with affected officials. Public statements should avoid repeating harmful content unnecessarily and should explain the evidence supporting authenticity judgments. Exercises should include simultaneous cyber disruption and disinformation because adversaries may use each to amplify the other.

Secure Procurement and Supply Chain Assurance

Public agencies often acquire AI as a cloud service or as a feature embedded in a larger product. Procurement should require a software and model dependency inventory, secure development practices, vulnerability handling, access-control design, data-use restrictions, audit support, and continuity arrangements. Contracts should define who may use agency data to train or improve models and whether outputs or interaction logs leave the government's controlled environment.

Assurance should continue after contract award. Agencies need notice of material model, data, infrastructure, and subcontractor changes. They should test updates in a representative environment and retain the ability to delay or roll back changes that affect security or legal compliance. Exit planning is essential: data and logs must be exportable in usable formats, credentials must be revoked, and the service must be replaceable without an unacceptable interruption to public functions.

Cross Agency Coordination and Public Accountability

Cyber incidents rarely respect administrative boundaries. A shared supplier, identity service, or vulnerability may affect national, regional, and municipal organizations at the same time. Coordination mechanisms should define what indicators can be exchanged, how sensitive investigations are protected, who leads public communication, and when assistance can be requested. Common taxonomies and machine-readable formats allow AI-supported analysis without forcing every agency onto one platform.

Accountability requires records that show who authorized an AI use, which data and model version were involved, what security tests were completed, and how consequential outputs were reviewed. Oversight bodies need access to meaningful evidence rather than only vendor assurances. Publishing system registers, impact summaries, and aggregate incident information can strengthen public scrutiny while detailed configurations and active investigative data remain protected.

Conclusion

AI can improve government cyber resilience when it strengthens shared visibility and disciplined response. Its use should be tied to public missions rather than technology adoption targets. Agencies need secure identity, asset inventories, segmentation, tested backups, procurement controls, and cross-agency exercises

before advanced analytics can deliver dependable value. The central measure is whether essential services remain trustworthy and recoverable under attack, with decisions that responsible officials can explain and review.

References

Cybersecurity and Infrastructure Security Agency. (2024). Federal Civilian Executive Branch Operational Cybersecurity Alignment Plan. https://www.cisa.gov/resources-tools/resources/federal-civilian-executive-branch-fceb-operational-cybersecurity-alignment-focal-plan

European Parliament and Council of the European Union. (2024). Regulation (EU) 2024/1689 laying down harmonised rules on artificial intelligence. Official Journal of the European Union.

National Institute of Standards and Technology. (2023). Artificial Intelligence Risk Management Framework 1.0 (NIST AI 100-1). https://doi.org/10.6028/NIST.AI.100-1

National Institute of Standards and Technology. (2024). Cybersecurity Framework 2.0 (NIST CSWP 29). https://doi.org/10.6028/NIST.CSWP.29

Organisation for Economic Co-operation and Development. (2025). Governing with artificial intelligence: The state of play and way forward in core government functions. https://doi.org/10.1787/795de142-en

Vassilev, A., Oprea, A., Fordyce, A., & Anderson, H. (2024). Adversarial Machine Learning: A Taxonomy and Terminology of Attacks and Mitigations. NIST. https://doi.org/10.6028/NIST.AI.100-2e2023


BİDGE