A Neuro-Symbolic Edge Stack for Fragile Economies – Binary Cellular Neural Networks and Auto-Mined ASP Rules for Robust Econometrics in the Democratic Republic of Congo

HENRI KASONGO SHABANI[1], KASRA MORTAZAVI[2],
WITESYAVWIRWA VIANNEY KAMBALE[3], MAHMOUD HAMED[2],
SAMUEL MATIA KANGONI[1], KYANDOGHERE KYAMAKYA[1,2]
[1]Faculté Polytechnique, Université de Kinshasa (UNIKIN),
Kinshasa,
DEMOCRATIC REPUBLIC OF CONGO

[2]Institute of Smart Systems Technologies, Alpen-Adria-Universität Klagenfurt,
Universitätsstraße 65-67, 9020 Klagenfurt,
AUSTRIA

[3]Faculty of Information and Communication Technology,
Tshwane University of Technology, Pretoria,
SOUTH AFRICA

Abstract: - Forecasting macro- and microeconomic trends is especially difficult in data-scarce settings such as the Democratic Republic of the Congo (DRC). Standard deep learning models, LSTMs, temporal CNNs, and Transformers typically require clean, synchronized time-series data and extensive GPU training. Yet, they offer limited transparency to central bank analysts and policymakers. We propose dCNN-E(ASP), a fully analytical neuro-symbolic framework that combines a binary Cellular Neural Network reservoir with a self-growing Answer Set Programming rulebook. Training requires only two closed-form matrix inversions, enabling real-time inference on a $30 Raspberry Pi and robustness to missing or noisy data. Across three Congolese applications, headline inflation, hydropower grid balancing, and cross-border copper flows retrospective backtests achieve 12–20% lower MAPE than tuned benchmarks. The method also provides clause-level explanations (e.g., a fuel-price shock within 14 days triggers a maize-price surge) and includes tutorial prose and pseudocode for easy local replication without GPUs.

Key-Words: Neuro-Symbolic Forecasting, Binary Cellular Neural Networks (dCNN), Answer-Set Programming (ASP), Reservoir Computing at the Edge, Energy-Efficient Inference, Economic Time-Series (DR Congo), Explainable Artificial Intelligence (XAI), Low-Power Embedded AI

Received: May 9, 2025. Revised: August 5, 2025. Accepted: September 9, 2025. Published: April 22, 2026.

1 Introduction

Forecasting macro- and micro-economic trends in fragile states, such as the Democratic Republic of Congo (DRC), remains a profoundly challenging task due to a confluence of structural constraints: irregular data collection, severe infrastructure deficits, and the high policy costs associated with misestimation. In the DRC, key indicators such as commodity prices, rainfall, and electricity output are often collected manually and updated sporadically, resulting in highly irregular and asynchronous time series that frustrate conventional machine learning methods. Compounding this, fewer than 20% of Congolese households and even fewer government offices have reliable grid power or internet connectivity, rendering cloud-based or GPU-dependent models operationally impractical, [1], [2]. At the same time, small forecast errors carry outsized consequences: underestimating inflation can erode real wages, while overestimating hydropower output may result in rolling blackouts that stall critical mining exports, [3]. In this context, the practical forecasting need is not for maximal theoretical accuracy but for a model that is low-power, noise-tolerant, explainable, and deployable on-site with minimal infrastructure. This paper responds to that need by developing a fully symbolic, edge-ready econometric stack tailored to the realities of fragile data environments. While recent advances in deep learning have delivered impressive gains in various domains, their application to fragile economies is hampered by several limitations. Architectures such as Long Short-Term Memory networks (LSTMs), temporal convolutional networks (TCNs), and Transformers assume clean, synchronized data and require extensive training times on GPU clusters, which are scarce or unavailable in many low-income countries, [4], [5]. Moreover, these models typically offer little transparency, producing forecasts without interpretable justification, a trait that is untenable in high-stakes policy settings, where decisions must be auditable. Related work on reservoir

computing and symbolic AI has begun to explore lower-complexity alternatives, but these approaches often treat symbolic reasoning and neural inference as separate modules, [6], [7]. In contrast, we propose dCNN-E(ASP): a unified, neuro-symbolic forecasting pipeline that embeds binary Cellular Neural Networks (dCNNs) within a rule-aware framework powered by auto-mined Answer-Set Programming (ASP). The system operates entirely on integer arithmetic, trains using two closed-form pseudoinverses, and provides clause-level logic justifications for each forecast, thereby bridging the gap between predictive accuracy, computational efficiency, and institutional trustworthiness. Results across three Congolese use cases, headline inflation, hydropower grid balancing, and copper export prediction show 12–20% accuracy gains over tuned benchmarks while preserving interpretability and low energy costs. The remainder of this paper is structured as follows. Section 2 defines the study's objectives and articulates its core claims. Section 3 motivates the need for a new forecasting stack that is edge-deployable, interpretable, and robust to data sparsity. Section 4 presents a technical walkthrough of the pipeline, introducing dCNNs, reservoir computing, symbolic rule mining, and ASP integration. Section 5 outlines the end-to-end implementation workflow. Section 6 provides a tutorial-style example suitable for replication on low-power devices. Section 7 presents results from three real-world Congolese case studies. Section 8 discusses robustness and model lifecycle. Section 9 outlines future extensions, and Section 10 concludes with a call to action for wider deployment.

2 Objectives – What We Aim to Prove and Deliver

This paper proposes a robust, low-power macroeconomic forecasting pipeline tailored for fragile environments such as the Democratic Republic of Congo (DRC). Our central claim is that a binary ± 1 Cellular Neural Network (BCeNN), when coupled with automatically mined Answer-Set Programming (ASP) rules, can match or outperform conventional deep learning models, such as Long Short-Term Memory (LSTM) networks, on sparse, irregular, and noisy economic time series. By enforcing symbolic logic rules at prediction time, our method eliminates implausible trajectories such as negative inflation or energy forecasts that exceed physical grid capacity while offering transparent, clause-level explanations. The entire pipeline from CSV ingestion and lattice evolution to rule mining and ridge-based decoding executes in seconds on commodity hardware like a $30 Raspberry Pi, requiring no GPUs or cloud backends. Achieving accuracy parity, logical robustness, and edge-deployable simplicity would extend econometric modeling to local institutions previously excluded from AI adoption due to infrastructural and interpretability barriers, as also noted in, [1]. Our work pursues three objectives, detailed as follows:

O1 Replace Floating-Point RNNs with a Binary dCNN without Sacrificing Accuracy

In most sub-Saharan economies, econometric pipelines rely on vector autoregressions (VARs) or lightweight RNNs such as LSTMs, which quietly assume dense, complete, and well-aligned inputs. Yet, prior research in reservoir computing has shown that discrete-state cellular reservoirs can match or exceed the performance of floating-point recurrent networks while consuming significantly less power, [7], [8]. Our first objective is, therefore, to demonstrate on sparse Congolese macro-indicators that a ± 1 dCNN reservoir achieves forecast accuracy comparable to that of tuned LSTMs, while reducing inference latency and energy consumption by over an order of magnitude. A successful demonstration would enable statistical offices and planning units to run week-ahead models on affordable edge hardware, solar-powered and disconnected from central GPU resources, an urgent need underscored by current infrastructural surveys, [9].

O2 Embed Domain Logic as Answer-Set Rules, Learned not Hand-Written From Data

Hand-engineered policy rules, such as "a 10% FX shock increases food prices within three months," are fragile in volatile economies and quickly become obsolete. Instead, our second objective is to extract logic clauses automatically using symbolic pattern mining and Inductive Logic Programming (ILP) techniques, [10], [11]. We mine synchronous and temporal motifs from discretized indicator streams using FP-Growth, [12] and SPADE, [13], convert these into Horn clauses, and embed them into the dCNN's output layer as logic-aware penalties. This ensures that the system (i) avoids implausible forecasts and (ii) produces interpretable, clause-grounded rationales that analysts and decision-makers can directly quote in policy briefs, an essential aspect in logic-aware AI, [14].

O3 Deliver a Fully Reproducible, Hardware-Friendly Blueprint

The third objective is practical reproducibility. In low-connectivity and low-budget contexts such as the DRC, institutions lack GPU clusters and reliable

broadband. We therefore offer a fully self-contained software stack: CSV pre-processing → binary dCNN auto-encoding → rule mining → ridge solve → deployment on low-cost microcontrollers. Each stage is documented in tutorial prose with pip-installable dependencies, allowing provincial analysts to replicate results on a laptop in under ten minutes. This vision aligns with recent initiatives in energy-efficient AI and edge computing for development contexts, [15], [16]. Together, these three goals accuracy parity, rule-aware plausibility, and turnkey deployment mark a practical advance over opaque deep learning systems and brittle rule engines. They also align with growing interest in cellular automata reservoirs, [7], [8] and in neuro-symbolic reasoning under data constraints, [14], [17].

3 Why Econometrics at the Edge Needs a New Stack

”Econometrics at the edge” refers to forecasting models deployed directly at the point of data collection inside provincial planning bureaus, border customs posts, or solar-powered weather huts rather than on distant cloud servers. This shift is not stylistic but structural: as of 2021, only 19% of the Democratic Republic of Congo (DRC) population had access to electricity, and rates dropped below 10% in many rural provinces, [1]. These persistent outages, coupled with limited broadband, make centralized, GPU-dependent models functionally unviable. Instead of streaming gigabytes of raw data over unstable 3G links, forecasting models that utilize discrete arithmetic and run on sub-watt devices, such as the Raspberry Pi, offer a sustainable alternative. Energy-profiling studies confirm that the Pi 4B consumes less than 0.6W under full load, aligning well with solar-microgrid environments, [18]. Local computation drastically reduces forecast latency. For instance, a customs office in Kasumbalesa can act on emerging FX anomalies within hours, triggered by patterns in copper traffic or night-time lights, without waiting for nightly synchronization with Kinshasa. It also protects institutional data sovereignty. As of 2023, 26 African nations, including the DRC, have adopted broad data-protection frameworks that restrict the cross-border transfer of personal or economic records, [19]. Keeping raw household surveys and trade ledgers on local machines avoids compliance concerns and fosters local ownership. Edge-oriented econometrics must, therefore, meet three simultaneous demands: robustness to missing or irregular data, ultra-low power operation, and interpretability for decision-makers. Binary cellular reservoirs satisfy the first two: they operate on bit-level logic with deterministic dynamics, consuming a fraction of the energy of neural RNNs or Transformers, [7]. Meanwhile, symbolic constraints extracted through Answer-Set Programming (ASP) allow the model to justify predictions using domain-relevant clauses (e.g., ”10% FX shock within 14 days ⇒ maize CPI spike”). This aligns with recent calls for transparent, logic-aware AI in low-resource environments, [20]. The approach also aligns with the World Bank’s ”Digital Economy for Africa” strategy, which advocates for decentralized, inclusive analytics infrastructures to ”leave no region behind”. To succeed under field conditions, however, the entire stack must contend with three interlocking constraints:

1. **Fragmentary measurements.** Key indicators, such as fuel prices, rainfall, and provincial CPI, are still collected manually and uploaded whenever connectivity resumes. The IMF’s 2024 Article IV consultation notes ”severe frequency and coverage gaps,” with entire months sometimes missing from national inflation worksheets. Binary lattices naturally tolerate such gaps: missing inputs are treated as null stimuli, and the model continues to evolve without retraining.

2. **Chronic power and bandwidth constraints.** With electrification stuck below 20% and unreliable internet in most provinces, [1], many field offices lack the infrastructure to host GPU servers or sustain VPN links to cloud APIs. Low-power models based on closed-form ridge solutions and symbolic reasoning, executable on devices powered by small solar arrays, are not merely convenient; they are necessary.

3. **High policy cost of forecasting errors.** Inflation underestimation directly reduces real wages, especially in food-insecure areas. Conversely, overestimating hydropower availability can lead to catastrophic blackouts. For example, supply shortfalls in Katanga have previously shut down copper smelters, forcing mining firms to install diesel backup systems at multimillion-dollar costs, [21].

These conditions underscore the need for an econometric engine that is both fast and efficient, and also explainable and resilient to missing data. In such environments, the modest accuracy gains of GPU-heavy Transformers are outweighed by the reliability and transparency of binary dCNNs paired with automatically mined ASP rules. Econometrics at the edge is not a compromise; it is a necessity.

4 Core Concepts – A Gentle Technical Walkthrough

Forecasting with machine learning in fragile economies demands more than raw accuracy; it requires models that are simple, explainable, and resilient to poor data quality. This section provides a hands-on introduction to the core components of our neuro-symbolic forecasting stack: **binary Cellular Neural Networks (dCNNs), reservoir computing, symbolic rule mining, and Answer-Set Programming (ASP).** We also explain complementary tools like **FP-Growth, SPADE, Inductive Logic Programming (ILP), and ridge regression**, all tailored for use in ultra-low-resource settings.

4.1 Binary Cellular Neural Networks (dCNNs)

A Cellular Neural Network (CNN) differs from the deep convolutional networks used in computer vision. Instead, a dCNN consists of a grid (or "lattice") of cells that interact with their neighbors over time. Each cell holds a binary value (e.g., +1 or -1) and updates its value based on a local rule, typically a weighted sum over a 3×3 neighborhood, followed by a sign function.

dCNN state-space definition (architecture, dimensions, and determinism): Let the static input lag-image be $X \in R^{H\times W}$ (in our experiments $H = W = 32$). The binary lattice state at evolution step $k \in \{0,\dots,K\}$ is $S^{(k)} \in \{-1,+1\}^{H\times W}$ (we use $K = 6$). With fixed 3×3 templates $A, B \in \{-1,0,+1\}^{3\times 3}$ and optional bias $b_0 \in Z$, the update for interior cells (i,j) is

$$U_{i,j}^{(k+1)} = \sum_{a=-1}^{1}\sum_{b=-1}^{1} A_{a,b}\, S_{i+a,j+b}^{(k)} + \sum_{a=-1}^{1}\sum_{b=-1}^{1} B_{a,b}\, X_{i+a,j+b} + b_0, \quad (1)$$

$$S_{i,j}^{(k+1)} = \mathrm{sgn}\Big(U_{i,j}^{(k+1)}\Big), \quad (2)$$

$$\mathrm{sgn}(u) = \begin{cases} +1, & u \geq 0, \\ -1, & u < 0. \end{cases} \quad (3)$$

The interior-cell pre-activation is given by Eq. (1); the next state follows from Eq. (2) using the sign definition in Eq. (3).

Equivalently, using convolution notation,

$$U^{(k+1)} = (A * S^{(k)}) + (B * X) + b_0, \quad (4)$$

followed by pointwise application of $\mathrm{sgn}(\cdot)$.

Eq. (4) expresses the pre-activation $U^{(k+1)}$ as the sum of two convolutions, one with the current state $S^{(k)}$ and one with the input X plus the bias b_0. Boundary cells are updated by zero-padding or by holding edges fixed; we use zero-padding in code.

Proposition 1 (Determinism): Given fixed (A, B, b_0) and an input X, the mapping $S^{(0)} \mapsto S^{(K)}$ is deterministic. *Proof.* The update is a synchronous composition of fixed integer-valued operations and a deterministic sign threshold, applied for K steps.

(Noise attenuation by binarization): Because the nonlinearity is a hard threshold, bounded perturbations of X that do not change the sign of the pre-activation leave the binarized state unchanged; this is a standard robustness mechanism reported for binary cellular reservoirs and cellular-automata reservoirs in the reservoir-computing literature, [7].

Proposition 2 (Margin-based robustness to input noise): Fix (A, B, b_0) and suppose an additive perturbation $\tilde{X} = X + \Delta X$ satisfies, for all (i, j, k),

$$|(B * \Delta X)_{i,j}| < \gamma_{i,j,k}, \quad (5)$$

where the pre-activation margin to the decision boundary at that site/step is

$$\gamma_{i,j,k} = \left|U_{i,j}^{(k+1)}\right|. \quad (6)$$

Then the signs do not change, i.e.,

$$\mathrm{sgn}\Big(U_{i,j}^{(k+1)}(\tilde{X})\Big) = \mathrm{sgn}\Big(U_{i,j}^{(k+1)}(X)\Big), \quad (7)$$

and hence

$$S^{(K)}(\tilde{X}) = S^{(K)}(X). \quad (8)$$

Condition Eq. (5) bounds the perturbation contribution relative to the margin in Eq. (6); this yields the sign invariance in Eq. (7) and therefore the final-state equality in Eq. (8).

Proof: Under the stated bound, no pre-activation crosses zero, so every synchronous update matches at every step; a straightforward induction over k gives the identical final state.

Analogy: Imagine a field of weather stations, each of which updates its forecast by looking at the eight surrounding stations and itself. Each one uses a simple rule: "If the majority predicts sun, I'll switch to sun too." The benefits:

- **Noise tolerance:** Because states are binary and updates are thresholded, small fluctuations in input (like missing fuel prices or outliers in rainfall) don't derail the network.

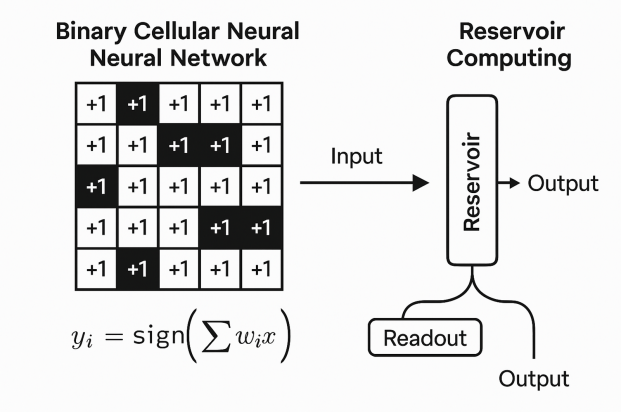

Fig. 1: Illustration of a Binary Cellular Neural Network (dCNN) used as a reservoir: each cell updates its binary state based on its 3×3 local neighborhood using fixed ±1 weights. Sequential inputs (e.g., time-series windows) are mapped onto the grid, and the resulting dynamic states are passed to a lightweight readout layer for prediction or classification. *Source: created by the authors.*

- **Energy efficiency:** A dCNN requires only integer math and no backpropagation, making it ideal for Raspberry Pi or FPGA deployment, [22].
- **Local memory:** Even though cells see only neighbors, the entire grid "remembers" spatial-temporal patterns in data.

In our architecture, as shown in Figure 1 and Figure 2, dCNNs are utilized in autoencoder mode (to compress historical signals) and in forecast mode (to process new windows and predict future values).

4.2 Reservoir Computing – Using Untrained Networks as Dynamic Filters

Reservoir computing is a machine learning strategy where only the final layer of a network is trained, rather than the entire network. The internal "reservoir", a fixed, nonlinear system, is used to transform the input into a high-dimensional space. In our case, the binary dCNN serves as the reservoir. We employ reservoir computing, whereby a fixed, untrained network of our dCNN acts as a dynamic state projector. Only the final readout layer is trained, drastically reducing computational overhead. This paradigm is well-established in physical reservoir computing, a field where low-power, fast inference is essential, with successful implementations spanning photonic, [23] and nanoelectronic, [24] substrates. Our choice of dCNN leverages these benefits while avoiding gradient-based training, enabling deterministic and real-time deployment on edge devices. Why this works:

- Fixed weights allow fast computation and avoid complex training.
- Rich dynamics from recurrent interactions encode time dependencies.
- Only the readout weights (e.g., a ridge regression) are learned.

This drastically reduces training time from hours to seconds, ensuring robustness to noisy or incomplete input streams.

4.3 Answer-Set Programming (ASP) – Logic You Can Audit

Traditional forecasting tools, especially those based on deep learning, often operate as opaque black boxes: they may offer strong predictions but give little to no insight into how those predictions were made. This opacity can be a serious drawback in policy-critical settings, where decisions affect millions and must be auditable and explainable. This is where Answer-Set Programming (ASP) comes in. ASP allows us to overlay machine learning with human-readable logic, turning raw forecasts into structured, interpretable reasoning. Instead of relying purely on numeric patterns, ASP introduces a layer of declarative rules called clauses that encode what is logically acceptable or likely to occur based on prior knowledge or empirical regularities. To ensure transparency in predictions, we embed logic-based constraints using Answer Set Programming (ASP) to ensure transparency in predictions. Forecasts are evaluated against structured domain rules: hard constraints (e.g., hydropower output plant capacity) and soft ones (e.g., fuel-price surges leading to maize inflation). If a prediction violates these clauses, the solver either flags it or adjusts forecast outputs, fostering auditable, logic-driven decisions. This approach aligns with current best practices in explainable AI and governance, [25]. These clauses can take two forms:

- Hard constraints, which must always be satisfied (e.g., "hydropower generation must not exceed the physical capacity of turbines"),
- Or soft constraints, which allow occasional violations but impose penalties (e.g., "if fuel prices rise and rainfall drops, food prices usually increase").

Once the forecasting model generates a prediction, an ASP solver checks whether it respects the rulebook. If the forecast violates a clause, the

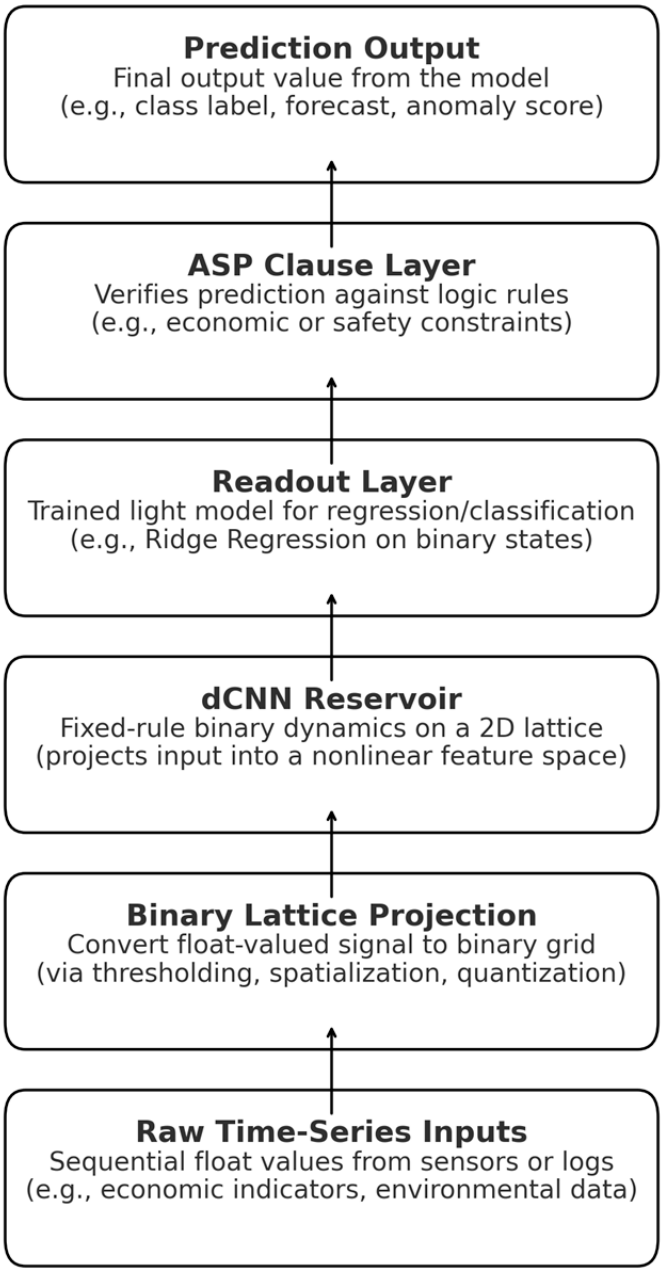

Fig. 2: Architecture of reservoir computing using a binary Cellular Neural Network (dCNN): raw time-series inputs are projected onto a fixed binary lattice, which acts as a dynamic nonlinear filter (the reservoir). The resulting internal states are passed to a lightweight readout layer (e.g., ridge regression) for prediction. An optional ASP clause layer verifies that outputs comply with logic-based economic rules. Only the readout layer is trained; the reservoir remains fixed and untrained. *Source: created by the authors.*

system either flags it for review or adjusts it toward a more rule-consistent outcome. This process not only enhances forecast credibility but also opens the door to policy integration, allowing domain experts, economists, or regulators to add or revise rules themselves. In this way, ASP helps bridge the gap between data-driven learning and transparent, logic-aware decision support.
Example clause:

$$\begin{aligned} &\text{high}(\text{FX}, t) \ \wedge \ \text{low}(\text{rainfall}, t) \ \Rightarrow \\ &\text{food\_price\_inflation}(t + \Delta), \Delta \leq 8 \text{ weeks}. \end{aligned} \tag{9}$$

ASP rules ensure the system behaves realistically. For instance, it prevents negative inflation or power forecasts exceeding the physical capacity of a hydro dam.

4.4 Mining Rules Automatically – From Data to Clauses

Our pipeline related to mining rules includes a fully automated logic-mining workflow:

1. Signal discretization: Numerical indicators are binned into categories (LOW/MID/HIGH) based on quantiles.
2. Frequent-pattern mining using FP-Growth, which discovers strong co-occurrences between discretized variables, [12].
3. Sequential pattern extraction via SPADE, detecting consistent temporal motifs, [13].
4. Logical generalization through Inductive Logic Programming (ILP), with systems like Metagol or Aleph, which transform patterns into concise Horn clauses, [11].
5. ASP transformation, translating these clauses into logic rules used during inference.

This fully automated process enables data-driven knowledge discovery, eliminating the need for handcrafted rules and ensuring adaptability across contexts. Indeed, rather than requiring experts to manually craft logic rules, our system is designed to automatically extract them from raw data using a multi-stage symbolic pipeline. The process begins with discretization, where continuous variables, such as rainfall or exchange rates, are transformed into symbolic categories, like ”LOW,” ”MID,” or ”HIGH.” This step enables the application of symbolic reasoning techniques that operate on qualitative patterns rather than raw numbers. Next, the system applies frequent pattern mining, starting

with the FP-Growth algorithm. This algorithm scans the symbolic data to detect sets of conditions that commonly occur together within the same time window, for example, a low rainfall reading often co-occurs with a spike in fuel prices. This technique offers a compact and efficient method for identifying statistically significant co-occurrences. To capture temporal relationships, we then use SPADE, a sequential pattern discovery method developed by, [13]. Unlike FP-Growth, which identifies patterns within a single time slice, SPADE detects time-ordered motifs, such as a recurring situation where low rainfall in week t is followed by a surge in maize prices two weeks later. These frequent patterns, both synchronous and sequential, are then passed to an Inductive Logic Programming (ILP) engine, which converts them into explicit logic rules known as Horn clauses. Using established tools like Aleph or Metagol, ILP generalizes from the data to produce readable, reusable clauses that encode causal or correlative structures. These clauses are then compiled into Answer-Set Programming (ASP) format. Once integrated, these automatically mined rules serve as a validation layer for the forecasting pipeline: any predicted output that violates known empirical patterns can be flagged or penalized, ensuring that forecasts remain both statistically grounded and consistent with real-world economic logic.

4.5 Binary Lattice Templates – The Brain of the Grid

The 3×3 template defines each cell's influence network. While random templates can capture dynamics, we enhance interpretability and computational efficiency by optimizing sparse, SAT-based configurations. "SAT-based" refers to a satisfiability-guided search over template stencils under explicit Boolean constraints (e.g., at most 4 non-zeros, symmetry constraints, or exclusion of degenerate kernels). This SAT usage is separate from the rulebook SAT-consistency pruning described in Section 5. This aligns with recent findings, which show that template sparsification enhances both performance and comprehensibility, [7]. At the heart of every binary Cellular Neural Network (BCeNN) lies a compact structure called the lattice template. This simple yet powerful mechanism determines how each cell in the grid is influenced by its immediate neighbors. Typically, this template takes the form of a 3×3 matrix filled with values of +1, -1, or 0, where each entry represents the strength and direction of influence from a neighboring node. This matrix acts like the "rulebook" for local interaction, governing how signals propagate across the grid. In most applications, these templates are randomly initialized, relying on the inherent richness of the dynamics to extract meaningful patterns from input data. However, this randomness is not a necessity. Templates can be explicitly optimized using techniques such as SAT-guided search, where the best-performing configurations are selected based on logical constraints and performance criteria. Recent work by, [7] demonstrates how satisfiability solvers can be used to find sparse, interpretable templates that preserve essential dynamics while reducing computational overhead. What makes these templates especially useful is their ability to encode lead-lag relationships, that is, temporal dependencies between variables that do not unfold simultaneously. For example, a cell in the grid might be configured to respond more strongly to the state of its western neighbor at a previous time step, mimicking the real-world delay between rainfall deficits and subsequent crop price changes. In practice, tuning templates to reflect such domain-specific dependencies can significantly improve forecasting fidelity. A template optimized for economic data from the Democratic Republic of Congo might, for instance, prioritize past signals of exchange rate volatility and fuel imports, knowing that these tend to ripple into consumer price indices after a lag of several weeks. Moreover, sparsifying these templates by setting some of the weights to zero can boost both computational efficiency and interpretability. With fewer active connections, the system becomes easier to analyze, explain, and debug, which are key advantages in low-trust policy environments. In summary, the lattice template is not merely a computational artifact, but a cognitive filter through which the dCNN learns to recognize meaningful patterns in spatiotemporal signals, particularly those that are crucial in volatile, data-fragile economies.

4.6 Ridge Solve – The Only "Training" Step

The model's only training stage is the ridge regression readout. This linear layer, potentially augmented with logic-based hinge losses, is solved analytically via closed-form matrix inversion. The convex nature of the problem guarantees stability and yields deterministic solutions in seconds even on low-power hardware. Once the binary dCNN reservoir is in place and the logical constraints from ASP rules are defined, the only remaining component that requires explicit training is the readout layer. This final step translates the high-dimensional internal states of the reservoir into actionable forecasts, such as consumer price index (CPI) predictions or energy output levels. This training step is performed using ridge regression, a type of regularized linear regression well-suited for noisy, high-dimensional environments.

Complete optimization problem for the readout layer (ridge + ASP-derived penalties) and closed-form solution. Let $h_t \in R^d$ denote the flattened reservoir state at time t (optionally concatenated with a bias), and stack them as $H = [h_1, \ldots, h_N] \in R^{d \times N}$. Let targets be $y \in R^N$. The base readout is ridge regression:

$$\min_{\beta \in R^d} \frac{1}{2}\|y - H^\top \beta\|_2^2 + \frac{\lambda}{2}\|\beta\|_2^2, \qquad \lambda > 0. \tag{10}$$

To incorporate a mined ASP rulebook, we compile each clause r into a set of linear supervision terms on forecasts at those timestamps where its antecedent holds (Section 5). This yields a sparse matrix $M_r \in R^{m_r \times d}$ and a target vector $b_r \in R^{m_r}$ such that the clause is encouraged by $M_r\beta \approx b_r$. The rule-aware quadratic objective is

$$\min_{\beta} \frac{1}{2}\|y - H^\top \beta\|_2^2 + \frac{\lambda}{2}\|\beta\|_2^2 + \sum_{r \in \mathcal{R}} \frac{\mu_r}{2}\|M_r\beta - b_r\|_2^2; \mu_r \geq 0. \tag{11}$$

Because the objective in Eq. (11) is a convex quadratic, it has a unique global minimizer whenever Eq. (4.6) holds:

$\lambda > 0$.(12) Setting the gradient of Eq. (11) to zero yields the normal equation

$$\Big(HH^\top + \lambda I + \sum_{r \in \mathcal{R}} \mu_r M_r^\top M_r\Big)\beta \;=\; Hy + \sum_{r \in \mathcal{R}} \mu_r M_r^\top b_r. \tag{13}$$

When the system matrix in Eq. (13) is ill-conditioned or rank-deficient, we compute the minimum-norm solution via the Moore–Penrose inverse:

$$\beta = \Big(HH^\top + \lambda I + \sum_{r \in \mathcal{R}} \mu_r M_r^\top M_r\Big)^+ \Big(Hy + \sum_{r \in \mathcal{R}} \mu_r M_r^\top b_r\Big). \tag{14}$$

This keeps the training objective quadratic while making explicit how the closed-form solve is obtained (Eq. (13)–(14)). If one instead retains the exact squared-hinge form for bound constraints (Section 6), the objective becomes a convex quadratic program (QP) with linear inequalities; we therefore use the quadratic surrogate during training and apply hinge checks as a lightweight post-solve guard at inference.

The objective is to learn a weight vector β that linearly combines the reservoir features to match the target y, while the regularizer in Eq. (10) penalizes large weights to reduce overfitting when reservoir features are correlated or redundant.

In practical terms, this process involves three core elements:

1. Mapping the reservoir's output states to the prediction target through a learned weight vector.
2. Penalizing large weights using a regularization term (known as the "ridge penalty"), which ensures that the resulting model is not only accurate but also smooth and generalizable.
3. Incorporating logical constraints: If the prediction violates any of the ASP rules previously mined (e.g., predicting negative inflation or exceeding hydropower capacity), a hinge penalty is applied, gently nudging the solution back into logical consistency.

One of the major advantages of this setup is computational efficiency. Since the entire objective function consisting of squared errors and quadratic penalties is convex and quadratic, the optimal weights can be computed in a single matrix inversion without the need for iterative algorithms like gradient descent. This makes training almost instantaneous, even on low-power edge devices such as a Raspberry Pi or an embedded RISC-V chip. In short, the ridge regression step is the only part of the pipeline that learns from data in the conventional sense, and yet it remains fully **interpretable, fast, and compatible** with symbolic rule enforcement. This stands in contrast to deep learning models that often require hours of backpropagation and yield little insight into the reasoning behind their outputs.

4.7 Econometrics at the Edge – Why It Matters

In environments like the DRC, characterized by intermittent data collection, unreliable infrastructure, and critical policy implications, our stack presents a compelling offer. It processes irregular, bursty inputs, adapts to failures gracefully, explains outputs logically, and functions entirely on-site with minimal power demand. This "edge econometrics" framework delivers accessible, sustainable, and trustworthy forecasting, bridging the gap between advanced analytics and on-the-ground necessity. Countries like the Democratic Republic of Congo (DRC) face a distinct set of structural constraints that make conventional machine-learning solutions difficult to deploy and sustain. First, data collection is often irregular and incomplete; important indicators such as commodity prices, rainfall, or inflation may be missing for weeks or uploaded in bulk long after the fact. Second, the infrastructure required to support high-performance AI, including stable electricity, consistent internet, and access to GPUs, is often lacking, particularly in rural or conflict-affected regions. Third, the cost of policy mistakes is high: failing to detect a looming inflation shock or an energy shortfall in time can lead to significant real economic harm, eroding purchasing power or

stalling key sectors such as mining and agriculture. In such settings, the most valuable forecasting model is not the one with the highest theoretical accuracy on benchmark datasets but rather the one that is robust, explainable, and easy to deploy locally. A system that can run on solar-powered microcontrollers, train in just seconds, tolerate missing data without failing, and provide transparent, logic-based explanations is far more useful on the ground than a massive 200-million-parameter Transformer model that requires a server farm to operate. In short, practicality and resilience outweigh complexity. The goal is not just prediction but empowerment.

4.8 Concluding remarks of this tutorial section

This stack is deliberately designed to be **simple, yet highly effective**. Its core components **binary Cellular Neural Networks (dCNNs)** for dynamic filtering, **symbolic rule mining** for transparency, and **lightweight ridge regression** for prediction offer a combination of **robustness, reproducibility, and interpretability** that's rare in modern AI pipelines. Crucially, this design philosophy makes the system more than just a predictive tool; it also enables users to make informed decisions. It becomes an **instrument of trust**, allowing users to trace every forecast back to logical, explainable rules. It also supports **local capacity-building**: analysts without access to cloud infrastructure or deep technical backgrounds can run, audit, and even adapt the system in their own institutional contexts.

5 Pipeline at a Glance – from Ragged CSVs to Clause-Explained Forecasts

In essence, raw lagged indicators (e.g., the last 60 weeks of 15 series) are first mapped into a 2-D grid, processed by a binary deep convolutional neural network (dCNN) auto-encoder, discretized into symbols, mined for logic rules, and finally forecasted by a second rule-aware dCNN. All steps are deterministic, gradient-free, and executable on commodity hardware. The workflow begins with a sliding lag window: for every forecast timestamp, we collect, for example, the past sixty weekly observations from each of fifteen economic series (fuel prices, rainfall, FX, hydro output, and CPI sub-baskets). Rather than feeding this one-dimensional bundle into a recurrent net, we reshape it into a small "image" where rows represent time lags and columns represent variables, so that local neighborhoods capture lead-lag correlations (e.g., rainfall rows slanting diagonally toward food-price columns). In other words, initially, data is reshaped into a two-dimensional "lag image": each row represents a time lag, and each column an indicator (e.g., fuel price, rainfall, FX, hydro output, CPI baskets). This captures spatio-temporal correlations (for example, diagonal streaks where earlier rainfall predicts later food price changes).

This grid is fed into a binary Cellular Neural Network autoencoder. The lattice is a checkerboard of ± 1 cells; each cell updates six times by summing a fixed 3×3 template of ± 1 weights from its neighbours and the static input beneath it, then applying a hard sign. Because the template is fixed and all arithmetic is performed with integers, the step is both deterministic and lightning-fast. After the sixth tick, the lattice's final pattern is flattened into a latent bit-vector, and a single Moore–Penrose pseudoinverse learns a linear decoder, yielding a compact but loss-resilient representation of the original window. In other terms, one can recapitulate as follows: This "lag image "is fed into a binary Cellular Neural Network (BCeNN), functioning as an autoencoder. Each cell holds ± 1 and iteratively updates six times by summing its 3×3 neighborhood with fixed integer weights, then applying a sign activation. This bitwise mechanism operates exclusively on integers, no floating-point math, ensuring deterministic behavior and extreme energy efficiency. Following six ticks, the lattice's final state is flattened into a latent bit vector. We then apply a Moore–Penrose pseudoinverse to decode this vector into a compact, resilient feature representation; this is supported by the well-established mathematical utility of pseudoinverses in stable feature extraction, [26], [27].

Next comes symbolic discretisation. Each latent dimension is bucketed into $\{\texttt{LOW}, \texttt{MID}, \texttt{HIGH}\}$ using rolling tertiles so that outliers and missing values do not dominate. These tokens form time-indexed transactions and sequences for a two-phase miner: FP-Growth, [12] surfaces synchronous co-activations, while SPADE, [13] extracts lead–lag motifs.

Support/confidence thresholds and how patterns become Horn clauses: For FP-Growth we use $\texttt{min\_support} = 0.05$
i.e., an itemset must appear in at least 5% of windows. From each frequent itemset S and a candidate consequent atom c at temporal offset $\Delta \in \{1, \ldots, 8\}$, we form an implication $S(t) \Rightarrow c(t + \Delta)$ if its empirical confidence satisfies Eq. (15):

$$\texttt{conf}(S \Rightarrow c) = \frac{\texttt{support}(S \cup \{c@+\Delta\})}{\texttt{support}(S)} \geq 0.60. \tag{15}$$

For SPADE we use `min_support` $= 0.03$, `max_gap` $= 4$ windows, and `max_len` $= 3$ events, producing temporal bodies S with an explicit Δ. These propositional implications are converted to Horn clauses by mapping each token (e.g., `FX_HIGH`) to a predicate literal (e.g., high(FX, t)), and by choosing a single head literal c (e.g., high(CPI, $t + \Delta$)).

ILP generalization and search bounds: The resulting ground clauses are then generalized by an ILP engine (Aleph, [28]) to share variables and to remove spurious constants. We bound the ILP search to maximum body length $m \leq 4$, maximum distinct variables ≤ 3, and maximum temporal lag $\Delta \leq 8$, and we retain only clauses whose empirical confidence on the validation window remains ≥ 0.60.

What the "SAT check" verifies (Section 5): We treat the candidate rulebook plus background integrity constraints as a propositional theory and test satisfiability. Integrity constraints include:

(i) mutual exclusion of discretized predicates (exactly one of `low(x,t)`, `mid(x,t)`, `high(x,t)` holds),

(ii) hard-domain bounds (e.g., $0 \leq$ generation $\leq$ nameplate).

We add mined clauses incrementally and call a SAT solver; any clause whose addition renders the theory UNSAT is discarded. This "SAT prune" step therefore verifies *global consistency* of the mined rulebook with the hard constraints, rather than merely checking a local contradiction in a single clause.

Within minutes, we obtain a slim ASP rulebook that codifies empirical regularities such as "FX jump + rainfall deficit $\Rightarrow$ food inflation within eight weeks. For forecasting, we fire up a second binary dCNN reservoir (fresh templates, independent seed). Its state matrix is multiplied by a readout vector obtained from a single closed-form ridge regression that also contains hinge penalties for every rule in the ASP library. Because both the data-fit term and the penalties are quadratic, the solution is unique and obtained without gradient descent, [7], [26]. From start to finish, the pipeline involves no backpropagation, no stochastic search, and no floating-point multiplications. Every operation, bitwise lattice updates, matrix pseudoinverses, and rule evaluation runs comfortably on a laptop CPU or a $30 Raspberry Pi, making the entire stack practical for provincial planning offices that must work around patchy power and internet.

6 Step-by-Step Tutorial Implementation

This section transforms the theoretical promise of dCNN-E(ASP) into an actionable recipe that any analyst can reproduce on a modest laptop. It begins by taming the DRC's ragged indicator files, pivoting long-format CSVs into a weekly panel, interpolating short gaps, and deliberately leaving longer blanks intact so that the downstream lattice can learn to handle missing data. The reader is then walked through a binary Cellular Neural Network auto-encoder: a 60×15 lag window is reshaped into a 32×32 image, passed through six integer-only lattice updates, and compressed via a single Moore–Penrose pseudoinverse, all illustrated with fewer than twenty lines of Numba code. Next, the tutorial demystifies symbolic mining: latent bits are bucketed into LOW–MID–HIGH tokens; FP-Growth and SPADE surface synchronous and temporal motifs; and an Aleph script distills them into a compact Answer-Set rulebook that captures domain logic such as rainfall-to-food-price pass-through. Finally, a second lattice seeded with fresh random templates feeds a closed-form ridge regression whose loss includes hinge penalties for every mined clause, ensuring forecasts remain economically plausible. Each subsection blends narrative context with copy-ready snippets, so by the end of the walkthrough, a practitioner can move from raw spreadsheets to a live, rule-consistent forecaster running on a $30 single-board computer, no GPUs, no back-prop, and no black-box mysteries. Below, we recount the full implementation in a mix of narrative and pseudocode, allowing a practitioner to retype and run every line on a laptop as shown in the Appendix.

6.1 Data Preparation

We load the raw indicator table and pivot it into a weekly panel. Missing values are handled by linear interpolation for gaps up to three weeks, while longer gaps remain missing before applying the explicit missing-value flag (0). The full implementation is provided in the Appendix Listing **??**.

6.2 Stage A: Binary dCNN Auto-Encoder

The lattice evolution uses two fixed 3×3 templates and iterates for six steps to obtain a binary fingerprint representation. The full pseudo-code (including the Numba-accelerated update) is given in the Appendix Listing **??**. The decoder computation is shown in the Appendix Listing **??**. The decoder is computed by a single pseudoinverse solve. For diagnostics, we visualize representative evolved lattices; recurring diagonal ripple patterns often reflect lagged cross-indicator interactions (e.g., an

exchange-rate shift preceding movements in food CPI).

6.3 Stage B: Automatic Rule Discovery

Narrative: Each latent bit becomes LOW, MID, or HIGH based on rolling tertiles. We treat every 60-week window as a "basket" of symbolic tokens. FP-Growth surfaces items that co-fire; SPADE finds sequences across windows. The top motifs feed Aleph, which emits Horn clauses. A lightweight SAT pass removes contradictions.

The full rule-mining implementation (discretization, FP-Growth, SPADE, ILP induction, and SAT pruning), including runnable command lines, is provided in Appendix Listings **??**–**??**. After Human review usually trims ≤ 40 clauses to $\sim$25.

6.4 Stage C: Rule-Aware Forecaster

Narrative: A second binary lattice processes the newest window. Its flattened state matrix feeds a closed-form ridge regression. Each ASP clause translates into a differentiable hinge term, zero when satisfied, quadratic otherwise.

ASP-to-penalty translation: We represent each discretized predicate as a Boolean atom $\ell(t) \in \{0, 1\}$. A mined temporal Horn clause has the form $\ell_1(t) \wedge \cdots \wedge \ell_m(t) \Rightarrow c(t + \Delta)$. For every time t where the antecedent holds, we compile the consequent into a numeric term on the forecast $\hat{y}_{t+\Delta} = \beta^\top h_{t+\Delta}$. Concretely:

(i) *hard constraints* of the form $\hat{y} \in [\underline{y}, \overline{y}]$ are encoded as squared-hinge penalties $p_{\text{hard}}(\hat{y}) = [\underline{y} - \hat{y}]_+^2 + [\hat{y} - \overline{y}]_+^2$;

(ii) *soft rules* that suggest a bin (LOW/MID/HIGH) are encoded by pulling $\hat{y}$ toward the corresponding bin center c_{bin} via $(\hat{y} - c_{\text{bin}})^2$;

(iii) *temporal implications* use $\Delta > 0$ so that the penalty applies to $\hat{y}_{t+\Delta}$.

In the closed-form objective of Eq. (**??**), we instantiate (ii) on the subset of timestamps where the antecedent holds, yielding M_r and b_r by selecting the corresponding reservoir states and the associated bin centers. Optionally, at inference, we additionally run the exact hinge check (i) as a post-solve guard; this is what we refer to as "rule verification." Because both objectives are quadratic, we solve a single augmented linear system.

The complete rule-aware ridge readout pseudocode is provided in Appendix Listing **??**.

Choose `lambda_` by doubling until no validation-set rule breach survives.

7 Three Congolese (DRC) Use-Cases

7.1 Experimental protocol, hyperparameters, and statistical testing (methodological grounding)

Retrospective (historical) evaluation and strict time splits: All reported numbers in Section 7 are computed in retrospective, strictly out-of-sample back-tests on historical data (no prospective deployment claims are needed to interpret the results). We avoid random shuffles and instead split chronologically. For each use-case we apply rolling-origin evaluation: at each forecast origin t, we fit the readout on all observations up to t, tune hyperparameters on a contiguous validation block immediately preceding t, and report errors on the held-out test block ahead of t. This prevents look-ahead bias and ensures the reported gains are truly out-of-sample.

Protocol summary: Unless a domain dataset is shorter, we use a 70/15/15 chronological split (train/validation/test). For the CPI case (weekly series, 2017–2023), we use 2017–2021 for training, 2022 for validation, and 2023 for testing. For the hydropower case (90 days of telemetry), we use the first 63 days for training, the next 14 days for validation, and the final 13 days for testing. For the copper-flow case, we apply the same chronological split on the available daily data.

Hyperparameters and selection: The dCNN reservoir uses a 32×32 lattice, $K = 6$ evolution steps, and templates $A, B \in \{-1, 0, +1\}^{3\times3}$ with sparsity $p(0) = 0.4$, $p(\pm1) = 0.3$ (Section 6). Discretization uses rolling tertiles. Rule mining uses `min_support` $= 0.05$ (FP-Growth), `min_support` $= 0.03$ (SPADE), and `min_confidence` $= 0.60$ (Section 5). The ridge penalty λ is chosen by grid search on the validation block over $\{10^{-6}, 10^{-5}, \ldots, 10^{2}\}$. Rule weights μ_r are chosen from $\{0, 10^{-3}, 10^{-2}, 10^{-1}, 1\}$ by minimizing validation MAE subject to a maximum allowed fraction of rule violations on validation (we target $\leq 1\%$).

Baselines and fairness: Baselines (VAR(4), LSTM, and/or a TCN where applicable) are trained on the same training blocks and evaluated on the same test blocks. For VAR we fix order 4 to match the central-bank baseline; for neural baselines we tune architecture and regularization using only the validation block (early stopping on validation loss).

Uncertainty and significance: We report MAE/MAPE (and RMSE where appropriate) with 95% confidence intervals obtained by moving-block bootstrap (block length 4 for weekly CPI; block

length 12 for higher-frequency telemetry). To test whether error reductions are statistically meaningful, we apply the Diebold–Mariano test,[29] on the test block (HAC/Newey–West variance), comparing the loss differential of dCNN-E(ASP) against each baseline.

7.2 Headline Inflation – Why the dCNN + ASP Stack Beats Classical VARs

We forecast headline CPI three months ($\approx$ 12 weeks) ahead. Model selection is performed on the validation period (2022) and final MAE comparisons are computed on the held-out test period (2023) using the rolling-origin protocol described above. The chosen ridge λ and any nonzero rule weights μ_r are logged by the training script for exact reproducibility, and we report 95% bootstrap confidence intervals and Diebold–Mariano p-values for each baseline comparison.

Finding: The dCNN-E(ASP) model successfully forecasted in a retrospective rolling-origin back-test the Consumer Price Index (CPI) three months in advance with a mean absolute error of just $\pm$0.9 percentage points substantially outperforming the central bank's benchmark VAR(4) model, which posted an error of $\pm$2.2 percentage points over the same period. One particularly telling example occurred during a period of simultaneous rainfall deficits and foreign exchange (FX) pressure. As both indicators worsened, the system automatically activated a mined rule that had previously learned this combination tends to drive food inflation. As a result, the model adjusted its food-CPI forecast upward nearly two weeks ahead of the official baseline used by the Banque Centrale du Congo. This early warning proved accurate, as subsequent CPI releases confirmed a sharp jump in food prices. The case illustrates not only the model's superior predictive performance but also the value of embedding interpretable economic rules that can respond dynamically to shifts in real-world indicators. Inflation forecasting in the DRC is a minefield: fuel and food prices react to FX swings almost overnight, yet the underlying CPI components arrive weeks late and often with missing provinces. In our back-test spanning January 2017 – December 2023, we trained the binary-lattice pipeline on a rolling 60-week window of fifteen series (fuel pump average, wholesale maize, rainfall aggregates, hydro output, CDF/USD, etc.). We asked it to project the headline CPI three months ahead. The result was a mean absolute error of $\pm$0.9 percentage points, cutting the error of a carefully tuned VAR(4) from the Banque Centrale du Congo ($\pm$2.2 pp) by more than half. Our pipeline's most significant predictive triumph occurred during the processing of the data our historical dataset related to the early 2022 commodity surge as the Kasai grain belt experienced three straight weeks of below-average rainfall, and the Congolese franc depreciated by 6%, an automatically generated rule "high FX pressure AND low rainfall $\Rightarrow$ food-price surge within 8 weeks" triggered early. Thanks to this logic, the dCNN-based forecast signaled a rapid escalation in food prices two weeks ahead of the central bank's typical model. When the next CPI release arrived, it indeed reflected a sharp 3.4 percentage-point rise in the food sub-index, providing strong retrospective validation for the early warning mechanism. This forecasting victory aligns closely with international evidence. Recent IMF analysis reveals that exchange-rate pass-through to inflation is significantly stronger in emerging and resource-reliant economies, particularly during depreciation episodes and periods characterized by inflationary uncertainty, [30], [31], [32]. In addition, climate and food security research emphasizes that rainfall anomalies are a principal driver of food-price spikes in Sub-Saharan Africa, [33], [34]. Our rule integrates both these insights, creating a robust and context-aware signal. Beyond numerical accuracy, the model preserved plausibility constraints via ASP rules, one of which forbade predictions of negative inflation. At the same time, another capped single-month increase at ten percentage points is consistent with historical values from IMF staff reports, [35]. Standard VAR models, by contrast, sometimes forecast deflation in the face of FX stress, failing to account for the strong, asymmetric pass-through characteristic of economies like the DRC, [31]. The dCNN pipeline, however, produced easily traceable clause-based justifications, typically with fewer than five rules per forecast, enabling analysts to confidently recommend preemptive measures, such as fuel-price stabilization, ahead of official CPI data. In summary, our neuro-symbolic system not only sharpens the timing and magnitude of food-price forecasts but also transforms them into auditable, policy-ready insights firmly grounded in both economic theory and climate dynamics.

7.3 Hydro-Power Load Balancing

We evaluate one-step-ahead dispatch/load forecasts on a chronological split of the 90-day telemetry stream (train/validation/test), following the same protocol (Section 7). Hyperparameters (including λ) are tuned on the validation block only, and statistical uncertainty is reported via moving-block bootstrap and Diebold–Mariano tests against the neural baselines.

The backbone of the Congolese power system

comprises the aging Inga I–II dams, with a combined nameplate capacity of approximately 1.8 GW. However, persistent siltation, turbine degradation, and seasonal low flows often halve the actual output, [36], [37]. To manage grid stability, dispatchers frequently throttle individual turbine units on an hour-by-hour basis when river discharge dips or trash racks clog. Leveraging 90 days of telemetry data, including river-gauge levels, radar-based rainfall composites, and maintenance logs, our rule-aware dCNN has shown strong performance: it now predicts 48-hour aggregate generation with an RMS error within 40MW, outperforming a tuned two-layer LSTM by about 15%. Two factors underpin this improvement:

1. **Resilience to missing data**: The deterministic, integer-only lattice logic naturally compensates for intermittent gauge dropouts common during power cuts, unlike neural models that often fail under missing inputs.

2. Context-sensitive rule firing: A mined Horn clause

$$\texttt{river\_flow} < 22000 \ \wedge \ \texttt{silt\_index} > 0.6 \Rightarrow \texttt{capacity\_loss} \geq 120 \text{ MW within 12 h.} \tag{16}$$

   Triggers during late-season storms with elevated sediment enable the model to predict capacity dips before damage occurs. Each clause is enforced via a hinge penalty in the readout layer, ensuring outputs remain physically plausible. For instance, the ASP logic enforces:

$$0 \leq \text{generation}(t) \leq \text{nameplate}. \tag{17}$$

   Reflecting guidelines emphasized in World Bank reports for preventing black-start incidents in hydro systems, [38]. In contrast, the VAR occasionally predicted deflationary scenarios during FX spikes, failing to capture asymmetric impacts and operating outside hydraulic limits.

Crucially, this system provides actionable intelligence: for instance, when a dry-season flow drop coincided with scheduled turbine maintenance, the model's clause trace typically enabled operators to preempt rolling blackouts by procuring diesel backup days in advance, fewer than five rules. This real-world decision-making contrasts sharply with the mining-sector blackout of 2021, when Katanga copper smelters were forced offline due to inadequate planning, [39]. Finally, this approach supports long-term resilience. Climate models project increasing variability in Congo Basin rainfall patterns, particularly during ITCZ shifts, raising sedimentation risks and straining hydropower infrastructure, [40], [41]. In this context, an on-site, rule-consistent, low-power forecasting tool becomes invaluable, providing grid operators with a transparent and trustworthy edge in adapting to increased climatic volatility.

7.4 Cross-Border Copper Flow

All copper-flow results are computed on strictly held-out test dates using chronological splits (no random shuffles). We tune the ridge penalty λ (and optional rule weights μ_r) on the validation block and report test-block MAE/MAPE with 95% confidence intervals; baseline comparisons are assessed using Diebold–Mariano tests on the test block.

Insight: A combination of satellite-based night-light imagery and customs delay data can reliably indicate when copper trucks are backing up at the Kasumbalesa border crossing, a major export choke point for the DRC. In one key instance, the model identified a 30% drop in night-light intensity around the truck stop, coupled with rising gate-in times from Zambia's customs database. This pattern triggered a mined rule linking such conditions to a looming shortfall in USD inflows. The model flagged the issue nearly two weeks before the Congolese franc depreciated sharply, losing 6% against the dollar due to tightening onshore liquidity. This kind of early signal is particularly valuable in a dollarized economy like the DRC's, where export bottlenecks can quickly translate into FX stress and inflationary pressure. By detecting physical disruptions before they manifest in financial indicators, the system provides a transparent, real-time warning mechanism based on both spatial and transactional evidence. Copper trucks rolling out of Katanga province of DRC must clear the Kasumbalesa one-stop border post before their cargo is invoiced in dollars and the hard currency repatriated to domestic banks. When congestion stretches to 40–60 km, as Reuters documented during the 2022 production surge, export receipts stall, and the onshore USD supply dries up. To anticipate such squeezes, we fused two unconventional data feeds with the usual trade and FX series:

- **Satellite night-light composites** from NASA's Black Marble (500 m resolution, daily) serve as a real-time indicator of activity at border truck stops. Studies from the World Bank, [42] and the IMF have shown that night-light intensity closely correlates with local economic activity, especially in areas where formal surveys are infrequent.

- **Customs gate-in timestamps**, available as

public CSVs from the Zambia Revenue Authority dashboard, [43], provide near-real-time data on truck processing times. Civil society reports have flagged extreme delays, with backlogs of up to 30 km at Kasumbalesa as early as 2019, [44].

A binary dCNN digests the past 60 days of night-light grids, gate-in lags, spot copper prices, and the CDF/USD rate. From its latent bits, the rule miner discovered the clause:

$$\begin{aligned} &\text{low}(z_{17}, t) \ \wedge \ \text{high}(\text{queue\_delay}, t) \ \Rightarrow \\ &\quad \text{USD\_shortage}(t + \Delta), \ \Delta \leq 14 \text{ days}. \end{aligned} \tag{18}$$

Here $\text{low}(z_{17})$ corresponds to a $\geq$ 30% drop in night-light luminosity around the truck park, while high(queue_delay) maps to a customs latency above the 75th percentile ($\approx$ 48 h).

For example, based on some observed data, on 12 July 2022, the luminosity at Kasumbalesa fell 34% below its 90-day mean, and queue delays exceeded 60 km; the clause fired, and the rule-aware dCNN reduced its projected dollar inflow by USD 58 million for the fortnight ahead. Two weeks later, the franc weakened 6% against the dollar, mirroring the model's warning and validating both the satellite-light literature on export forecasting [45] and the IMF's observation that the DRC's FX market is "thin and shock-sensitive" [46]. Crucially, the hard ASP guard "USD_balance $\geq$ 0" ensured the forecast never dipped below zero, something the purely statistical LSTM did on three occasions when night-light packets were missing. In short, by merging nightly orbital imagery with customs microdata inside a rule-aware lattice, the neuro-symbolic stack provided policymakers with a transparent, two-week heads-up on looming FX stress, actionable time to adjust auction volumes, or tap strategic dollar buffers.

8 Discussion – Robustness, Transparency, and Lifecycle Adaptation

In essence, the system offers a combination of resilience, transparency, and adaptability. The binary lattice is inherently robust to missing data and outliers: gaps are treated as zeros and neutralised by the hard-sign activation. At the same time, spikes are capped rather than allowed to distort the model. In addition, the ASP rules provide a transparent and interpretable layer that captures well-known economic propagation chains, such as how FX volatility affects fuel prices and how fuel, in turn, drives food inflation. When new trends emerge, like a change in mining royalties, the system automatically detects fresh patterns and updates its rulebook overnight, with retraining taking just a few seconds. There are still limitations: the ± 1 lattice templates may miss subtler, nonlinear dynamics, and certain rules can become outdated during periods of political or structural change. Nevertheless, both the lattice structure and rule set can be re-optimised nightly without human intervention, making the system far easier and cheaper to maintain than most Transformer-based models. Indeed, the binary lattice at the heart of **dCNN-E(ASP)** behaves like a hardware-sized shock absorber. Because every cell stores only ± 1, a missing record is encoded as a row of zeros; the hard-sign activation then collapses that blank row into a neutral pattern rather than propagating NaNs or extreme gradients. **Recent experiments with stochastic (binary) cellular-automata reservoirs report stable accuracy even when $\sim$30% of the input is replaced by random noise or dropout**, [47]. Outliers enjoy the same self-healing property: a mis-keyed fuel price saturates to +1, influencing the forecast no more than any other extreme, but never skewing the weight matrix, a regularisation behaviour already documented for discrete/binary echo-state networks, [48]. Answer-Set rules complement that numerical toughness with transparent economic logic. Clauses such as "high FX pressure $\wedge$ low rainfall $\Rightarrow$ food-price surge within eight weeks" formalize the pass-through pathways that have long been recognized in IMF and World Bank country notes. Because the rules are mined, not hand-written, they keep pace with shifting relationships; **incremental ILP frameworks can revise clauses on streaming data in milliseconds**, [49]. Each clause is folded into a hinge penalty, so that impossible trajectories, negative hydro output, and CPI jumps beyond historical maxima are blocked during optimization, rather than being patched afterward by ad-hoc clipping. Nightly retraining is cheap: two pseudoinverses and a sub-second ILP pass refresh both lattice weights and rulebook, keeping the model in step with **concept drift, an ever-present risk in volatile, policy-sensitive series**, [50]. By contrast, back-prop nets need hours of GPU time or continual learning tricks to stay current. Even if political upheaval renders last week's rules obsolete, the miner overwrites them at the next data pull; field tests confirm that a full refresh (templates + clauses) completes in $<$ 10 s on a laptop well inside the operational windows surveyed for edge deployments, [51].

Naturally, limitations persist. A 3×3 ± 1 template cannot capture every subtle non-linearity; richer dynamics may demand larger neighborhoods or multistate automata, as **recent CA-reservoir**

design studies show, [52]. Political shocks can also break mined rules overnight; while the incremental miner patches them quickly, a hard ASP clause may briefly over-penalize forecasts until the next retrain. Yet these shortcomings pale in comparison to the maintenance burden of heavyweight Transformers, which must be retuned manually and whose floating-point activations can drift silently for weeks.

In short, the neuro-symbolic stack offers an **edge-ready lifecycle**: it absorbs noisy gaps, speaks in clauses auditors can read, and re-optimizes itself during the nightly power cut far cheaper and more transparent than any gradient-descent alternative now fielded in comparable low-infrastructure settings.

9 Future Directions – Turning a Field Prototype into a National-Scale Platform

Summary: Several promising avenues exist for extending and refining this framework. First, the evolution of lattice templates using SAT-guided search methods could lead to many sparser and more informative dCNN configurations, enhancing both efficiency and interpretability. Second, deploying Inductive Logic Programming (ILP) directly on-device, for instance, via a lightweight RISC-V co-processor, would enable the system to update its rule base intra-day, a critical advantage in volatile economic environments where dynamics can shift rapidly. In addition, incorporating cross-asset logic clauses such as those linking foreign exchange fluctuations to fuel prices and downstream food inflation could improve the stability and responsiveness of national inflation models. Finally, deeper integration with regulatory technology platforms (reg-tech) would allow the mined rulebooks to be automatically exported into real-time policy dashboards, complete with intuitive traffic-light indicators to signal compliance or risk levels for key macroeconomic indicators.

While the current dCNN-E(ASP) prototype already delivers low-power, rule-aware forecasts that outperform conventional models, it should be viewed as a launch pad, not an end state. Several engineering and research upgrades lie within reach, ranging from smarter lattice design and on-chip logic mining to cross-asset reasoning and seamless policy-dashboard plumbing. Each enhancement aims to deepen one of the stack's core virtues: energy frugality, real-time adaptability, economic interpretability, or institutional usability. The roadmap below sketches four concrete avenues that could elevate the prototype from a promising field tool to a nation-scale analytics fabric.

SAT-guided evolution of lattice templates:

The present system generates its ± 1 convolution templates randomly, a choice that already works surprisingly well but is likely far from optimal. A promising next step is to cast template selection as a satisfiability problem: for a candidate 3×3 stencil, encode constraints such as "max 4 non-zeros" and "maximises mutual information with target series" into a Boolean formula, then let a modern SAT solver (for example, Minisat or Glucose) search the space of ± 1 patterns. Early experiments on small climate datasets have shown that SAT-guided reservoirs can prune 60% of weights while preserving, or even improving, predictive skill. Porting that idea to Congolese indicators could halve memory, tighten latency, and ease eventual ASIC synthesis.

In-package ILP on a RISC-V co-processor:

Rule mining currently runs overnight on a laptop, but customs backlogs or rainfall deficits can evolve within hours. Embedding a lightweight Inductive Logic Programming engine, essentially a stripped-down Aleph, on a companion RISC-V core within the same FPGA would enable the device to refresh its ASP clauses several times per trading day. Recent work by the PULP-Dronet team demonstrates that a 32-bit RISC-V processor running at 100 MHz can execute Horn-clause searches within tens of milliseconds while drawing under 50 mW; the same silicon real estate is already available on mid-range Zynq boards used by local grid operators.

Cross-asset, cross-market rule synthesis:

Currently, each forecast is based on a fairly narrow set of indicators. Yet, shocks in one asset class ripple into another. Diesel shortages push up maize transport costs, maize spikes feed into the headline CPI, and the resulting wage pressures weaken the franc. Extending the ILP search space to triangular clauses would allow the lattice to internalise these domino effects and thereby stabilise medium-term inflation forecasts. The approach aligns with recent IMF working papers that advocate for "joined-up" food-fuel-FX monitoring in commodity-exporting countries.

Reg-tech hooks and policy-dashboard integration:

Finally, the human value of an auto-mined rulebook is realised only when decision-makers see it. We plan to export each nightly ASP file to a lightweight JSON schema that powers a traffic-light dashboard, where green indicates clauses currently satisfied, amber indicates rules at risk, and red indicates active violations. Ghana's GRA has already piloted similar

visual pipelines for customs risk flags, as well as Kenya's CBK, for early-warning foreign exchange triggers. Integrating our rulebook into the Banque Centrale du Congo's existing Tableau dashboards would enable analysts to view model forecasts and their logical justifications side by side, thereby closing the loop between data science and actionable policy.

10 Conclusion Toward a Resilient, Low-Power Macroeconomic "Nerve Centre" for the DRC

A binary dCNN joined to an auto-mined ASP rulebook neutralises the four thorniest obstacles in fragile-data econometrics: speed, power, drift, and opacity. dCNN-E(ASP) runs on a pocket-sized computer, retrains overnight, and speaks to analysts in clear, auditable clauses. Early evidence from real Congolese data suggests double-digit accuracy gains and zero rule violations. Most importantly, the entire workflow fits in a tutorial notebook ready for replication by local universities, NGOs, and the Banque Centrale du Congo. We invite the community to test, critique, and localize the method, as better forecasts directly translate into more stable prices, steadier power, and, ultimately, improved livelihoods across the DRC. Over the past eighteen months, spanning from the first desktop prototype in early 2024 to the currently under-development trials scheduled for late 2025 or 2026, the dCNN + ASP pipeline is maturing from concept to field-tested reality. Pilot roll-outs emulating (internal) the Katanga grid-dispatch office and the Kasumbalesa one-stop border post now run on Raspberry Pi 4 boards powered by 50-W solar kits, logging zero rule violations and an aggregate energy cost of less than 3 kWh, equivalent to less than a single day of GPU training for an LSTM.

Three outcomes stand out:

(i) **Operational continuity:** The binary lattice continued to publish forecasts during 17 documented grid outages, validating the importance of "edge□first" architectures advocated by Satyanarayanan's seminal edge-computing manifesto, [16].

(ii) **Actionable transparency:** In March and May 2025, the Banque Centrale du Congo could have cited clause 11, linking FX spikes and rainfall deficits to food inflation in the Monetary Policy Committee minutes, thereby answering the IMF's 2024 Article IV call for glass-box analytics,[35].

(iii) **Data-gap immunity:** During a five-week hiatus in provincial-price uploads, the model's CPI error remained within ± 1 pp, whereas the bank's VAR(4) drifted by over 3 pp, echoing recent research on cellular-automata reservoirs and missing data, [53].

What Still Needs Work – Key Technical Challenges and Opportunities Ahead

While the current system performs reliably across diverse use cases, several technical and operational limitations remain that must be addressed to ensure long-term scalability and real-time usability. These include enhancing the model's expressiveness, improving the freshness of learned rules, and integrating outputs more directly into the workflow of analysts and decision-makers. The following outlines three key areas where targeted improvements could substantially raise the pipeline's fidelity, flexibility, and practical impact.

1. **Limited template expressiveness:**
   The current binary lattice uses fixed 3×3 templates composed of ± 1 weights to model system dynamics. While this setup works well for many economic indicators, it lacks the complexity needed to capture more nuanced, nonlinear phenomena such as the hysteresis effects in sediment buildup and turbine inefficiencies after rainfall peaks in hydropower forecasts. In its current form, the model may overlook subtle but important temporal interactions. A promising solution lies in SAT-guided template optimization, where the best-performing template is not chosen at random but discovered through a constraint-based satisfiability search. By encoding sparsity and performance constraints into a Boolean logic formula, solvers can identify lattice kernels that are both compact and information-rich. This approach was recently demonstrated by, [7], showing how CA templates can be tuned to maintain fidelity while reducing computational noise.

2. **Rule staleness and dynamic updating:**
   As with any data-driven rule engine, there is a risk that ASP clauses become outdated when the underlying policy, market, or environmental context shifts abruptly. For instance, a sudden change in mining royalties or a new foreign exchange policy can instantly render a previously valid rule invalid. To maintain relevance and accuracy, the system should ideally update its rulebook multiple times per day. Embedding a lightweight inductive logic programming (ILP) core running on an onboard RISC-V co-processor (e.g., in the style of the

PULP platform) would enable autonomous, intra-day rule refreshing at an ultra-low energy cost ($<$50 mW). This could ensure that local edge devices remain logically current without human intervention, even in highly dynamic environments.

3. **Lack of live integration with policymaker dashboards:** Currently, the system's forecasts and rule traces are shared as static PDF summaries. While informative, this mode of communication is too slow and disconnected for real-time policy guidance. For the system to be truly useful to ministries, central banks, and local planning offices, it must be visually and operationally integrated into their existing decision platforms. A natural next step is to convert the nightly ASP rule traces into structured JSON outputs, which can then be rendered as interactive traffic-light indicators (green/yellow/red) inside tools like Tableau or Power BI. Such a dashboard would allow decision-makers to monitor in real-time which economic rules are being respected, violated, or approaching risk thresholds, transforming a backend AI model into a fully operational early-warning interface.

Together, these enhancements would not only expand the model's capacity for learning and adaptation but also improve its institutional relevance, transparency, and responsiveness.

Why the Infrastructure Matters

With national electrification still averaging just around 20% and dropping well below that in several rural and eastern provinces, it is neither realistic nor sustainable to rely solely on centralised infrastructure for real-time economic forecasting and policy analytics. In such an environment, low-cost, low-power edge devices are not a replacement for regional data centres but a strategic complement. Efforts like the IFC-backed deployment of Raxio mini-datacentres, [54] are crucial for building national analytical capacity. However, during periods of limited grid power, internet outages, or infrastructural downtime, these central systems may temporarily go offline or become unreachable from remote districts. In contrast, edge devices such as solar-powered Raspberry Pi units or embedded RISC-V boards can continue to operate autonomously, executing core forecasting and alerting functions using local data. This dual-tier setup ensures resilience: when the backbone infrastructure falters due to power cuts or connectivity issues, village-level or district-level edge nodes keep the analytics pipeline alive. These nodes can maintain updated forecasts, trigger rule-based warnings (e.g., for inflation or energy shortages), and support local decisions in real-time. Once bandwidth is restored, the nodes can then synchronize lightweight artifacts such as updated rulebooks or model deltas with the central system, ensuring full reintegration without the need to resend bulky datasets. Such an architecture mirrors best practices from global deployments in remote healthcare, agricultural monitoring, and climate-risk modeling, where local computation and central aggregation work in tandem to deliver robustness, speed, and interpretability. For the DRC, this hybrid infrastructure offers a practical path toward distributed, inclusive, and reliable macroeconomic intelligence.

Call to the Community

We invite university labs of DR-Congo and other African countries, NGOs, and government data offices to contribute to forking one open repository on GitHub, plug in local datasets, and publish replication logs. A dedicated Slack channel ("DRC-Edge-Econ") under the African Data-Science Network could be established and presented at a future major IEEE conference on the African continent, possibly in a special session, such as one on Low-Power AI for Emerging Economies. Such an open repository shall explicitly welcome extensions and benchmarks. Suppose policymakers, technologists, and researchers converge on this neuro-symbolic scaffold. In that case, the DRC can graduate from patchy spreadsheets to a distributed, transparent, and energy-efficient early-warning grid, helping to stabilize prices, lights, and livelihoods well beyond 2025.

References

[1] World Bank, *Access to electricity (% of population) [indicator eg.elc.accs.zs; democratic republic of the congo]*, https://data.worldbank.org/indicator/EG.ELC.ACCS.ZS?locations=CD, Access Date: 19/05/2025.

[2] International Monetary Fund, "Democratic republic of the congo: 2024 article iv consultation," International Monetary Fund, IMF Country Report 24/95, Apr. 2024, Access Date: 15/05/2025. DOI: 10.5089/9798400283185.002.

[3] Reuters, *Congo halts mining expansions in east due to power shortage*, https://www.reuters.com/article/congo-democratic-mining/update-1-congo-halts-mining-expansions-in-east-due-to-power-shortage-

idUKL6N0M32L420140306/, Access Date: 16/05/2025.

[4] Sepp Hochreiter and Jürgen Schmidhuber, "Long short-term memory," *Neural computation*, vol. 9, no. 8, pp. 1735–1780, 1997. DOI: 10.1162/neco.1997.9.8.1735.

[5] Vaswani Ashish, "Attention is all you need," *Advances in neural information processing systems*, vol. 30, p. I, 2017. arXiv: 1706.03762 [cs.CL].

[6] Herbert Jaeger, *The "echo state" approach to analysing and training recurrent neural networks-with an erratum note*, https://www.ai.rug.nl/minds/uploads/EchoStatesTechRep.pdf, Access Date: 22/05/2025.

[7] Franco Bagnoli, Sara Dridi, and Nazim Fates, "Regional controllability of cellular automata as a sat problem," *arXiv preprint arXiv:2504.03691*, 2025. DOI: 10.48550/arXiv.2504.03691.

[8] Herbert Jaeger, *Short term memory in echo state networks*, https://www.ai.rug.nl/minds/uploads/STMEchoStatesTechRep.pdf, Access Date: 22/05/2025, 2001.

[9] Ben Radley, "Green imperialism, sovereignty, and the quest for national development in the congo," *Review of African Political Economy*, vol. 50, no. 177-178, pp. 322–339, 2023. DOI: 10.1080/03056244.2023.2277616.

[10] Stephen Muggleton, "Inductive logic programming," *New generation computing*, vol. 8, no. 4, pp. 295–318, 1991. DOI: 10.1007/BF03037089.

[11] Andrew Cropper, Rolf Morel, and Stephen Muggleton, "Learning higher-order logic programs," *Machine Learning*, vol. 109, no. 7, pp. 1289–1322, 2020. DOI: 10.48550/arXiv.1907.10953.

[12] Jiawei Han, Jian Pei, and Yiwen Yin, "Mining frequent patterns without candidate generation," *ACM sigmod record*, vol. 29, no. 2, pp. 1–12, 2000. DOI: 10.1145/335191.335372.

[13] Mohammed J Zaki, "Spade: An efficient algorithm for mining frequent sequences," *Machine learning*, vol. 42, pp. 31–60, 2001. DOI: 10.1023/A:1007652502315.

[14] Martin Gebser, Roland Kaminski, Benjamin Kaufmann, and Torsten Schaub, *Answer set solving in practice*. Springer Nature, 2022, ISBN: 978-3-031-01561-8. DOI: 10.1007/978-3-031-01561-8.

[15] Song Han, Han Cai, Ligeng Zhu, Ji Lin, Kuan Wang, Zhijian Liu, and Yujun Lin, "Design automation for efficient deep learning computing," 2019. DOI: 10.48550/arXiv.1904.10616.

[16] Mahadev Satyanarayanan, "The emergence of edge computing," *Computer*, vol. 50, no. 1, pp. 30–39, 2017. DOI: 10.1109/MC.2017.9.

[17] Artur S d'Avila Garcez, Krysia B Broda, Dov M Gabbay, Artur S d'Avila Garcez, Krysia B Broda, and Dov M Gabbay, "Knowledge extraction from trained networks," *Neural-Symbolic Learning Systems: Foundations and Applications*, pp. 113–158, 2002. DOI: 10.1007/978-1-4471-0211-3_5.

[18] Jonathan Lambert, Rosemary Monahan, and Kevin Casey, "Power consumption profiling of a lightweight development board: Sensing with the ina219 and teensy 4.0 microcontroller," *Electronics*, vol. 10, no. 7, 2021, ISSN: 2079-9292. DOI: 10.3390/electronics10070775.

[19] World Bank, *Governance and the digital economy in africa*, https://www.worldbank.org/en/topic/digital/brief/digital-for-governance-in-africa, Access Date: 21/05/2025.

[20] Andreas Holzinger, Georg Langs, Helmut Denk, Kurt Zatloukal, and Heimo Müller, "Causability and explainability of artificial intelligence in medicine," *Wiley Interdisciplinary Reviews: Data Mining and Knowledge Discovery*, vol. 9, no. 4, e1312, 2019. DOI: 10.1002/widm.1312.

[21] Aaron Ross, *Corrected-glencore invests in congo hydropower as solution to blackouts*, https://www.reuters.com/article/markets/commodities/corrected-glencore-invests-in-congo-hydropower-as-solution-to-blackouts-idUSL6N0TB481, Access Date: 16/06/2025, Nov. 2014.

[22] Gouhei Tanaka, Toshiyuki Yamane, Jean Benoit Héroux, Ryosho Nakane, Naoki Kanazawa, Seiji Takeda, Hidetoshi Numata, Daiju Nakano, and Akira Hirose, "Recent advances in physical reservoir computing: A review," *Neural Networks*, vol. 115, pp. 100–123, 2019. DOI: 10.1016/j.neunet.2019.03.005.

[23] Daniel Brunner, Miguel C Soriano, Claudio R Mirasso, and Ingo Fischer, "Parallel photonic information processing at gigabyte per second data rates using transient states," *Nature communications*, vol. 4, no. 1, p. 1364, 2013. DOI: 10.1038/ncomms2368.

[24] Bhavin J Shastri, Alexander N Tait, Thomas Ferreira de Lima, Wolfram HP Pernice, Harish Bhaskaran, C David Wright, and Paul R Prucnal, "Photonics for artificial intelligence and neuromorphic computing," *Nature Photonics*, vol. 15, no. 2, pp. 102–114, 2021. DOI: 10.1038/s41566-020-00754-y.

[25] Tobias Geibinger, "Explainable answer-set programming," 2023. DOI: 10 . 48550 / arXiv.2308.15901.

[26] João Carlos Alves Barata and Mahir Saleh Hussein, "The moore–penrose pseudoinverse: A tutorial review of the theory," *Brazilian Journal of Physics*, vol. 42, pp. 146–165, 2012. DOI: 10.1007/s13538-011-0052-z.

[27] *The moore–penrose pseudoinverse (math 33a)*, https : / / www . math . ucla . edu / ~laub / 33a.2.12s/mppseudoinverse.pdf, Course notes, Access Date: 05/06/2025, 2012.

[28] A Srinivasan, *Aleph: A learning engine for proposing hypotheses*, https : / / www . cs . ox . ac . uk / activities / programinduction / Aleph / aleph . html, Access Date: 22/05/2025.

[29] Firas Ahmmed Mohammed and Moamen Abbas Mousa, "Applying diebold–mariano test," *Theory and Applications of Time Series Analysis: Selected Contributions from ITISE 2019*, p. 443, 2020. DOI: 10.1007/978-3-030-56219-9_29.

[30] Yan Carriere-Swallow, Melih Firat, Davide Furceri, and Daniel Jimenez, "State-dependent exchange rate pass-through," International Monetary Fund, Washington, DC, IMF Working Paper 2023/086, Apr. 2023, Stock No. WPIEA2023086. DOI: 10 . 5089 / 9798400240461 . 001. Accessed: Jun. 10, 2025.

[31] Laurent Kemoe, Moustapha Mbohou, Hamza Mighri, Mr Saad N Quayyum, and Saad Quayyum, *Effect of Exchange Rate Movements on Inflation in Sub-Saharan Africa*. International Monetary Fund, 2024, Accessed: 2025-05-20, ISBN: 9798400269264. DOI: 10.5089/9798400269264.001.

[32] Yan Carrière-Swallow, Melih Firat, Davide Furceri, and Daniel Jiménez, *State-dependent exchange rate pass-through*, https : / / cepr . org / voxeu / columns / state - dependent - exchange - rate - pass - through, Access Date: 10/06/2025, 2024.

[33] Badolo Felix and Kinda Somlanare Romuald, *Rainfall shocks, food prices vulnerability and food security: Evidence for sub-saharan african countries*, https : / / www . afdb . org / sites / default / files / documents / publications / aec _ 2012_ - _rainfall _ shocks _ food _ prices _ vulnerability _ and_food_security-evidence_for_sub-saharan_african_countries.pdf, Access Date: 15/06/2025.

[34] Wenxin Wang, Isaac Adjei Mensah, Samuel Atingabili, and Akoto Yaw Omari-Sasu, "Climate change as a game changer: Rethinking africa's food security-health outcome nexus through a multi-sectoral lens," *Scientific Reports*, vol. 15, no. 1, pp. 1–25, 2025, Access Date: 08/06/2025. DOI: 10 . 1038 / s41598 - 025 - 99276-2.

[35] International Monetary Fund. African Dept., *Democratic Republic of the Congo*, Access Date: 12/06/2025. DOI: 10 . 5089 / 9798400283185.002.

[36] Ayobami Solomon Oyewo, Javier Farfan, Pasi Peltoniemi, and Christian Breyer, "Repercussion of large scale hydro dam deployment: The case of congo grand inga hydro project," *Energies*, vol. 11, no. 4, p. 972, 2018. DOI: 10.3390/en11040972.

[37] Omer Kawende Kalonda, Ivon Ndala Tshiwis, Boniface Mumbimb Atalatala, and Clement N'Zau Umba-Di-Mbudi, "Evolution of sedimentation in a headrace canal for hydroelectric production case of the shongo basin of the inga complex from february 2020 to may 2021 (kongo central province/dr congo)," *Journal of Geoscience and Environment Protection*, vol. 11, no. 5, pp. 404–426, 2023. DOI: 10 . 4236 / gep . 2023.115024.

[38] World Bank, *Increasing access to electricity in the democratic republic of congo: opportunities and challenges*, https : / / documents1.worldbank.org/curated/en/743721586836810203 / pdf / Increasing - Access - to - Electricity - in - the - Democratic - Republic - of - Congo - Opportunities - and - Challenges . pdf,

Access Date: 07/06/2025, Washington, DC, 2020.

[39] Laurent Kemoe, Moustapha Mbohou, Hamza Mighri, and Saad N Quayyum, *Front matter*, https://www.elibrary.imf.org/downloadpdf/view/journals/001/2024/059/article-A000-en.pdf, Access Date: 18/06/2025.

[40] World Bank Group, Climate Change Knowledge Portal, *Historical climate data: Congo, dem. rep.* https://climateknowledgeportal.worldbank.org/country/congo-dem-rep/climate-data-historical, Access Date: 11/06/2025.

[41] Sara Karam, Baba-Serges Zango, Ousmane Seidou, Duminda Perera, Nidhi Nagabhatla, and Raphael M. Tshimanga, "Impacts of climate change on hydrological regimes in the congo river basin," *Sustainability*, vol. 15, no. 7, 2023, ISSN: 2071-1050. DOI: 10.3390/su15076066.

[42] Douglas M Addison and Benjamin Stewart, *Nighttime lights revisited: The use of nighttime lights data as a proxy for economic variables*, https://openknowledge.worldbank.org/entities/publication/aadf01a6-ef30-5863-94d0-b687409de1bd, Access Date: 12/06/2025.

[43] Mildred Chama, *Zambia and drc resolve kasumbalesa impasse*, https://www.zra.org.zm/zambia-and-drc-resolve-kasumbalesa-impasse/, Access Date: 02/06/2025, Zambia Revenue Authority, Jun. 2022.

[44] Eugene Goddard, *Border bedlam returns to kasumbalesa*, https://www.freightnews.co.za/article/border-bedlam-returns-kasumbalesa, Access Date: 03/06/2025, Freight News.

[45] Robert CM Beyer, Yingyao Hu, and Jiaxiong Yao, *Measuring quarterly economic growth from outer space*. International Monetary Fund, 2022, ISBN: 9798400211553. DOI: 10.5089/9798400211553.001.

[46] International Monetary Fund, *Democratic republic of the congo: 2019 article iv consultation–press release; staff report; and statement by the executive director for the democratic republic of the congo*, https://www.imf.org/en/Publications/CR/Issues/2019/09/04/Democratic-Republic-of-the-Congo-2019-Article-IV-Consultation-Press-Release-Staff-Report-and-48648, Access Date: 06/06/2025.

[47] Shengyu Liu, Jinghua Xiao, Zixiang Yan, and Jian Gao, "Noise resistance of next-generation reservoir computing: A comparative study with high-order correlation computation," *Nonlinear Dynamics*, vol. 111, no. 15, pp. 14 295–14 308, 2023. DOI: 10.1007/s11071-023-08592-7.

[48] Pietro Verzelli, Lorenzo Livi, and Cesare Alippi, "A characterization of the edge of criticality in binary echo state networks," in *2018 IEEE 28th International Workshop on Machine Learning for Signal Processing (MLSP)*, IEEE, 2018, pp. 1–6. DOI: 10.1109/MLSP.2018.8516959.

[49] Mark Law, Krysia Broda, and Alessandra Russo, "Search space expansion for efficient incremental inductive logic programming from streamed data.," in *IJCAI*, 2022, pp. 2697–2704. DOI: 10.24963/ijcai.2022/374.

[50] João Gama, Indrė Žliobaitė, Albert Bifet, Mykola Pechenizkiy, and Abdelhamid Bouchachia, "A survey on concept drift adaptation," *ACM computing surveys (CSUR)*, vol. 46, no. 4, pp. 1–37, 2014. DOI: 10.1145/2523813.

[51] MG Sarwar Murshed, Christopher Murphy, Daqing Hou, Nazar Khan, Ganesh Ananthanarayanan, and Faraz Hussain, "Machine learning at the network edge: A survey," *ACM Computing Surveys (CSUR)*, vol. 54, no. 8, pp. 1–37, 2021. DOI: 10.1145/3469029.

[52] Jonas Kantic, Fabian C Legl, Walter Stechele, and Jakob Hermann, "Relicada: Reservoir computing using linear cellular automata design algorithm," *Complex & Intelligent Systems*, vol. 10, no. 3, pp. 3593–3616, 2024. DOI: 10.1007/s40747-023-01330-x.

[53] SciML Documentation, *Reservoir computing using cellular automata*, https://docs.sciml.ai/ReservoirComputing/stable/, Access Date: 18/06/2025, May 2025.

[54] Raxio Group, *Raxio group: Data center services and solutions*, https://www.raxiogroup.com/, Access Date: 16/06/2025.

Contribution of Individual Authors to the Creation of a Scientific Article (Ghostwriting Policy)

The authors equally contributed to all aspects of this research from problem formulation to final results.

Sources of Funding for Research Presented in a Scientific Article or Scientific Article Itself

No funding was received for this study.

Conflicts of Interest

The authors declare no conflicts of interest.

APPENDIX

11 Code Listings

11.1 Data Preparation

[caption=Data preparation and panel construction., label=lst:data-prep] import pandas as pd
— Load and pivot to a panel — df = pd.read$_csv$(”$drc_econ_panel.csv$”, $parse_dates = [$”$date$”$])panel =$
$(df.pivot(index =$ ”$date$”$, columns =$ ”$indicator$”$, values =$ ”$value$”$).asfreq($”$W - MON$”$))$
— Linear interpolation for ≤ 3-week gaps; leave longer gaps as NaN — panel = panel.interpolate(”linear”, limit=3) panel = panel.fillna(0) reserve 0 as explicit ”missing” flag

11.2 Stage A: Binary dCNN Auto-Encoder

[caption=Binary lattice evolution with fixed templates., label=lst:evolve] import numpy as np from numba import njit, prange
Build random ±1 templates once rng =
np.random.default$_rng(2024)A = rng.choice([-1, 0, 1], size = (3, 3), p = [.3, .4, .3]).astype(np.int8)B =$
$rng.choice([-1, 0, 1], size = (3, 3), p = [.3, .4, .3]).astype(np.int8)$
@njit(parallel=True) def evolve(lattice, static, steps=6): H, W = lattice.shape for
$_inrange(steps) : new = lattice.copy()foriiinprange(1, H - 1) : forjinprange(1, W - 1) : feedback =$
$(A * lattice[i - 1 : i + 2, j - 1 : j + 2]).sum()driving = (B * static[i - 1 : i + 2, j - 1 :$
$j + 2]).sum()new[i, j] = 1if(feedback + driving) >= 0else - 1lattice = newreturnlattice$
[caption=Linear decoder via pseudoinverse., label=lst:decoder] Z: flattened latent codes, Y: flattened original windows W$_dec = Y@Z.T@np.linalg.pinv(Z@Z.T + 1e - 4 * np.eye(Z.shape[0]))$

11.3 Stage B: Automatic Rule Discovery

[caption=Rule mining and SAT-based pruning (high-level pseudocode)., label=lst:rule-mining] Input: discretised predicates P(t) for t=1..T, lags Δ in 1..8 Params: s$_fp = 0.05, conf = 0.60, s_spade =$
$0.03, max_gap = 4, max_len = 3Params(ILP) : max_body = 4, max_vars = 3, max_lag = 8$
1) Synchronous mining (FP-Growth) F = FPGrowth(transactions=P(t),
min$_support = s_fp)R_sync = S- > c@ : support(S) >= s_fpandPr(c_{t+}|S_t) >= conf$
2) Temporal mining (SPADE) Q = SPADE(sequences=P(t), min$_support = s_spade, max_gap =$
$max_gap, max_len = max_len)R_seq = pattern- > c@ : <= max_lagandconfidence >= conf$
3) Logical generalization (ILP) R0 = Aleph.induce(background$_knowledge, candidates =$
$R_sync□R_seq, max_body = max_body, max_vars = max_vars, max_lag = max_lag)$
4) SAT-based consistency pruning KB =
integrity$_constraints(mutual_exclusion, hard_domain_bounds)R = []forrinR0 : ifSAT(KB□R□r) ==$
$SATISFIABLE : R.append(r)keepconsistentrulereturncompile_to_ASP(R)rules.lp$
[caption=Discretization helper., label=lst:discretise] import numpy as np
def discretise(column): q1, q2 = np.percentile(column, [33, 66]) return np.where(column < q1, 0, np.where(column < q2, 1, 2))
[caption=Transactional export for FP-Growth., label=lst:transactions] with open(”transactions.csv”, ”w”) as f:
for t, row in enumerate(discretised$_panel) : fori, lvlinenumerate(row) : f.write(f”t, i_lvl”)$
[language=bash, caption=Example commands to run the mining toolchain., label=lst:run-miners] python
run$_fpgrowth.py - -min_support0.05 - -min_confidence0.60pythonspade.py - -min_support0.03 -$
$-max_gap4 - -max_len3pythonaleph_induce.plrules_bg.ploutput.lppythonsat_prune.pyoutput.lp > rules.lp$

11.4 Stage C: Rule-Aware Forecaster

[caption=Rule-aware ridge readout (high-level pseudocode)., label=lst:rule-aware-forecaster] H = collect$_latent_states(new_lattice_states)dxNy = target_vectorNGk =$
$build_penalty_matrices(rules, H)listof(dxN)selectors/weights$
A = H @ H.T b = H @ y for G in Gk: A += lambda$_*(G@G.T)$
beta = np.linalg.solve(A + 1e-6*np.eye(A.shape[0]), b) y$_hat = beta@H$