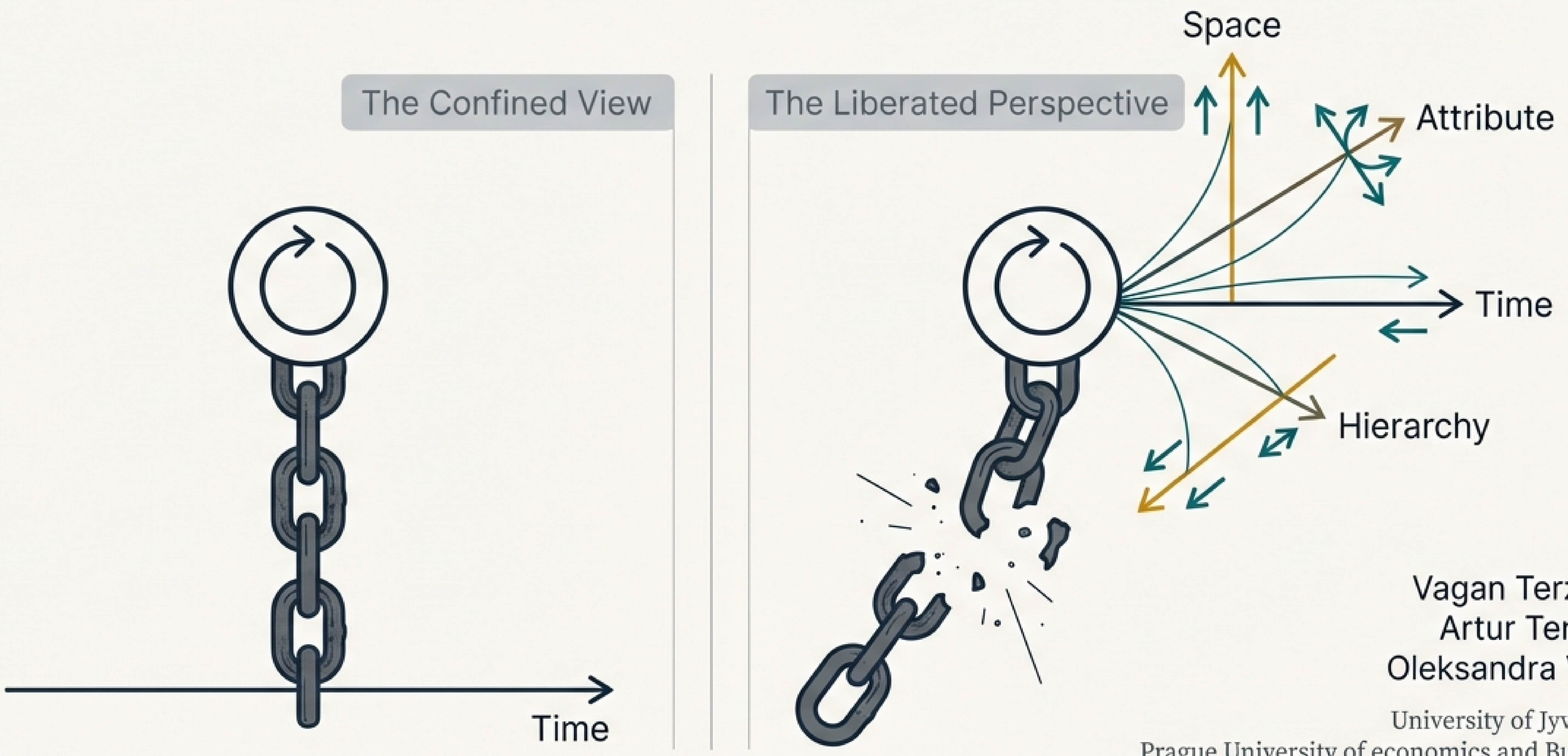
Rethinking Order

Recurrent Neural Networks Beyond the Confines of Time

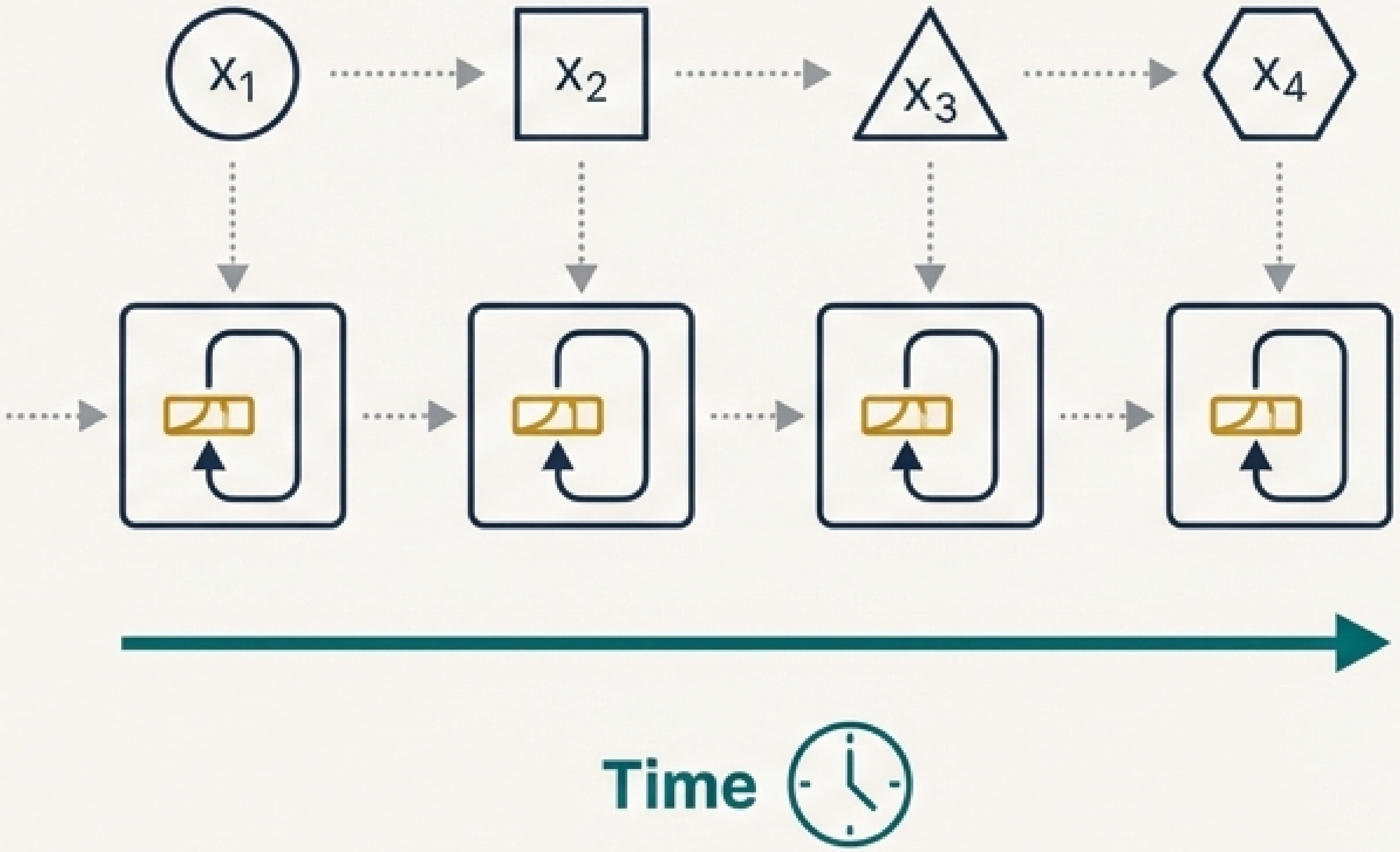


Vagan Terziyan
Artur Terziian
Oleksandra Vitko

University of Jyväskylä
Prague University of economics and Business
Kharkiv National University of Radio Electronics

For Decades, We've Equated Recurrence With Time

The success of RNNs in modeling language, speech, and time-series has deeply embedded the notion of *time* into our understanding of recurrent architectures. The sequence index is almost always interpreted as a temporal step.

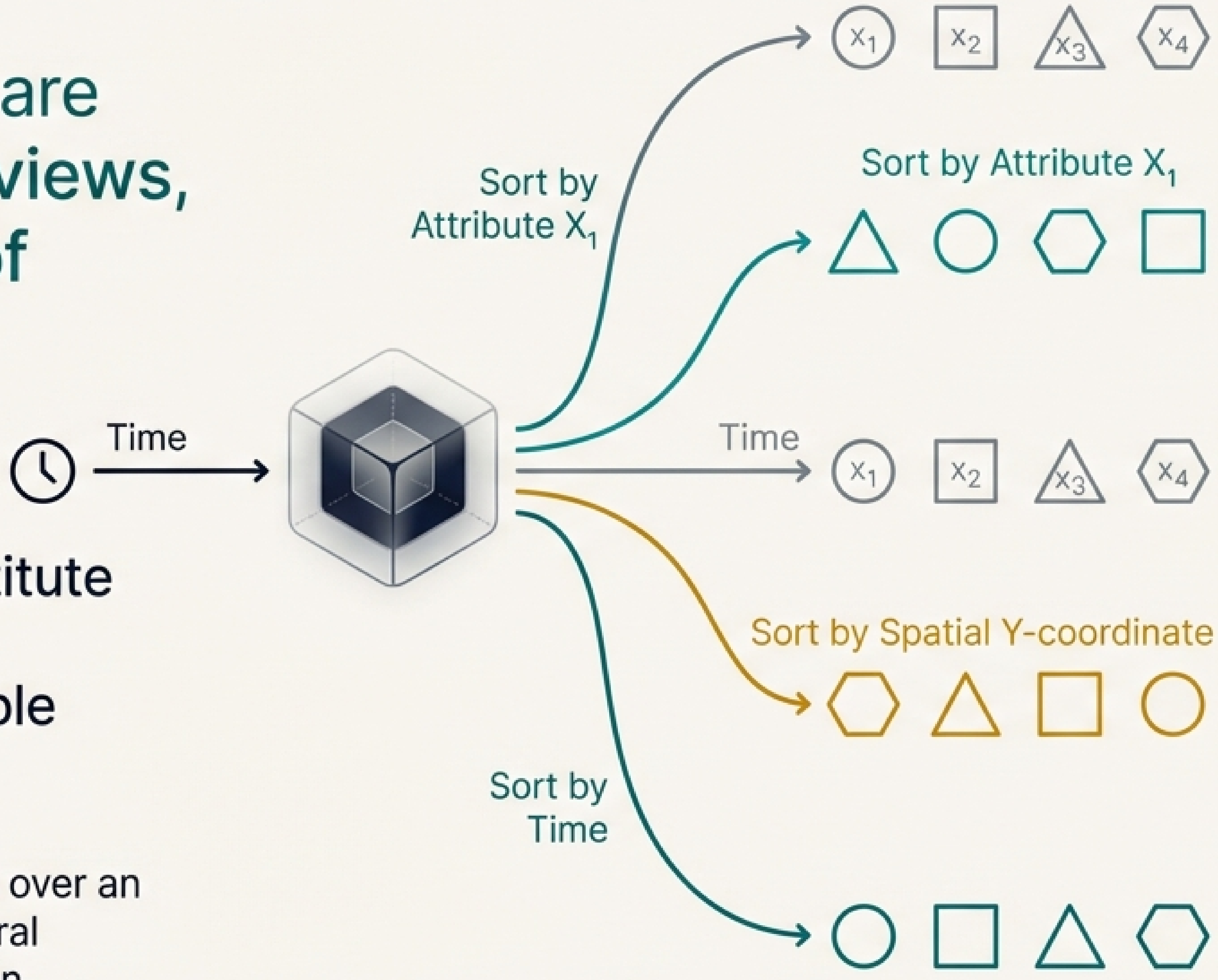


RNNs and their gated variants have long been a foundational tool for modeling sequential data... As a result, the notion of time... has become deeply embedded in both the conceptual understanding and practical use of recurrent architectures.

This close association between recurrence and time obscures a more fundamental property of RNNs: they are, at their core, models of *ordered data*, not of time per se.

The Core Insight: Order is a Design Variable, Not a Given

What happens if the same data are projected into multiple **ordered views**, each capturing a **distinct form of structural evolution?**



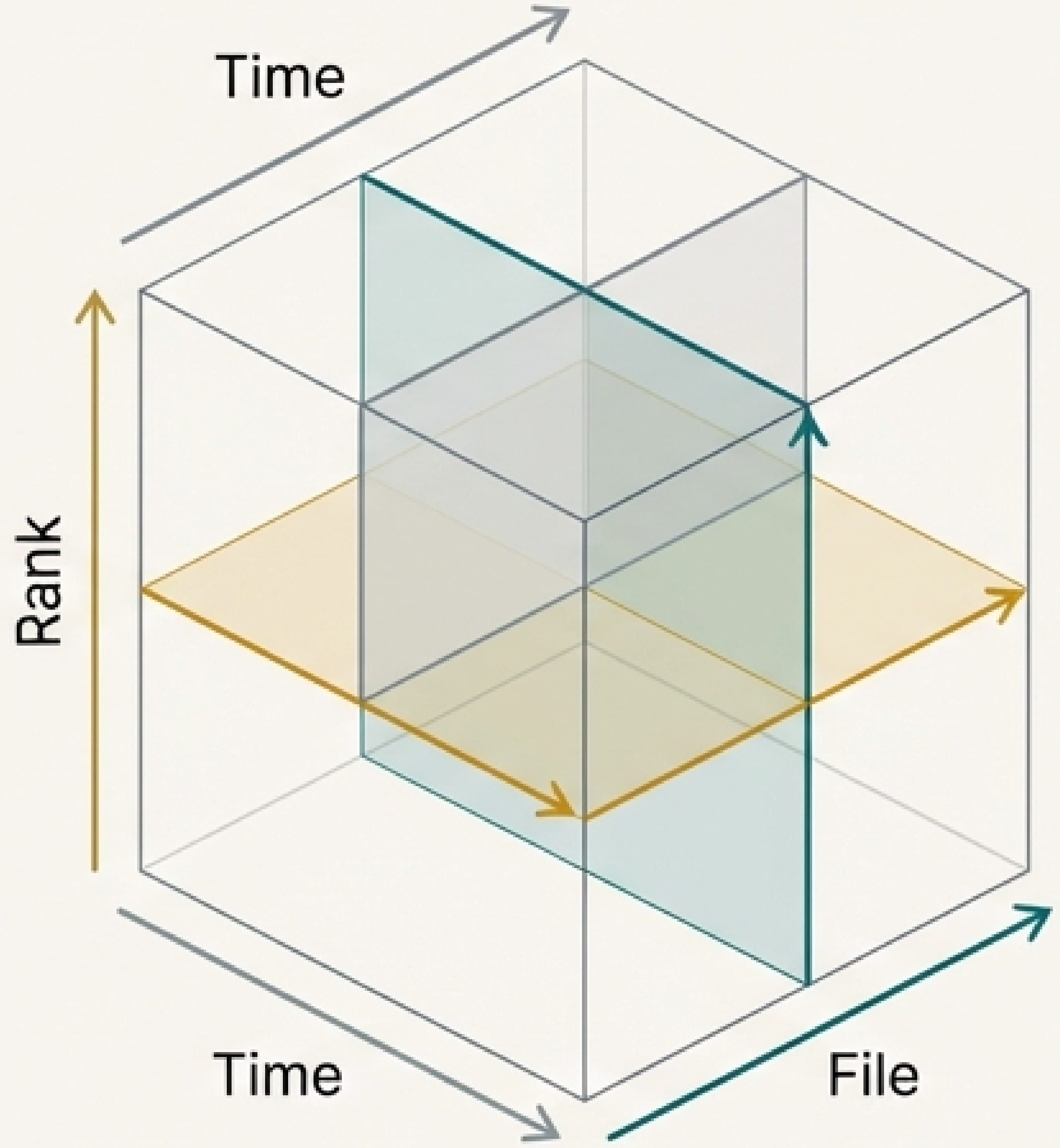
We argue that “ordered projections constitute a missing conceptual layer in sequential modeling.” Time becomes just one possible axis among many.

Formalism teaser: An RNN’s recurrence relation operates over an ordered index set. It does not intrinsically encode temporal duration, causality, or simultaneity. Any index permutation creates a new, valid sequence for an RNN to model.

A Chess Game Is More Than Just a Sequence of Moves

The Conventional View

A chess game is traditionally seen as a temporal sequence of board states. An RNN would process move 1, then move 2, etc. This is the 'Time Slice' view.



A Multi-Axis View

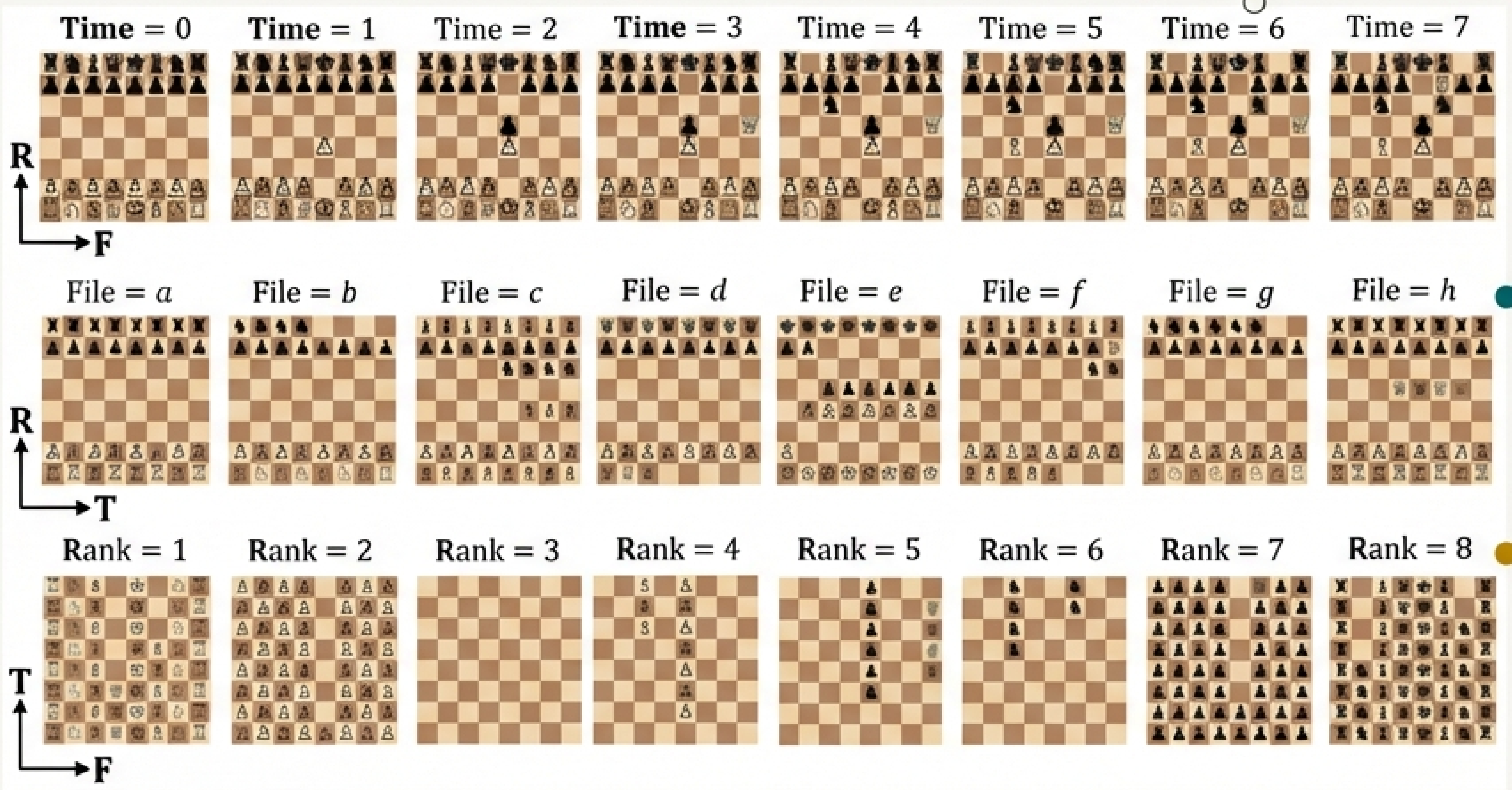
The same game, represented as an 8x8xT tensor, can be 'sliced' in other ways:

- **File Slices:** Sequences showing the evolution of a single column (file) over the entire game. This reveals patterns of vertical control and pawn structure.
- **Rank Slices:** Sequences showing the evolution of a single row (rank) over the game. This highlights horizontal control and defensive alignments.

These alternative slices encode 'orthogonal structural information' that is difficult to extract from a purely temporal analysis. They transform spatial regularities into ordered sequences.

Visualizing a Multi-Order World: Time, File, and Rank Slices

Showing the 'Scholar's Mate' game (1. e4 e5 2. Qh5 Nc6 3. Bc4 Nf6 4. Qxf7#) represented as a multi-axis tensor.



Conventional View: A Sequence of Board States

Captures the sequential dynamics and move-by-move progression.

Structural View 1: Evolution of Vertical Space

Reveals patterns in vertical control, like pawn chains and attacks along a file.

Structural View 2: Evolution of Horizontal Space

Highlights horizontal piece coordination and defensive alignments.

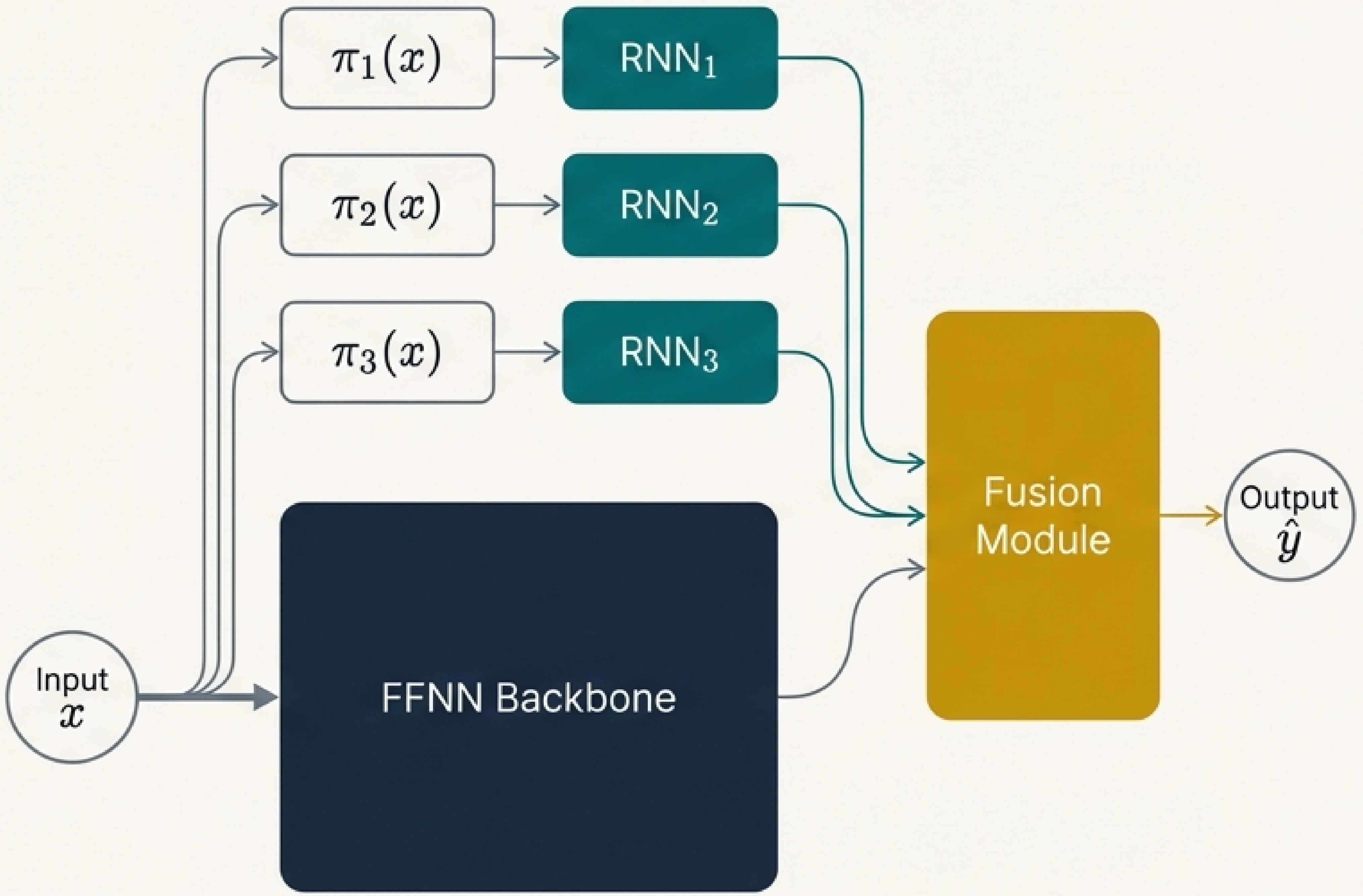
While non-temporal slices appear unusual to a human eye, they provide rich, structured sequences for a machine learning model to learn from.

SE-RNNs: An Architecture for a Multi-Order World

Key Concept: Structural Evolution RNNs (SE-RNNs) are designed to learn from multiple ordered projections simultaneously. The term “structural evolution” is used to emphasize that recurrence is applied to ordered transformations, not just temporal dynamics.

Architectural Blueprint (High-Level):

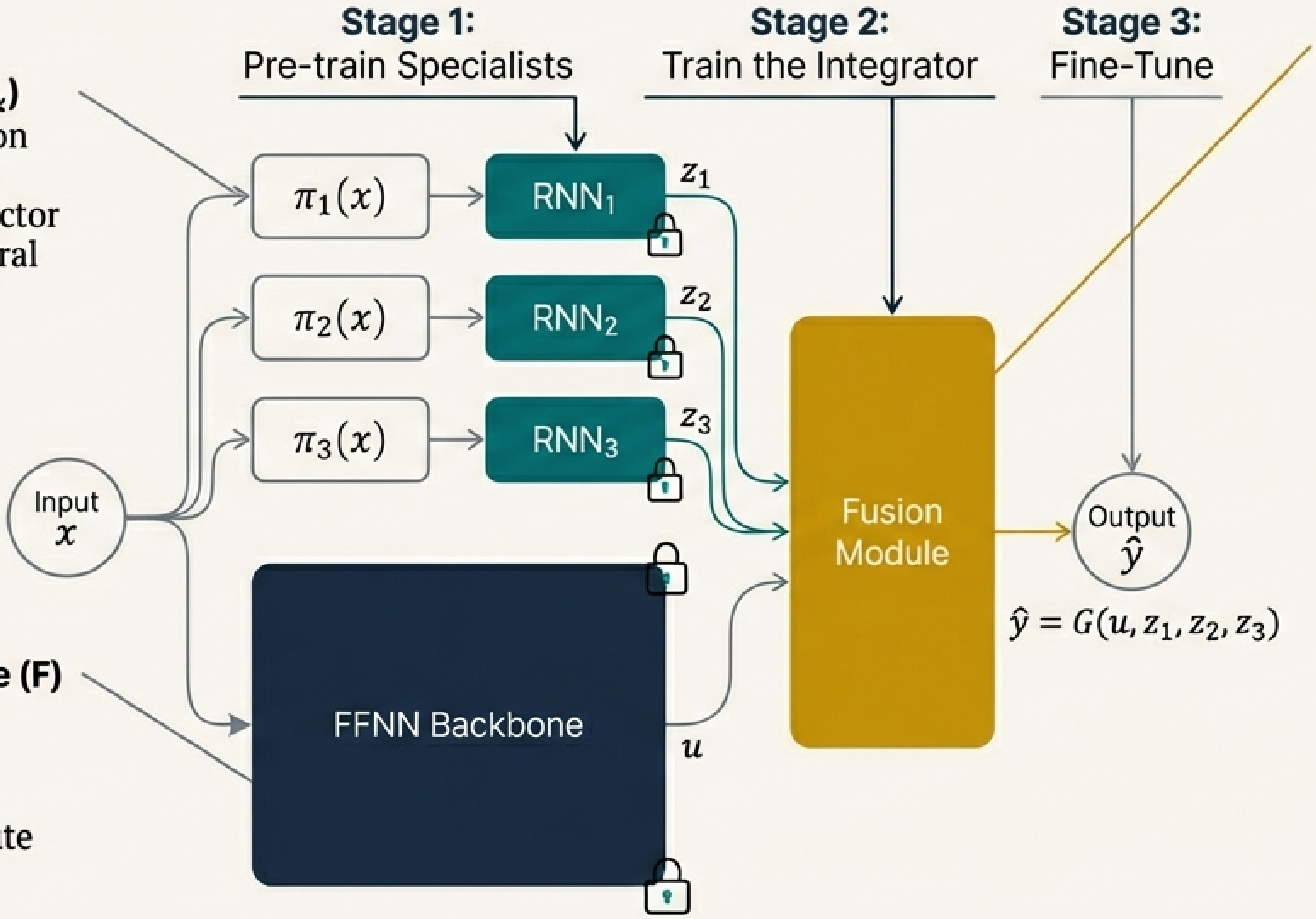
1. **FFNN Backbone**: A standard feedforward network acts as the primary decision-maker, learning from the original, non-sequential data.
2. **RNN 'Advisors'**: For each ordered projection (e.g., along Time, File, Rank), a separate, independent RNN learns the 'structural evolution' along that axis.
3. **Fusion Module**: A dedicated 'Integrator' module learns to combine the insights from the FFNN backbone and the various RNN advisors into a single, unified prediction.



The SE-RNN Blueprint: A Symphony of Specialists

Axis-Specific RNNs (R_k)
Each RNN is an expert on one projection $\pi_k(x)$. It produces a summary vector z_k encoding the structural evolution along its assigned axis.

Feedforward Backbone (F)
Provides a strong, non-sequential baseline prediction pathway by capturing global attribute interactions.
Output is $u = F(x)$.



Context-Aware Fusion (G)
The key integrator. It does *not* simply concatenate inputs. Instead, it uses mechanisms like gating or attention to modulate the influence of each each RNN advisor based on the global context provided by the FFNN backbone.

Staged Training Strategy

1. **Pre-train Specialists:** Train the FFNN backbone and each RNN advisor independently and in parallel.
2. **Train the Integrator:** Freeze the pre-trained specialists. Train only the fusion module to learn how to combine their outputs.
3. **Fine-Tuning (Optional):** Perform limited end-to-end fine-tuning with controlled learning rates.

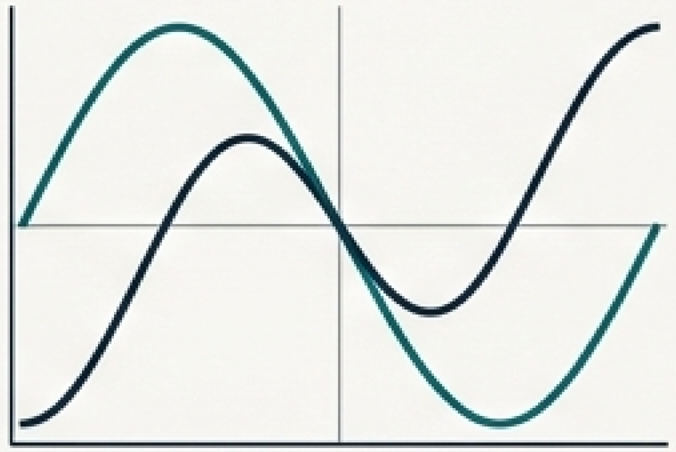
The Experimental Question: Does Multi-Axis Modeling Actually Help?

To conduct proof-of-concept experiments to validate that SE-RNNs can outperform a strong FFNN baseline, particularly when complex, hidden interdependencies exist in the data.

Dataset 1: Moderate Complexity

Smooth, trigonometric interactions designed to test the ability to capture subtle structural evolution.

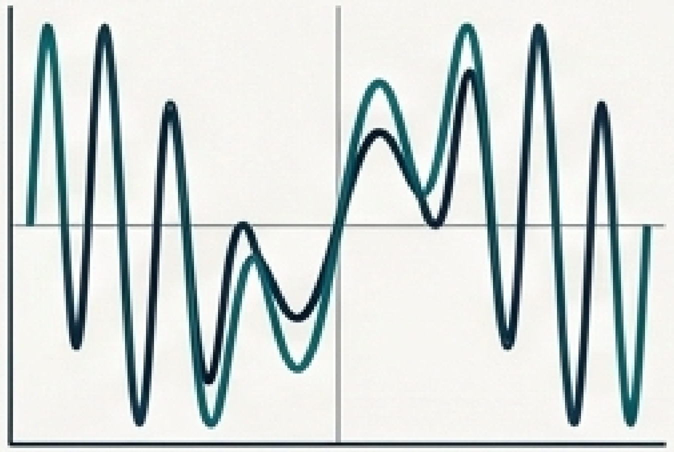
$$y = \sin(x_1 \cdot x_2) + \cos(x_2 \cdot x_3) + \tanh(x_1 - x_3) + \varepsilon$$



Dataset 2: High Complexity

Highly nonlinear, multiplicative, and oscillatory interactions designed to stress the model's capacity to extract non-obvious patterns.

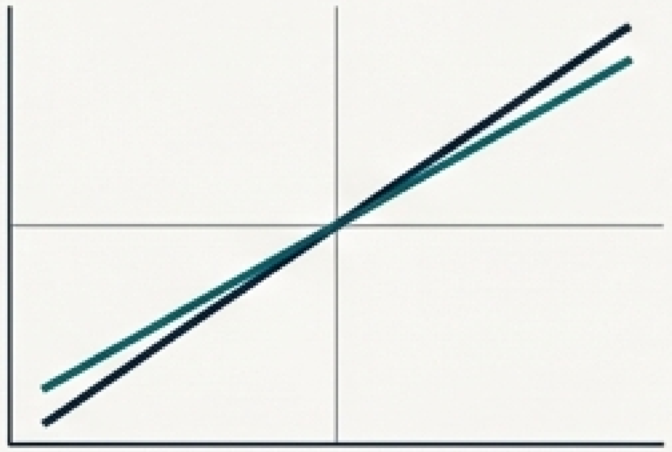
$$y = \cos(x_1) \cdot x_2 - \sin(x_2 \cdot x_3) + 0.1 \cdot \varepsilon$$



Dataset 3: Adversarial/Simple

A simple linear combination where an FFNN should be sufficient, testing the architecture's robustness against redundancy.

$$y = 1.5x_1 - 2.0x_2 + 0.5x_3 + \varepsilon$$



Finding 1: The Richer the Hidden Structure, the Greater the SE-RNN Advantage

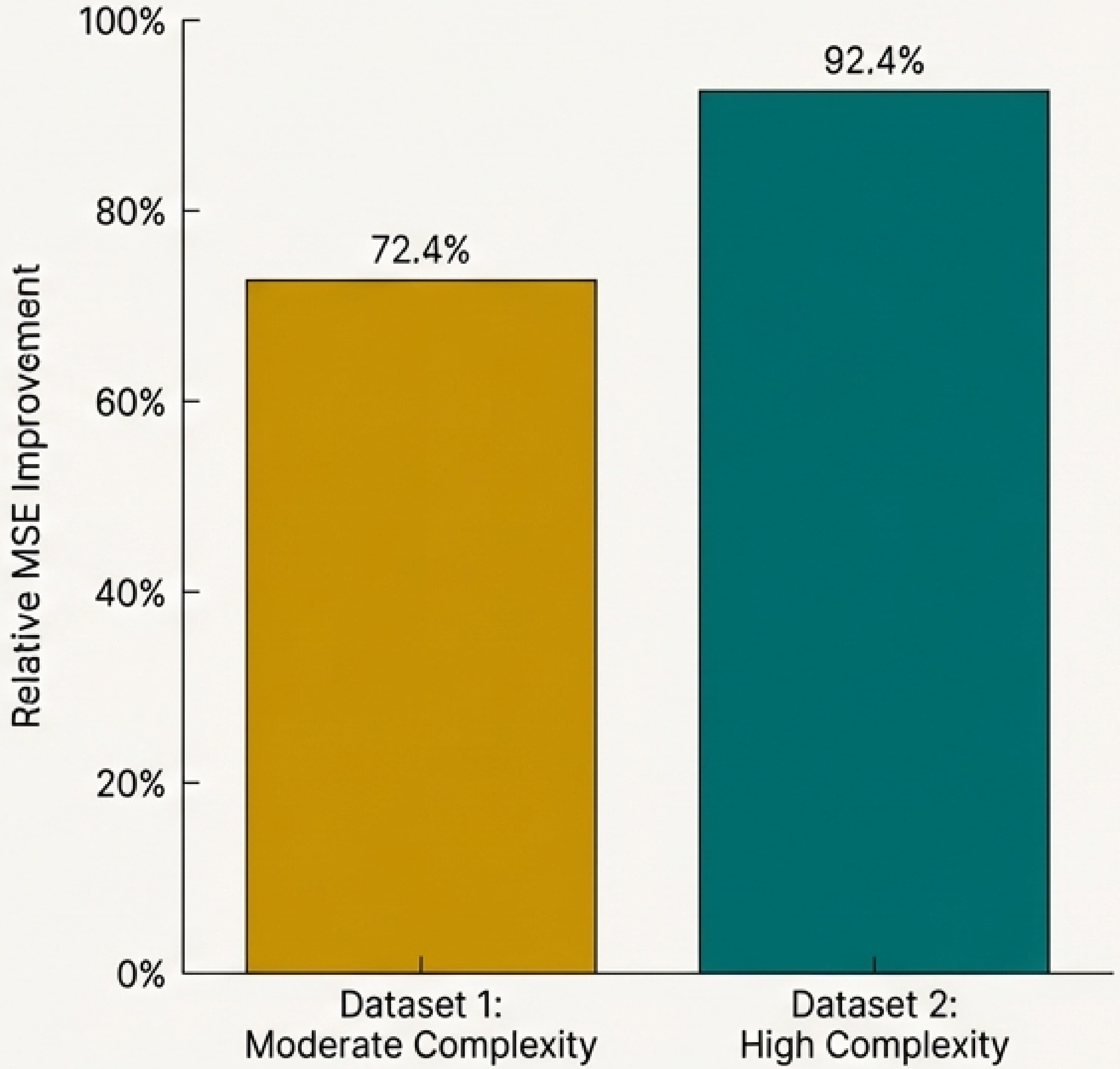
Across datasets with non-trivial interactions, SE-RNNs consistently and substantially outperform the FFNN baseline. The benefit grows with the complexity of the underlying data structure.

Distilled Results

On **Dataset 1 (Moderate)**, SE-RNNs achieved an improvement of up to **72.4%** over the baseline FFNN.

On **Dataset 2 (Complex)**, the advantage became even more pronounced, with an improvement of up to **92.4%.**

This trend strongly supports the hypothesis that SE-RNNs are particularly well-suited for datasets where structural evolution along multiple axes carries predictive information.



Finding 2: The Architecture is Graceful, Not Brittle

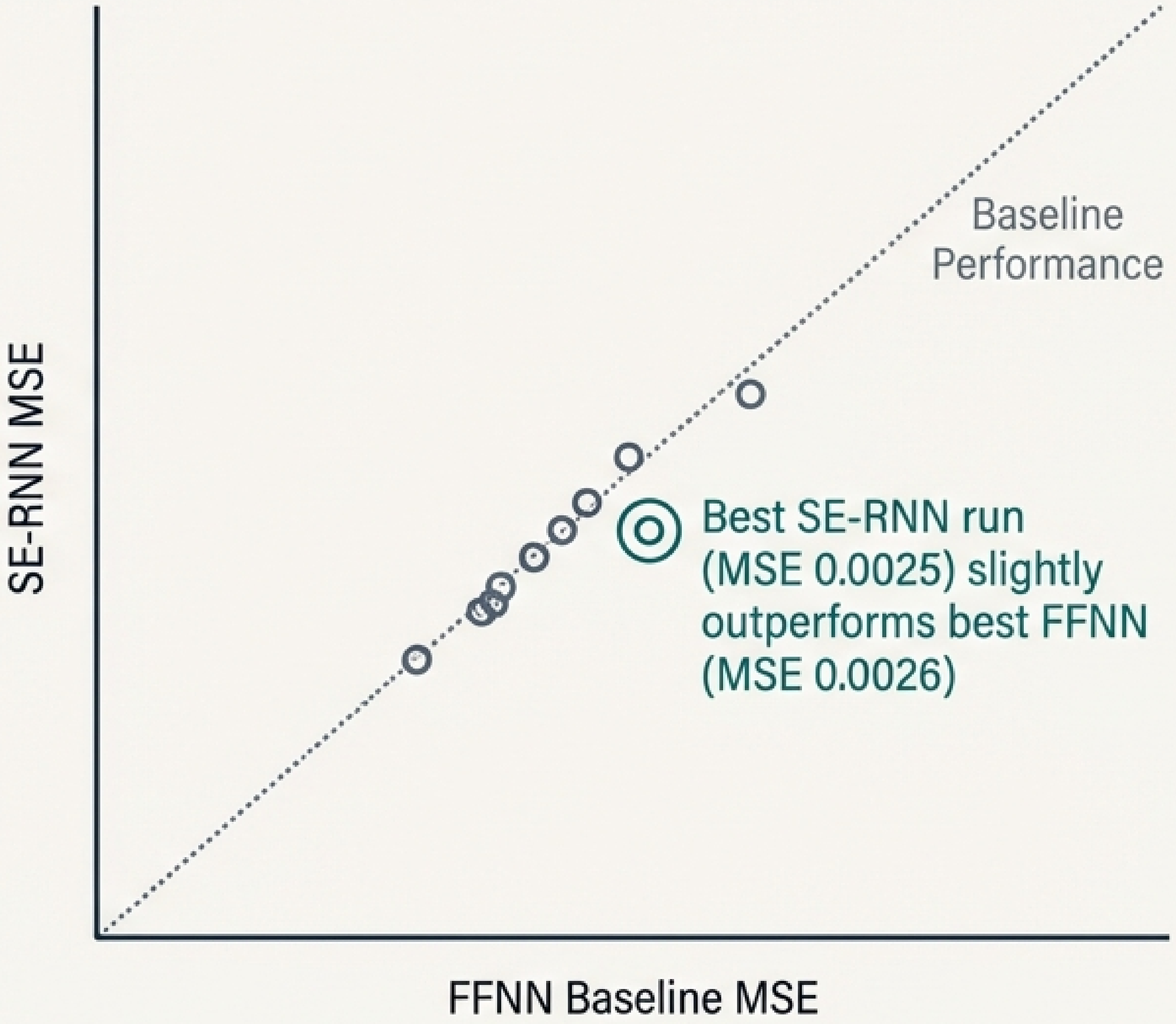
The "Adversarial" Test: On a simple linear dataset (Dataset 3), where an FFNN is already near-optimal, how does SE-RNN perform?

The SE-RNN architecture gracefully defaults to performance comparable to the FFNN baseline. It does not catastrophically fail.

When the FFNN is strong, SE-RNN performance is very close, sometimes with a slight degradation due to redundancy (e.g., -10.95% relative improvement in Run 8).

Even in this simple case, the best SE-RNN configuration (Run 6, MSE=0.0025) still slightly surpassed the best FFNN baseline (Run 2, MSE=0.0026).

Conclusion: SE-RNNs provide a safe, adaptive framework. They provide substantial improvement when rich hidden structure exists, and maintain near-baseline performance when that structure is trivial.



Efficiency by Design: Parallel Training and a Lightweight Integrator

The Challenge

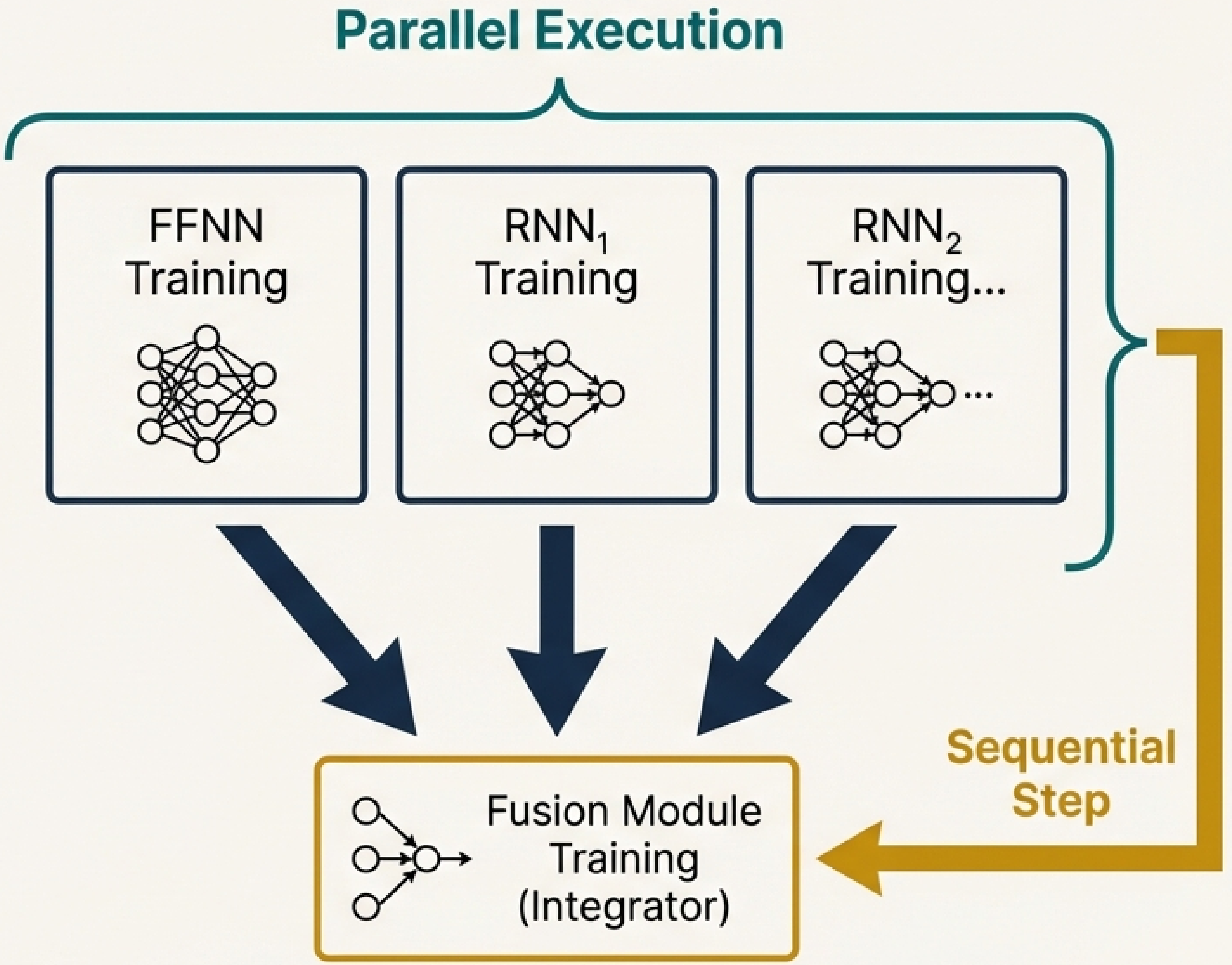
An architecture with multiple RNNs could seem computationally expensive.

The SE-RNN Solution

Modularity enables efficiency.

- **Parallel Pre-Training:** The FFNN backbone and ***all*** axis-specific RNNs can be trained simultaneously. The total time is determined by the longest single component's training, not the sum of all components.
- **Sequential Bottleneck is Small:** The only sequential step is training the fusion module, which integrates the frozen outputs. This module is typically small and requires far fewer resources.

Takeaway: The SE-RNN architecture allows exploitation of multi-axis structural information without incurring prohibitive time costs... structural enrichment is achieved with minimal additional computational burden.



Positioning SE-RNNs in the Landscape of Structural Learning

vs. Multi-Dimensional RNNs (MDRNNs)

MDRNNs entangle dimensions within a single, complex recurrent process on a grid.

SE-RNNs decouple projections, training independent 1D RNNs in parallel and integrating their outputs.

This allows for non-grid data and heterogeneous axes.

vs. Multi-View Learning

Multi-view models typically use externally defined, heterogeneous data sources (e.g., image + text).

SE-RNNs internally generate views by creating ordered projections of the *same* feature space.

It's structural re-interpretation, not data fusion.

vs. Transformers

Transformers replace recurrence with self-attention on a single sequence.

SE-RNNs are orthogonal: they multiply the number of axes along which ordered processing occurs, retaining the inductive bias of recurrence on each axis.

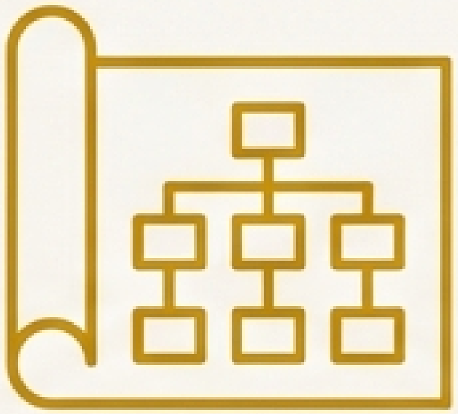
The Unique Niche

SE-RNNs introduce systematic recırrence over multiple feature-derived orderings, subsuming the FFNN as a core component, creating a flexible architecture whose benefits scale with the data's structural complexity.

A Three-Fold Contribution to Sequence Modeling

Conceptual Contribution

We provide a precise analysis that disentangles recurrence from time, reframing RNNs as models of **ordered structural evolution**. This introduces “order” as a first-class design variable.



Architectural Contribution

We propose the **SE-RNN architecture**, a modular and parallelizable framework combining an FFNN backbone with multiple axis-specific RNN “advisors” and a context-aware fusion module.

Experimental Contribution

We demonstrate through systematic experiments that SE-RNNs offer significant performance gains on data with complex hidden structures, while remaining robust and safe to deploy on simpler data.

The Future of Sequence Modeling is Multi-Ordered

The Paradigm Shift

By decoupling recurrence from time, we move from a single, privileged timeline to a richer, multi-perspective view of data.

This work contributes both a novel modeling paradigm and concrete empirical evidence for its relevance, inspiring future research at the intersection of structure, sequence, and learning.

Future Directions

- **Applications**: Exploring SE-RNNs on real-world spatial-temporal analytics, structured tabular data, and complex decision-making tasks.
- **Architecture**: Developing methods to automatically learn the most informative projections and designing more expressive fusion mechanisms.
- **Theory**: Further exploring the implications of treating order as a modeling choice for multi-view learning and modular neural systems.

Code and Datasets: ai.it.jyu.fi/experiments/SE-RNNs/