Probabilistic Circuits for Irregular Multivariate Time Series Forecasting

Christian Klötergens
ISMLL & VWFS DARC
University of Hildesheim
Hildesheim, Germany
kloetergens@ismll.de

Lars Schmidt-Thieme
ISMLL & VWFS DARC
University of Hildesheim
Hildesheim, Germany

Vijaya Krishna Yalavarthi
ISMLL & VWFS DARC
University of Hildesheim
Hildesheim, Germany

Abstract

Joint probabilistic modeling is essential for forecasting irregular multivariate time series (IMTS) to accurately quantify uncertainty. Existing approaches often struggle to balance model expressivity with consistent marginalization, frequently leading to unreliable or contradictory forecasts. To address this, we propose **CircuITS**, a novel architecture for probabilistic IMTS forecasting based on sum-product networks (SPNs). Our model is flexible in capturing intricate dependencies between time series channels while structurally guaranteeing valid joint distributions. Experiments on five real-world datasets demonstrate that CircuITS achieves superior joint and marginal density estimation compared to state-of-the-art baselines.

1 Introduction

Forecasting irregular multivariate time series (IMTS) means predicting the value of a variable (a *channel*) at a given time from sparse, irregularly sampled observations. Because real-world IMTS are generated by stochastic processes, point predictions are rarely sufficient: a useful forecast must quantify uncertainty and capture dependencies between channels, which requires predicting a *joint* distribution over a flexible number of query variables. A critical limitation of the state-of-the-art model in probabilistic IMTS forecasting (ProFITi [Yalavarthi et al., 2025a]) is the lack of marginalization consistency [Yalavarthi et al., 2025b]. This flaw leads to contradictory predictions. For example, a weather model might predict that snow is unlikely at $10°C$, yet revise that probability the moment we also ask about humidity. In a well-behaved model, adding or removing variables should not alter the predicted distribution over the remaining ones. When a model is inconsistent, its answers are artifacts of the question rather than the underlying distribution.

Recently MOSES [Yalavarthi et al., 2025b] was introduced to be a marginalization consistent, but expressive model for probabilistic IMTS forecasting. MOSES, however, lags behind ProFITi in estimating accurate conditional joint densities. While theoretically capable of approximating any distribution, MOSES struggles with the exponential scaling required to represent independent multi-modal processes. As shown in Figure 1, accurately modeling a simple bifurcation across four independent channels requires a mixture of $2^C = 16$ components to represent all joint modes (C representing the number of channels). MOSES with 10 components fails due to insufficient capacity; however, even with 16 components MOSES fails to capture the distinct branches despite having sufficient components in principle. The optimization landscape for such high-dimensional mixtures becomes difficult to navigate, leading to poor local minima that fail to represent the ground truth distribution. While one could trivially solve this specific example by manually enforcing independence between channels, real-world data contains a complex, unknown mix of dependencies that cannot be hard-coded upfront. Hence, there seems to be a trade-off between accurate modeling and marginalization consistency. This is not only a dilemma for practitioners but also unintuitive as marginalization consistency should introduce the correct inductive bias into the model.

40th Conference on Neural Information Processing Systems (NeurIPS 2026).

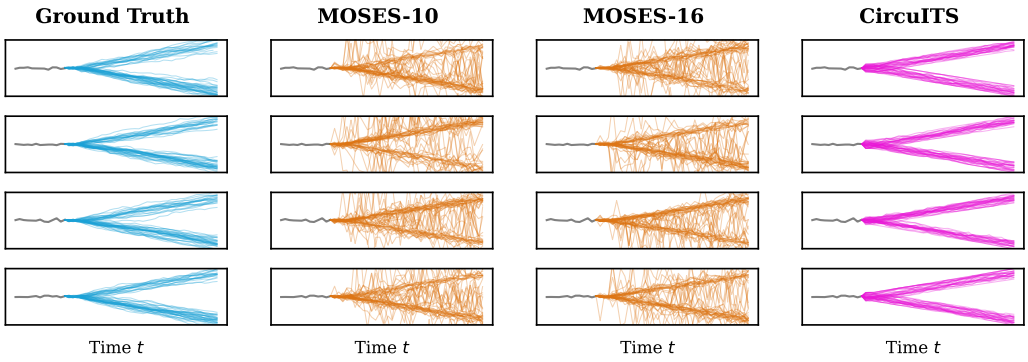

Figure 1: Demonstrating MOSES' inability to learn a multivariate bifurcation with 4 independent channels. MOSES-K, refers to MOSES with K-many mixture components. The plots of CircuITS are created by using 2 circuit components.

Consequently, we require an architecture that avoids this trade-off. The ideal model should be capable of representing complex joint dependencies where they exist, while efficiently factoring into independent components where they do not. This motivates our proposal of **Circu**its for **I**rregular Multivariate **T**ime **S**eries (**CircuITS**), a model that leverages the hierarchical nature of Sum-Product Networks (SPNs) [Poon and Domingos, 2011], a class of probabilistic circuits [Choi et al., 2020]. SPNs are computation graphs that represent a joint density as a hierarchy of weighted mixtures (sum nodes) and conditional-independence factorizations (product nodes), with tractable distributions at the leaves. This structure makes both density evaluation and marginalization linear in the size of the graph. By recursively stacking sum and product nodes, CircuITS flexibly models the joint distribution over IMTS forecasting queries. This allows it to capture intricate correlations and independent factors alike, while strictly guaranteeing marginalization consistency. As a result, CircuITS trivially resolves the independent bifurcation task shown in Figure 1. Furthermore, our experiments on five real-world datasets, featuring varying lookback windows and forecasting horizons, demonstrate that CircuITS not only outperforms MOSES on every dataset in terms of joint density but also surpasses ProFITi and establishes a new state-of-the-art. Our contributions can be summarized as follows[1]:

1. We propose CircuITS, a novel architecture to model joint densities of IMTS queries. Unlike previous models, it structurally guarantees marginalization consistency, ensuring valid and non-contradictory predictions for any subset of variables.
2. We introduce an encoder that handles irregular data to generate encodings of forecasting queries and the weights of the SPN.
3. We demonstrate through extensive experiments on five datasets that CircuITS achieves superior joint and marginal density estimation compared to its baselines.

2 Related Work

Recently published approaches that model IMTS often focus on point forecasting using set/graph based models [Yalavarthi et al., 2024, Klötergens et al., 2024, Zhang et al., 2024, Luo et al., 2025, Li et al., 2025]. While effective for deterministic tasks, many downstream applications require quantifying uncertainty.

Early probabilistic approaches for IMTS extrapolation are restricted to predicting the *univariate marginal distributions* (i.e., the distribution of a single channel at a single time step) by modeling a stochastic process with a Neural ODE [Chen et al., 2018]. Models such as NeuralFlows [Biloš et al., 2021], GRU-ODE-Bayes [De Brouwer et al., 2019], and Continuous Recurrent Units (CRU) [Schirmer et al., 2022] fall into this category.

Modeling the joint distribution allows for capturing dependencies, but it introduces the challenge of *marginalization consistency*. This property is not merely a mathematical constraint; it repre-

[1]Code is available here: `https://github.com/kloetergensc/CircuITS-for-IMTS`

sents the correct inductive bias for any model approximating a stochastic process [Øksendal, 2003, Yalavarthi et al., 2025b]. Models that violate this, such as ProFITi [Yalavarthi et al., 2025a], lack this inductive bias, leading to contradictory predictions where the forecast for a subset of variables disagrees with the forecast derived from the joint distribution [Murphy, 2022]. Such inconsistencies render models unreliable for critical decision-making in high-stakes domains like healthcare. While marginalization consistency is essential, current consistent approaches struggle to combine this inductive bias with sufficient expressivity for high-dimensional joint distributions. Gaussian Process Regression (GPR) [Dürichen et al., 2015] satisfies consistency but is severely limited by its reliance on Gaussian assumptions. MOSES (Mixtures of Separable Flows) [Yalavarthi et al., 2025a] attempts to bridge this gap by parameterizing a mixture of multivariate Gaussians combined with separable normalizing flows applied to the univariate marginals. This architecture guarantees consistency and achieves high expressivity for univariate predictions. However, its ability to model complex *joint* dependencies is bottlenecked by the mixture model itself. While Gaussian mixtures are theoretically universal approximators, the amount of data required to accurately estimate an increasing number of components, grows exponentially with the number of variables [Anderson et al., 2014].

TACTiS [Drouin et al., 2022] and TACTiS-2 [Ashok et al., 2023], employ Transformers to parameterize attentional copulas, achieving state-of-the-art performance on multivariate probabilistic forecasting. However, these models are primarily designed for regularly sampled or fully observed settings and their implementations are not readily adaptable to the sparse irregular setting. Both TACTiS and TACTiS-2 do not provide structural guarantees to be valid copulas, but implement learning procedures so that the final model parameters hopefully result in valid copulas.

3 Probabilistic IMTS forecasting

We represent an irregular multivariate time series (IMTS) as a sequence of observations $\mathcal{X}$. Unlike regular time series represented as a matrix, $\mathcal{X}$ is a sequence of L triplets: $\mathcal{X} = \Big((t_n, c_n, y_n)\Big)_{n=1}^{L}$, with $(t_n, c_n, y_n) \in \mathbb{R} \times \{1, \dots, C\} \times \mathbb{R}$, where t_n denotes the continuous timestamp, c_n is the channel index, and y_n is the observed scalar value. The observations are typically sparse and irregularly sampled across time and channels.

Forecasting Query. A forecasting task is defined by a query $\mathcal{Q}$, which is a sequence of N target pairs specifying where predictions are required: $\mathcal{Q} = \Big((t_n^{\text{qry}}, c_n^{\text{qry}})\Big)_{n=1}^{N}$, with $(t_n^{\text{qry}}, c_n^{\text{qry}}) \in \mathbb{R} \times \{1, \dots, C\}$. The goal is to predict the values $\mathbf{y} = (y_1, \dots, y_N)^\top$ corresponding to the query points in $\mathcal{Q}$, conditioned on the observed history $\mathcal{X}$. In general, the order of the sequences $\mathcal{X}$ and $(\mathcal{Q}, \mathbf{y})$ is of no importance, hence we perceive them often as sets.

Probabilistic Objective. Given the stochastic nature of the underlying processes generating IMTS data, point predictions are often insufficient [Murphy, 2022]. Therefore, we aim to model the full conditional joint probability density function over the query targets:

$$p(\mathbf{y} \mid \mathcal{Q}, \mathcal{X}) : \mathbb{R}^N \to \mathbb{R}_{\geq 0} \tag{1}$$

This mapping must be valid for any query size N and any configuration of query points $\mathcal{Q}$. To ensure the model serves as a reliable approximation of the underlying stochastic process, we impose the following strict requirements:

1. Flexibility and Joint Modelling. The model must accept variable-length observation sequences $\mathcal{X}$ and queries $\mathcal{Q}$. Crucially, it must model the dependencies between target variables by outputting a joint distribution, rather than factorized independent marginals.

2. Marginalization Consistency. A valid stochastic process must satisfy the Kolmogorov extension theorem [Øksendal, 2003]. This implies that the predicted marginal distribution of a query subset must match the integrated joint distribution. Let $\mathbf{y}_S$ denote a subset of variables that is dropped from $\mathbf{y}$. Then the model fulfills *Marginalization Consistency* if:

$$\int p(\mathbf{y} \mid \mathcal{Q}, \mathcal{X}) \, \mathrm{d}\mathbf{y}_S = p(\mathbf{y}_{-S} \mid \mathcal{Q}_{-S}, \mathcal{X}) \tag{2}$$

This property ensures that the prediction for a specific variable does not change arbitrarily simply because we decided to also query another variable [Yalavarthi et al., 2025b].

4 CircuITS

CircuITS models the joint distribution over IMTS forecasting queries in two stages. First, for each channel, we model the marginal distribution over its query points as a mixture. Second, we combine these per-channel mixtures into a single joint distribution over all channels using a recursive sum-product network (SPN [Poon and Domingos, 2011], Section 4.3). Figure 2 illustrates the full pipeline.

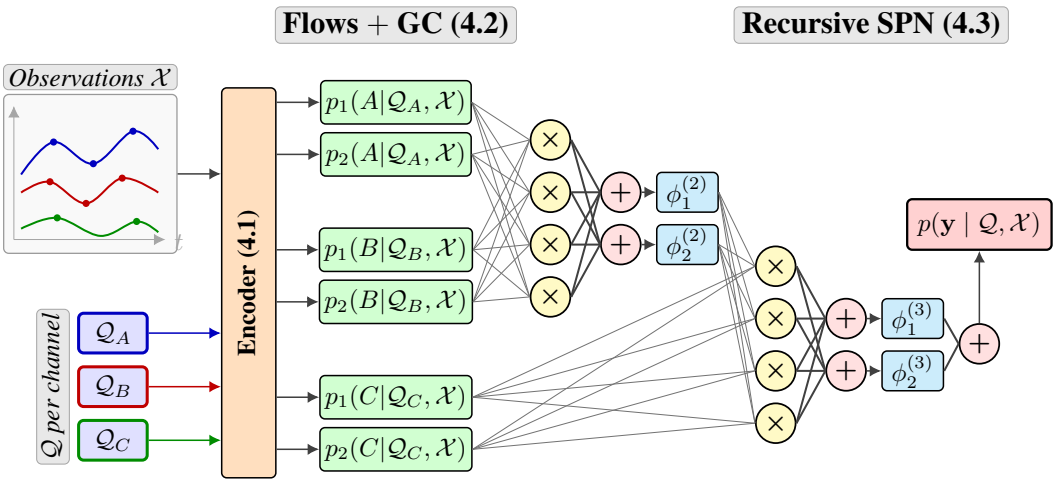

Figure 2: CircuITS' architecture illustrated for 3 channels A, B, C and $K = 2$ mixture component. The channel-wise component distributions are modeled using separable Flows and Gaussian Copulas (GC) and then combined by the recursive sum-product network (SPN).

4.1 Encoder

We aim to infer the conditional joint distributions $p(\mathbf{y} \mid \mathcal{Q}, \mathcal{X})$ of IMTS forecasting queries. Therefore, we need an encoder to extract the relevant information contained in $\mathcal{X}$ and $\mathcal{Q}$ into latent embeddings which can be used to parameterize the distributions. To guarantee that we compute likelihoods marginalization consistently, the elements of $\mathcal{Q}$ need to be encoded *seperable*.

Observation Representation. Each observation is a triplet consisting of a continuous timestamp $t_n \in \mathbb{R}$, a channel index $c_n \in \{1, \dots, C\}$, and a scalar value $y_n \in \mathbb{R}$. We project these discrete observations into a latent feature space of dimension D ($\mathbf{H}^{\text{obs}} \in \mathbb{R}^{L \times D}$). The n-th observation embedding $\mathbf{h}_n \in \mathbb{R}^D$ is computed as $\mathbf{h}_n = \left[\phi^{\text{TIME}}(t_n) \,\|\, \mathbf{E}^{\text{chan}}_{c_n} \,\|\, y_n\right]$, where $\mathbf{E}^{\text{chan}} \in \mathbb{R}^{C \times D}$ is a learnable channel embedding matrix, $\phi^{\text{TIME}}(\cdot)$ is a sinusoidal time embedding [Kazemi et al., 2019], and $[\cdot\|\cdot]$ denotes concatenation.

Intra-Channel Aggregation. To aggregate an arbitrary number of observations per channel into a fixed-size representation, we employ Multi-Head Attention (MHA) [Vaswani et al., 2017]. We introduce a set of static learnable embeddings $\mathbf{E}^{\text{proto}} \in \mathbb{R}^{C \times D}$, where each row represents a channel prototype. These serve as queries in a masked Cross-Attention layer where we use the observation embeddings $\mathbf{H}^{\text{obs}}$ as keys and values:

$$\mathbf{H}^{\text{chan}} = \text{MHA}\left(\mathbf{E}^{\text{proto}}, \mathbf{H}^{\text{obs}}, \mathbf{H}^{\text{obs}}; \mathbf{M}\right) \tag{3}$$

Here, $\mathbf{M}$ is a binary attention mask where $M_{c,n} = 1$ if observation n belongs to channel c, and 0 otherwise. This operation effectively *groups* observations by channel, producing a matrix $\mathbf{H}^{\text{chan}} \in \mathbb{R}^{C \times D}$ that summarizes the observations of each channel.

Inter-Channel Interaction. To capture dependencies between variables (e.g., correlations between channels), we update the channel representations using a Self-Attention layer:

$$\tilde{\mathbf{H}} = \mathbf{H}^{\text{chan}} + \text{MHA}(\mathbf{H}^{\text{chan}}, \mathbf{H}^{\text{chan}}, \mathbf{H}^{\text{chan}}) \tag{4}$$

The updated states $\tilde{\mathbf{H}} \in \mathbb{R}^{C \times D}$ serve as the context for the subsequent generation tasks. If a channel is completely unobserved, its representation in $\tilde{\mathbf{H}}$ is derived solely from its static query and interactions with other channels.

Encoding the Query. For a target query at time t_n^{qry} on channel c_n, we generate the context embedding $\mathbf{e}_n$ by retrieving the corresponding row $\tilde{\mathbf{h}}_{c_n}$ from the channels' hidden state and combining it with the query time: $\mathbf{e}_n = \text{Lin}^{\text{QRY}}\left(\left[\phi^{\text{TIME}}(t_n^{\text{qry}}) \,\|\, \tilde{\mathbf{h}}_{c_n}\right]\right)$. For simplicity, we use a single linear layer (Lin^{QRY}) to project the concatenation back to dimension D. Note that $\tilde{\mathbf{h}}_{c_n}$ is computed before any query is introduced, so per-channel parameters depend only on $\mathcal{X}$ — a property we will rely on later to establish marginalization consistency (Appendix B.2). We provide a detailed architecture diagram of our encoder in Appendix D.1.

4.2 Marginal Distributions of Channels (Leaf Distributions)

For each channel c, we model K *candidate* distributions (leaves) over its query elements. Section 4.3 then combines these per-channel candidates into the full joint via the SPN. Intuitively, each candidate captures one plausible regime that the channel could follow. In the bifurcation task of Figure 1, for instance, $K = 2$ candidates per channel suffice: one for the upward branch and one for the downward branch. These K candidates take the role of mixture components in the resulting SPN, and we use both terms interchangeably below.

For a given channel c, let $\mathbf{y}^{(c)} = (y_1, \ldots, y_{N_c})$ denote the sequence of target values (observations) and $\mathbf{E}^{(c)} = (\mathbf{e}_1, \ldots, \mathbf{e}_{N_c})$ denote the corresponding context embeddings generated by the encoder, where $\mathbf{e}_i \in \mathbb{R}^D$. We split the embeddings into K partitions $\mathbf{e}_i^k \in \mathbb{R}^{D'}$, with $D = KD'$, so that each candidate is parameterized from a separate slice of the encoder output.

Univariate Marginals. The Deep Sigmoidal Flow (DSF) [Huang et al., 2018] approximates the cumulative distribution function (CDF) via a composition of L^{DSF} monotonic layers, denoted as $F(y_i; \theta_i^k) = f_{L^{\text{DSF}}} \circ \cdots \circ f_1(y_i)$. The l-th layer operation is defined as:

$$f_l(x) = \sigma(A_l x + b_l), \tag{5}$$

where $\sigma(\cdot)$ is the sigmoid activation function. The parameters θ_i^k include the weight matrices A and bias vectors b for all layers. To condition the flow on the input context, we use a two-layer MLP that maps the query embedding e_i^k to the instance-specific flow parameters θ_i^k. To ensure the strict monotonicity required for a valid normalizing flow, all elements of A are enforced to be strictly positive. The sigmoid activation in the last layer ensures the output lies in $(0, 1)$, effectively modeling the CDF of a continuously defined distribution. The density for the observation y_i is then computed via the change of variables formula:

$$\log p(y_i | e_i^k) = \log \frac{\partial F(y_i; \theta_i^k)}{\partial y_i} \tag{6}$$

The derivative $\frac{\partial F}{\partial y_i}$ is obtained via a forward-pass application of the chain rule.

Gaussian Copula. The Gaussian copula is a statistical construct that models the dependence structure between multiple random variables by mapping their marginal CDF to a multivariate standard normal distribution. It assumes that the dependence between variables is captured entirely by a correlation matrix $\mathbf{R}$, allowing for the decoupling of marginal distributions from the joint dependency structure. The log-likelihood of a Gaussian copula is defined as:

$$\mathcal{L}^{\text{GC}}(\mathbf{u}, R) = -\frac{n}{2} \log |R| - \frac{1}{2} \sum_{i=1}^{n} \mathbf{z}_i^\top (R^{-1} - I) \mathbf{z}_i, \tag{7}$$

where, $\mathbf{z}_i = \Phi^{-1}(\mathbf{u}_i)$, represents the inverse CDF of a standard normal distribution [Nelsen, 2006]. To apply Gaussian copulas for our channel-wise marginal distributions we compute a correlation matrix

$\mathbf{R}^{c,k} \in \mathbb{R}^{N_c \times N_c}$ for each channel c and mixture component k. First, we project the embeddings into a feature space of dimension D^{CORR} in which we aim to model correlations:

$$\mathbf{v}_i^{c,k} = \tanh(\text{MLP}_{\text{corr}}(\mathbf{e}_i^k)) \quad \in \mathbb{R}^{D^{\text{CORR}}} \tag{8}$$

The raw correlation score between query target i and target j is the scaled inner product of their features. The $\tanh$ activation ensures that any $\mathbf{v}_i^\top \mathbf{v}_j$ is within $(-D^{\text{CORR}}, D^{\text{CORR}})$. Therefore, instead of scaling by $1/\sqrt{D^{\text{CORR}}}$ which is commonly used in scaled dot product attention, we divide by D^{CORR} to ensure $\mathbf{R}^{c,k} \in (-1, 1)$. To ensure a valid correlation structure, we obtain $\mathbf{R}^{c,k}$ by replacing the diagonal of $\dot{R}^{c,k}$ with ones:

$$\mathbf{R}^{c,k} = \dot{R}^{c,k} + (\mathbf{I} - \text{diag}(\dot{R}^{c,k})), \quad \text{with } \dot{R}_{ij}^{c,k} = \frac{(\mathbf{v}_i^{c,k})^\top (\mathbf{v}_j^{c,k})}{D^{\text{CORR}}} \tag{9}$$

This preserves positive-definiteness: $\dot{R}^{c,k}$ is a Gram matrix and thus positive semi-definite, and the correction $(\mathbf{I} - \text{diag}(\dot{R}^{c,k}))$ is a diagonal matrix with strictly positive entries since $\dot{R}_{ii}^{c,k} = \|\mathbf{v}_i^{c,k}\|^2 / D^{\text{CORR}} \leq 1$ (with equality only when $\mathbf{v}_i^{c,k}$ is fully saturated by $\tanh$). With $\mathbf{R}^{c,k}$ we can combine the univariate marginals modeled by DSF to the log-likelihood function of the channel-wise candidate distribution:

$$\log p_{c,k} = \underbrace{\sum_{i=1}^{N_c} \log p(y_i | e_i^k)}_{\text{univariate marginals}} + \underbrace{\mathcal{L}^{\text{GC}}(F(y_i, \theta_i^k), \mathbf{R}^{c,k})}_{\text{copula density}} \tag{10}$$

Since we aim for CircuITS to be marginalization consistent, the channel-wise candidate distributions are designed accordingly: normalizing flows model the univariate marginals, while Gaussian copulas combine them into a joint distribution. We provide a more formal argument in Appendix B.2.

4.3 Recursive Sum-Product Aggregation

To capture dependencies across channels, we propose a *Recursive Sum-Product Network (SPN)* that combines the per-channel mixtures from Section 4.2 into a single joint distribution. An SPN is a computation graph over three kinds of nodes: *leaves*, which represent tractable density functions over a subset of variables; *product nodes*, which multiply children with disjoint scopes and thereby encode conditional independence; and *sum nodes*, which take a convex combination of children with the same scope and thereby encode mixtures. Provided every product is over disjoint scopes (decomposability) and every sum is over identical scopes (smoothness), the network represents a valid joint density that can be evaluated and marginalized in time linear in its size [Choi et al., 2020].

In CircuITS, each channel c contributes K leaves $\{p_{c,k}\}_{k=1}^K$ — the channel-wise component distributions defined in Section 4.2. We build the joint density over all channels by processing them in sequence, alternating product nodes (to combine the previously aggregated channels with a new channel) and sum nodes (to mix the resulting components back to a fixed size K). Let $p^{(c)}(\mathbf{y}_{1:c})$ denote the joint density over the first c channels, represented by a vector of K component densities $\mathbf{\Phi}^{(c)}(\mathbf{y}_{1:c}) = [\phi_1^{(c)}, \ldots, \phi_K^{(c)}]^\top$. The aggregation proceeds recursively:

Initialization ($c = 1$): For the first channel, the joint state is simply the vector of its leaf densities:

$$\phi_k^{(1)}(\mathbf{y}_1) = p_{1,k}(y_1), \quad \forall k \in \{1, \ldots, K\} \tag{11}$$

Recursive Step ($c > 1$): To incorporate channel c, we instantiate one product node for every pair (i, j) of a previous component $\phi_i^{(c-1)}$ and a new leaf $p_{c,j}$. This yields K^2 product terms:

$$\psi_m(\mathbf{y}_{1:c}) = \phi_i^{(c-1)}(\mathbf{y}_{1:c-1}) \cdot p_{c,j}(y_c) \tag{12}$$

where m indexes the pair (i, j) in the flattened space of size K^2. Decomposability is satisfied because $\phi_i^{(c-1)}$ has scope $\mathbf{y}_{1:c-1}$ and $p_{c,j}$ has scope y_c. Intuitively, each product term represents a scenario in which the previous channels and the new one can be conditionally independent given a pair of latent components. A naive composition would let the number of components grow as K^c, since each

new channel multiplies the size of the joint state by K. We therefore follow each product layer with K sum nodes that project the K^2 products back onto a basis of size K. Each sum node k at level c takes a convex combination of the K^2 products and shares the same scope $\mathbf{y}_{1:c}$ as its children, so smoothness holds:

$$\phi_k^{(c)}(\mathbf{y}_{1:c}) = \sum_{m=1}^{K^2} w_{m,k}^{(c)} \cdot \psi_m(\mathbf{y}_{1:c}), \quad \text{with} \sum_{m=1}^{K^2} w_{m,k}^{(c)} = 1 \tag{13}$$

Here, $\mathbf{W}^{(c)} \in \mathbb{R}^{K^2 \times K}$ is a context-dependent weight matrix that captures the correlations between the previous joint state and the new channel. Intuitively: since all product nodes are valid distributions over the same random variables, any convex sum of them is also a valid distribution. Therefore, every sum node represents a valid density too.

Full Joint Distribution and Training Objective. After processing all C channels, the final joint likelihood is given by a weighted sum of the final K components using the root weights $\mathbf{w}^{\text{root}}$:

$$p(\mathbf{y} \mid \mathcal{Q}, \mathcal{X}) = \sum_{k=1}^{K} w_k^{\text{root}} \cdot \phi_k^{(C)}(\mathbf{y}) \tag{14}$$

Because every leaf, product, and sum node in the construction admits an exact tractable density, this joint log-likelihood can be evaluated end-to-end on a single forward pass through the recursion in (12)–(13) and is fully differentiable in the encoder, the leaf parameters, and the SPN weights. We therefore train CircuITS by directly minimizing the normalized joint Negative Log Likelihood (njNLL), which divides the joint log-likelihood by the number of query targets N to make the objective comparable across queries of varying size:

$$\text{njNLL}(\mathbf{y} \mid \mathcal{Q}, \mathcal{X}) = \frac{-\log p(\mathbf{y} \mid \mathcal{Q}, \mathcal{X})}{N}. \tag{15}$$

For numerical stability, all computations are carried out in the log-domain: products become sums and the sum layer is implemented via the Log-Sum-Exp (LSE) function. The recursive update from (12) thus becomes: $\log \phi_k^{(c)} = \text{LSE}\left(\log \mathbf{\Psi} + \log \mathbf{W}_{\cdot,k}^{(c)}\right)$, where $\log \mathbf{\Psi}$ collects the log-probabilities of the pairwise products.

SPN Weights. The sum-layer weights are conditioned on the observations $\mathcal{X}$ through the channel embeddings $\tilde{\mathbf{H}}$ produced by the encoder (Section 4.1). Starting from a set of learnable channel-wise queries $\mathbf{E}^{\text{cw}} \in \mathbb{R}^{C \times D}$, we attend over $\tilde{\mathbf{H}}$ via multi-head attention:

$$\mathbf{Z} = \text{MHA}(\mathbf{E}^{\text{cw}}, \tilde{\mathbf{H}}, \tilde{\mathbf{H}}), \quad \mathbf{Z} \in \mathbb{R}^{C \times D}. \tag{16}$$

Each row $\mathbf{Z}_c$ then yields the weights for the c-th sum layer (and, for $c = 1$, the root weights):

$$\mathbf{w}^{\text{root}} = \text{softmax}\big(\text{Lin}^{\text{ROOT}}(\mathbf{Z}_1)\big), \quad \mathbf{W}^{(c)} = \text{softmax}\big(\text{Lin}^{\text{W}}(\mathbf{Z}_c)\big) \tag{17}$$

where $\text{Lin}^{\text{ROOT}} : D \to K$ and $\text{Lin}^{\text{W}} : D \to K^2 \times K$ are linear projections, and the softmax in (17) is taken over the K^2 dimension so that each column forms a valid mixture (cf. (13)).

4.4 Sampling

Samples from the joint distribution are generated by ancestral sampling: starting at the root and traversing down to the leaves. Because of our recursive construction, this top-down traversal corresponds to iterating through channels from C down to 1 (see Appendix D.2 for details). At each leaf, given the active component k, we draw $z \sim \mathcal{N}(0, \mathbf{R}^{c,k})$, map it to the uniform domain via $u = \Phi(z)$, and numerically invert the DSF by bisection to obtain $\hat{y}_i = F^{-1}(u_i; \theta_i^k)$ [Drouin et al., 2022].

4.5 Computational Complexity

For Gaussian-based joint models, the bottleneck is the Cholesky decomposition over all N query points, which scales as $\mathcal{O}(N^3)$. CircuITS avoids this by handling inter-channel dependencies through the SPN and restricting matrix operations to the per-channel leaves. Assuming each channel has

N/C query points, the total cost drops to $\mathcal{O}(C \cdot (N/C)^3) = \mathcal{O}(N^3/C^2)$ — a saving of factor C^2 over the dense Gaussian baseline [Yalavarthi et al., 2025b]. The price is cubic scaling in the number of mixture components K: each recursive step projects K^2 products back to K components via $\mathbf{W}^{(c)} \in \mathbb{R}^{K^2 \times K}$, costing $\mathcal{O}(K^3)$ per channel. We therefore keep $K \leq 4$ in our experiments. Although the SPN aggregation is sequential by construction, it can be parallelized across channels using the Hillis-Steele scan [Hillis and Steele, 1986] (see Appendix D.3). We analyze which joint densities the recursive projection onto K sum nodes can represent in Appendix C.

4.6 Relation to the SPN Literature

CircuITS combines two ideas that the SPN literature has studied separately. Conditional SPNs [Shao et al., 2020] make the sum weights functions of a set of conditioning variables, which is also true for CircuITS. However, in a conditional SPN the modeled variables are fixed, so every remaining variable is available to condition on. In IMTS forecasting, which random variables need to be predicted (i.e. the query $\mathcal{Q}$) is itself part of the model input, and the sum weights must not depend on it to preserve marginalization consistency (Appendix B). Dynamic SPNs [Melibari et al., 2016] handle variable-length sequence data by unrolling a template SPN over time steps. We instead unroll a template over the C channels and absorb the variable number of query points into the leaves, whose scope $\mathbf{y}^{(c)}$ has instance-dependent size N_c.

5 Experiments

We evaluate CircuITS in two settings. First, we use a synthetic bifurcation task to probe whether the model can recover distinct multi-modal channel dynamics. Second, we benchmark CircuITS on five real-world IMTS datasets (one climate, three medical, one human motion) covering a range of observation and prediction horizons, comparing against state-of-the-art probabilistic IMTS forecasting baselines and ablating the main components of CircuITS.

5.1 Independent Multi-channel Bifurcation

We validate CircuITS' capabilities on learning meaningful SPNs on a synthetic bifurcation task (Figure 1). The synthetic dataset consists of 10,000 simulated IMTS with four channels. Each IMTS channel simulates a trajectory that starts as stationary Gaussian noise and bifurcates into a biased random walk (Brownian motion with drift) shortly after the final observation. MOSES fails to capture the distinct modes even with 16 components due to optimization difficulties in high-dimensional mixture spaces. In contrast, CircuITS solves the task with 2 components per channel, successfully isolating the upward and downward trends. We report the corresponding njNLLs in Appendix E.

5.2 Benchmark on Real-World Datasets

We evaluate CircuITS on five datasets: one climate dataset (USHCN), three medical datasets (Physionet-2012, MIMIC-III, MIMIC-IV), and one human motion dataset (Human Activity [Kaluža et al., 2010]). Our evaluation covers a broad range of varying observation and prediction horizons, ranging from observing 36 hours/months to predict the next 3 steps (36–3*), to longer-range forecasting scenarios (e.g., 12–36, where the models are tasked to predict the final 36 hours / months). For Human Activity, a 4 s window takes the role of the 48 hours (Appendix F). For a fair comparison, we adopt the preprocessing, splitting, and metric-computation protocol of Yalavarthi et al. [2025b]. Each model is trained and evaluated five times across different train/validation/test splits, and every model is evaluated on the *same* five splits so that runs are paired across methods. Since the protocol is identical, we reuse their reported numbers for shared baselines and settings wherever applicable. Additional information about the datasets (Appx. F) and hyperparameters (Appx. G) can be found in the appendix.

Metrics. Besides njNLL (Equation (15)), we report the mNLL (Appendix I.2), the Energy Score and CRPS (Appendix I.1), and the MSE of the predictive mean (Table 10). CRPS and mNLL assess only the univariate marginals and are blind to the dependence between channels, njNLL and the Energy Score assess the full joint distribution, and the MSE only measures point accuracy. Of the two joint metrics, njNLL is the more discriminative one: the Energy Score has low power to detect a

Table 1: Comparison of njNLL for different forecasting splits. Lower values are better. Best results are in **bold**, second best in *italics*. The $\star$ represents that the forecasting horizon includes the next n steps, instead of n-many hours/months.

	Model	USHCN	PhysioNet'12	MIMIC-III	MIMIC-IV	Activity
36–3*	NeuralFlows	0.775 ± 0.15	0.496 ± 0.00	0.998 ± 0.18	0.689 ± 0.09	0.360 ± 0.16
	ProFITi	-3.226 ± 0.23	**-0.647 ± 0.08**	*-0.377 ± 0.03*	*-1.777 ± 0.07*	*-0.023 ± 0.03*
	MOSES	*-3.357 ± 0.18*	-0.491 ± 0.04	-0.305 ± 0.03	-1.668 ± 0.10	0.144 ± 0.05
	CircuITS	**-3.789 ± 0.22**	*-0.550 ± 0.01*	**-0.574 ± 0.08**	**-2.113 ± 0.04**	**-0.062 ± 0.02**
36–12	NeuralFlows	1.344 ± 0.13	0.708 ± 0.05	1.426 ± 0.12	1.033 ± 0.04	0.715 ± 0.08
	ProFITi	*-3.355 ± 0.20*	**-0.768 ± 0.04**	*0.095 ± 0.16*	-0.511 ± 0.43	0.186 ± 0.05
	MOSES	-1.790 ± 0.03	-0.315 ± 0.02	0.275 ± 0.04	*-0.517 ± 0.03*	*0.072 ± 0.02*
	CircuITS	**-3.755 ± 0.16**	*-0.536 ± 0.04*	**-0.253 ± 0.01**	**-1.201 ± 0.04**	**-0.120 ± 0.02**
24–24	NeuralFlows	1.436 ± 0.20	1.097 ± 0.04	1.457 ± 0.06	1.195 ± 0.03	1.060 ± 0.12
	ProFITi	-1.513 ± 0.14	*-0.355 ± 0.24*	*0.196 ± 0.40*	0.402 ± 0.21	0.428 ± 0.04
	MOSES	*-1.550 ± 0.65*	-0.298 ± 0.03	0.198 ± 0.01	*-0.478 ± 0.01*	*0.107 ± 0.03*
	CircuITS	**-3.585 ± 0.40**	**-0.523 ± 0.01**	**-0.112 ± 0.05**	**-0.851 ± 0.10**	**-0.074 ± 0.02**
12–36	NeuralFlows	1.456 ± 0.27	1.436 ± 0.19	1.525 ± 0.05	1.186 ± 0.06	1.169 ± 0.10
	ProFITi	-1.284 ± 0.51	*-0.291 ± 0.42*	*0.285 ± 0.35*	-0.272 ± 0.40	0.597 ± 0.03
	MOSES	*-1.519 ± 0.17*	-0.063 ± 0.05	0.362 ± 0.00	*-0.294 ± 0.03*	*0.157 ± 0.03*
	CircuITS	**-3.400 ± 0.38**	**-0.402 ± 0.05**	**-0.462 ± 0.07**	**-0.859 ± 0.00**	**-0.021 ± 0.03**

misspecified dependence structure, with likelihood-based scores being far more discriminative [Marcotte et al., 2023]. Moreover, njNLL can be optimized directly, whereas the Energy Score is not differentiable end-to-end in CircuITS, since sampling inverts the DSF by bisection (Section 4.4). We therefore focus on njNLL and also tune hyperparameters on it.

Baselines. In our main experiment we compare against state-of-the-art probabilistic forecasting models for joint density estimation for IMTS forecasting: ProFITi [Yalavarthi et al., 2025a] and MOSES. While MOSES guarantees marginalization consistency ProFITi does not as outlined in previous work. [Yalavarthi et al., 2025b]. Additionally, we compare against NeuralFlows [Biloš et al., 2021], that predicts independent univariate normal distributions. Since it does not model dependencies between variables it is automatically marginalization consistent.

Results. The primary goal of probabilistic forecasting is to accurately model the joint distribution over future queries. As shown in Table 1, CircuITS achieves the lowest (best) njNLL in 18 of the 20 scenarios. While we train each model on njNLL, we also evaluate it on the marginal negative log-likelihood (mNLL) (Table 9 in Appendix I.2). When we compare the mNLL, CircuITS outperforms the baselines on majority of scenarios. On most datasets ProFITi has a relatively high mNLL highlighting the importance of marginalization consistency.

On the sample-based scores (Tables 7 and 8), CircuITS attains the best CRPS in 13 and the best or second-best CRPS in 18 of the 20 settings. On the Energy Score it is best in 10 and best or second best in 15 of 20 settings, ahead of the strongest baseline (mean rank 1.75 vs. 1.95 for MOSES). On the MSE of the predictive mean (36–3*) it is the most accurate model on three of the four datasets (Table 10). A low njNLL does not automatically translate into low sample-based scores: njNLL rewards placing sharp density on the observed values, whereas CRPS and the Energy Score are distance-based and penalize occasional outlying samples. They therefore disagree with njNLL precisely where sharp predictions are attainable. On tasks with relatively low njNLL (USHCN, and 36–3* on all datasets) no model wins consistently, and the best Energy Score is attained by three different baselines, among them NeuralFlows, which models the channels as independent univariate normals. In the remaining scenarios, CircuITS is best on both scores in most cases. It fares better on CRPS than on the Energy Score, because CRPS averages over the query points, damping a single badly placed coordinate by $1/N$, whereas in the Euclidean norm of the Energy Score the same coordinate inflates the distance of every sample. We show an experiment about the impact of channel order in Appx. K and evaluate CircuITS' efficiency in Appx. N.

Table 2: Showing the njNLL of CircuITS compared and with ablations on the 36–12 task.

	USHCN	PhysioNet	MIMIC-III	MIMIC-IV
CircuITS	**-3.755 ± 0.16**	**-0.536 ± 0.04**	**-0.253 ± 0.01**	**-1.201 ± 0.04**
w/o SPN	-3.527 ± 0.05	-0.299 ± 0.05	0.013 ± 0.02	-0.892 ± 0.00
w/o DSF	-1.283 ± 0.30	-0.253 ± 0.17	0.040 ± 0.03	-1.019 ± 0.05
w/o GC	-3.588 ± 0.08	-0.303 ± 0.01	-0.082 ± 0.06	-0.972 ± 0.02
w/ TF	-2.702 ± 0.39	-0.173 ± 0.19	-0.167 ± 0.15	-1.167 ± 0.01

Ablations In order to validate the effectivity of the Sum-Product Network we evaluate a model (w/o SPN) that combines the channel marginals within a single product without mixing multiple components. This ablation models channels as independent marginals and is less expressive. Additionally, we also present ablations where we replace the DSF with normal distributions (w/o DSF) and drop the Gaussian Copula (w/o GC). Finally, we also replace our encoder with Tripletformer (TF) [Yalavarthi et al., 2023]. We provide more details on the ablation set up in Appendix H. The respective results are shown in Table 2. Every ablation is worse than the full model on every dataset: the components are complementary rather than redundant. *Which component matters most is dataset-dependent.* The largest degradation stems from the DSF leaves on USHCN ($+2.47$) and MIMIC-III ($+0.29$), from the encoder on PhysioNet ($+0.36$), and from the SPN on MIMIC-IV ($+0.31$). Rather than a global ranking, the ablation thus shows which property of a dataset each component addresses: flexible marginals are decisive where the channel marginals are far from Gaussian (USHCN) and merely helpful where they are not (MIMIC-IV, $+0.18$). *The two dependence mechanisms contribute a small but consistent amount.* Removing the SPN (inter-channel dependence) costs $+0.23$ to $+0.31$ njNLL, removing the Gaussian copula (intra-channel dependence) $+0.17$ to $+0.23$. These two components buy expressivity *without* sacrificing marginalization consistency. *The SPN ablation also diagnoses leaf specialization.* w/o SPN is exactly the case $K = 1$. If the K leaves of a channel converged to the same distribution, every sum node would mix identical components and the SPN would have no effect; the consistent gap therefore shows that the leaves specialize. *The encoder mainly affects stability.* Tripletformer is nearly free on MIMIC-III ($+0.09$) and MIMIC-IV ($+0.03$) but costly on USHCN ($+1.05$) and PhysioNet ($+0.36$), with standard deviations of up to ± 0.39, whereas the full model stays within ± 0.16 on all datasets (similarly, w/o DSF: ± 0.30 on USHCN). The full model is thus both the most accurate and the most stable across splits.

6 Limitations

The sequential SPN construction makes the modeled joint density dependent on channel order. We use a fixed arbitrary ordering since searching all C permutations is intractable, although the empirical impact is small relative to the gap between models (Appendix K). Because inter-channel dependencies are captured via discrete mixtures rather than continuous correlations, smooth structures such as linear dependencies between channels cannot be represented compactly and must be approximated by a finite number of components. The $\mathcal{O}(N^3/C^2)$ complexity in Section 4.5 additionally assumes evenly distributed queries. On real-world IMTS where queries concentrate on few channels (see Appendix N), the per-channel grouping amortizes poorly and CircuITS can have longer epoch-times than its baselines (Table 14). More generally, the circuit structure itself is fixed: channels are always combined sequentially along one chain, and the structure is neither learned nor adapted to the data.

7 Conclusion

In this work, we introduced CircuITS, a novel Sum-Product Network architecture for irregular multivariate time series that resolves the trade-off between expressivity and marginalization consistency. Our experiments demonstrate that CircuITS achieves state-of-the-art joint density estimation on real-world datasets while structurally guaranteeing valid, non-contradictory forecasts. For future work, we plan to explore deeper and less rigid circuit structures than the fixed sequential chain, e.g. a tree-structured aggregation that recursively splits the channels into groups, aggregates each group independently, and merges the resulting sub-circuits pairwise.

Acknowledgments and Disclosure of Funding

The authors gratefully acknowledge the computing time granted by the Resource Allocation Board and provided on the supercomputer Grete at NHR-Nord@Göttingen as part of the NHR infrastructure. The calculations for this research were conducted with computing resources under the project nim00026.

References

Joseph Anderson, Mikhail Belkin, Navin Goyal, Luis Rademacher, and James Voss. The More, the Merrier: The Blessing of Dimensionality for Learning Large Gaussian Mixtures. In *Proceedings of The 27th Conference on Learning Theory*, pages 1135–1164. PMLR, May 2014.

Arjun Ashok, Étienne Marcotte, Valentina Zantedeschi, Nicolas Chapados, and Alexandre Drouin. TACTiS-2: Better, Faster, Simpler Attentional Copulas for Multivariate Time Series. In *The Twelfth International Conference on Learning Representations*, October 2023.

Marin Biloš, Johanna Sommer, Syama Sundar Rangapuram, Tim Januschowski, and Stephan Günnemann. Neural Flows: Efficient Alternative to Neural ODEs. In *Advances in Neural Information Processing Systems*, volume 34, pages 21325–21337. Curran Associates, Inc., 2021.

Zhengping Che, Sanjay Purushotham, Kyunghyun Cho, David Sontag, and Yan Liu. Recurrent Neural Networks for Multivariate Time Series with Missing Values. *Scientific Reports*, 8(1):6085, April 2018. ISSN 2045-2322. doi: 10.1038/s41598-018-24271-9.

Ricky T. Q. Chen, Yulia Rubanova, Jesse Bettencourt, and David K Duvenaud. Neural Ordinary Differential Equations. In *Advances in Neural Information Processing Systems*, volume 31. Curran Associates, Inc., 2018.

Y Choi, Antonio Vergari, and Guy Van den Broeck. Probabilistic circuits: A unifying framework for tractable probabilistic models. *UCLA. URL: http://starai.cs.ucla.edu/papers/ProbCirc20.pdf*, page 6, 2020.

Edward De Brouwer, Jaak Simm, Adam Arany, and Yves Moreau. GRU-ODE-Bayes: Continuous Modeling of Sporadically-Observed Time Series. In *Advances in Neural Information Processing Systems*, volume 32. Curran Associates, Inc., 2019.

Alexandre Drouin, Étienne Marcotte, and Nicolas Chapados. TACTiS: Transformer-Attentional Copulas for Time Series. In *Proceedings of the 39th International Conference on Machine Learning*, pages 5447–5493. PMLR, June 2022.

Robert Dürichen, Marco A. F. Pimentel, Lei Clifton, Achim Schweikard, and David A. Clifton. Multitask Gaussian Processes for Multivariate Physiological Time-Series Analysis. *IEEE Transactions on Biomedical Engineering*, 62(1):314–322, January 2015. ISSN 1558-2531. doi: 10.1109/TBME.2014.2351376.

W. Daniel Hillis and Guy L. Steele. Data parallel algorithms. *Commun. ACM*, 29(12):1170–1183, December 1986. ISSN 0001-0782. doi: 10.1145/7902.7903.

Chin-Wei Huang, David Krueger, Alexandre Lacoste, and Aaron Courville. Neural Autoregressive Flows. In *Proceedings of the 35th International Conference on Machine Learning*, pages 2078–2087. PMLR, July 2018.

Alistair Johnson, Lucas Bulgarelli, Tom Pollard, Steven Horng, Leo Anthony Celi, and Roger Mark. MIMIC-IV. *PhysioNet*, January 2023a. doi: 10.13026/6mm1-ek67. URL `https://doi.org/10.13026/6mm1-ek67`. Version 2.2.

Alistair E. W. Johnson, Tom J. Pollard, Lu Shen, Li-wei H. Lehman, Mengling Feng, Mohammad Ghassemi, Benjamin Moody, Peter Szolovits, Leo Anthony Celi, and Roger G. Mark. MIMIC-III, a freely accessible critical care database. *Scientific Data*, 3(1):160035, May 2016. ISSN 2052-4463. doi: 10.1038/sdata.2016.35.

Alistair E. W. Johnson, Lucas Bulgarelli, Lu Shen, Alvin Gayles, Ayad Shammout, Steven Horng, Tom J. Pollard, Sicheng Hao, Benjamin Moody, Brian Gow, Li-wei H. Lehman, Leo A. Celi, and Roger G. Mark. MIMIC-IV, a freely accessible electronic health record dataset. *Scientific Data*, 10(1):1, January 2023b. ISSN 2052-4463. doi: 10.1038/s41597-022-01899-x.

Boštjan Kaluža, Violeta Mirchevska, Erik Dovgan, Mitja Luštrek, and Matjaž Gams. An agent-based approach to care in independent living. In *Ambient Intelligence*, pages 177–186. Springer, 2010.

Seyed Mehran Kazemi, Rishab Goel, Sepehr Eghbali, Janahan Ramanan, Jaspreet Sahota, Sanjay Thakur, Stella Wu, Cathal Smyth, Pascal Poupart, and Marcus Brubaker. Time2Vec: Learning a Vector Representation of Time, July 2019.

Diederik P. Kingma and Jimmy Ba. Adam: A method for stochastic optimization. In Yoshua Bengio and Yann LeCun, editors, *3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Conference Track Proceedings*, 2015. URL http://arxiv.org/abs/1412.6980.

Christian Klötergens, Vijaya Krishna Yalavarthi, Maximilian Stubbemann, and Lars Schmidt-Thieme. Functional Latent Dynamics for Irregularly Sampled Time Series Forecasting. In *Machine Learning and Knowledge Discovery in Databases. Research Track*, pages 421–436, Cham, 2024. Springer Nature Switzerland. ISBN 978-3-031-70359-1. doi: 10.1007/978-3-031-70359-1_25.

Boyuan Li, Yicheng Luo, Zhen Liu, Junhao Zheng, Jianming Lv, and Qianli Ma. HyperIMTS: Hypergraph Neural Network for Irregular Multivariate Time Series Forecasting. In *Forty-Second International Conference on Machine Learning*, June 2025.

Ilya Loshchilov and Frank Hutter. Decoupled weight decay regularization, 2019. URL https://arxiv.org/abs/1711.05101.

Yicheng Luo, Bowen Zhang, Zhen Liu, and Qianli Ma. Hi-Patch: Hierarchical Patch GNN for Irregular Multivariate Time Series. In *Forty-Second International Conference on Machine Learning*, June 2025.

Étienne Marcotte, Valentina Zantedeschi, Alexandre Drouin, and Nicolas Chapados. Regions of reliability in the evaluation of multivariate probabilistic forecasts. In *Proceedings of the 40th International Conference on Machine Learning*, volume 202 of *Proceedings of Machine Learning Research*. PMLR, 2023.

Mazen Melibari, Pascal Poupart, Prashant Doshi, and George Trimponias. Dynamic sum product networks for tractable inference on sequence data. In *Proceedings of the Eighth International Conference on Probabilistic Graphical Models*, volume 52 of *Proceedings of Machine Learning Research*, pages 345–355. PMLR, 2016.

M. J. Menne, Jr Williams, and R. S. Vose. U.S. HISTORICAL CLIMATOLOGY NETWORK (USHCN): Daily Temperature, Precipitation and Snow Data. Technical report, Environmental System Science Data Infrastructure for a Virtual Ecosystem (ESS-DIVE) (United States); Carbon Dioxide Information Analysis Center (CDIAC), Oak Ridge National Laboratory (ORNL), Oak Ridge, TN (United States), January 2006.

Kevin P. Murphy. *Probabilistic Machine Learning: An Introduction*. Adaptive Computation and Machine Learning Series. MIT Press, Cambridge, MA, USA, March 2022. ISBN 978-0-262-04682-4.

Roger B. Nelsen. *An Introduction to Copulas*. Springer Series in Statistics. Springer, New York, NY, 2006. ISBN 978-0-387-28659-4. doi: 10.1007/0-387-28678-0.

Bernt Øksendal. *Stochastic Differential Equations*. Universitext. Springer, Berlin, Heidelberg, 2003. ISBN 978-3-540-04758-2 978-3-642-14394-6. doi: 10.1007/978-3-642-14394-6.

Hoifung Poon and Pedro Domingos. Sum-product networks: A new deep architecture. In *Proceedings of the Twenty-Seventh Conference on Uncertainty in Artificial Intelligence*, UAI'11, pages 337–346, Arlington, Virginia, USA, July 2011. AUAI Press. ISBN 978-0-9749039-7-2.

L.R. Rabiner. A tutorial on hidden Markov models and selected applications in speech recognition. *Proceedings of the IEEE*, 77(2):257–286, February 1989. ISSN 1558-2256. doi: 10.1109/5.18626.

Yulia Rubanova, Ricky T. Q. Chen, and David K Duvenaud. Latent ordinary differential equations for irregularly-sampled time series. In *Advances in Neural Information Processing Systems*, volume 32. Curran Associates, Inc., 2019.

Simo Särkkä and Ángel F. García-Fernández. Temporal Parallelization of Bayesian Smoothers. *IEEE Transactions on Automatic Control*, 66(1):299–306, January 2021. ISSN 1558-2523. doi: 10.1109/TAC.2020.2976316.

Mona Schirmer, Mazin Eltayeb, Stefan Lessmann, and Maja Rudolph. Modeling Irregular Time Series with Continuous Recurrent Units. In *Proceedings of the 39th International Conference on Machine Learning*, pages 19388–19405. PMLR, June 2022.

Xiaoting Shao, Alejandro Molina, Antonio Vergari, Karl Stelzner, Robert Peharz, Thomas Liebig, and Kristian Kersting. Conditional sum-product networks: Imposing structure on deep probabilistic architectures. In *Proceedings of the Tenth International Conference on Probabilistic Graphical Models*, volume 138 of *Proceedings of Machine Learning Research*, pages 401–412. PMLR, 2020.

Ikaro Silva, George Moody, Daniel J Scott, Leo A Celi, and Roger G Mark. Predicting in-hospital mortality of ICU patients: The PhysioNet/Computing in cardiology challenge 2012. In *2012 Computing in Cardiology*, pages 245–248, September 2012.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Ł ukasz Kaiser, and Illia Polosukhin. Attention is All you Need. In *Advances in Neural Information Processing Systems*, volume 30. Curran Associates, Inc., 2017.

Vedrana Vidulin, Mitja Luštrek, Boštjan Kaluža, Rok Piltaver, and Jana Krivec. Localization data for person activity. UCI Machine Learning Repository, 2010.

Vijaya Krishna Yalavarthi, Johannes Burchert, and Lars Schmidt-Thieme. Tripletformer for Probabilistic Interpolation of Irregularly sampled Time Series. In *2023 IEEE International Conference on Big Data (BigData)*, pages 986–995, December 2023. doi: 10.1109/BigData59044.2023.10386325.

Vijaya Krishna Yalavarthi, Kiran Madhusudhanan, Randolf Scholz, Nourhan Ahmed, Johannes Burchert, Shayan Jawed, Stefan Born, and Lars Schmidt-Thieme. GraFITi: Graphs for Forecasting Irregularly Sampled Time Series. *Proceedings of the AAAI Conference on Artificial Intelligence*, 38(15):16255–16263, March 2024. ISSN 2374-3468. doi: 10.1609/aaai.v38i15.29560.

Vijaya Krishna Yalavarthi, Randolf Scholz, Stefan Born, and Lars Schmidt-Thieme. Probabilistic Forecasting of Irregularly Sampled Time Series with Missing Values via Conditional Normalizing Flows. *Proceedings of the AAAI Conference on Artificial Intelligence*, 39(20):21877–21885, April 2025a. ISSN 2374-3468. doi: 10.1609/aaai.v39i20.35494.

Vijaya Krishna Yalavarthi, Randolf Scholz, Christian Klötergens, Kiran Madhusudhanan, Stefan Born, and Lars Schmidt-Thieme. Reliable Probabilistic Forecasting of Irregular Time Series through Marginalization-Consistent Flows. In *The Fourteenth International Conference on Learning Representations*, October 2025b.

Weijia Zhang, Chenlong Yin, Hao Liu, Xiaofang Zhou, and Hui Xiong. Irregular Multivariate Time Series Forecasting: A Transformable Patching Graph Neural Networks Approach. In *Forty-First International Conference on Machine Learning*, June 2024.

Appendix Contents

A Notation **15**

B Proofs **16**

B.1 Marginalization Consistency up to the Leaves of CircuITS 16

B.2 CircuITS' Leaves are Marginalization Consistent 17

B.3 Proof of CircuITS' Marginalization Consistency 17

B.4 On the Necessity of Query-Independent Sum Weights 18

C Expressivity of the Recursive Projection **19**

D Model Details **20**

D.1 Encoder Architecture Diagram 20

D.2 Sampling with CircuITS' Sum Product Network 20

D.3 Parallelizing the Circuit Computation 21

E Details on the Synthetic Bifurcation Experiment **22**

F Dataset Descriptions **23**

G Hyperparameters **24**

H Details on Ablation Studies **25**

I Additional Metrics **26**

I.1 Energy Score and CRPS 26

I.2 Marginal NLL on the Main Benchmark 27

I.3 MSE of the Predicted Means 28

J Additional Baselines **29**

K Importance of Channel Ordering **30**

L Quantifying the Effect of Query-Dependent SPN Weights **31**

M Sensitivity Analysis for the Number of Components **32**

N Memory Consumption and Epoch Times **33**

O Broader Impacts **34**

A Notation

Table 3: Notation used in this work. Lowercase bold denotes vectors ($\mathbf{y}$, $\mathbf{e}_n$), uppercase bold denotes matrices ($\mathbf{R}$, $\mathbf{W}$), calligraphic letters denote sets or sequences ($\mathcal{X}$, $\mathcal{Q}$), $[n]$ abbreviates $\{1, \dots, n\}$, and a superscript in parentheses indexes the recursion step of the circuit ($\phi^{(c)}$) rather than a power.

Symbol	Domain / Shape	Meaning
Data, task, and target distribution (Section 3)		
C, L, N	$\mathbb{N}$	Number of channels, observations in $\mathcal{X}$, and query points
N_c	$\mathbb{N}$	Number of query points in channel c; $\sum_c N_c = N$
$\mathcal{X}$	$(\mathbb{R} \times [C] \times \mathbb{R})^L$	History $((t_n, c_n, y_n))_{n=1}^L$ of (time, channel, value) triplets
$\mathcal{Q}$	$(\mathbb{R} \times [C])^N$	Query $((t_n^{\text{qry}}, c_n^{\text{qry}}))_{n=1}^N$ of (time, channel) pairs
$\mathcal{I}_c$	$\subseteq [N]$	Index set of the query points of channel c: $\{n \mid c_n^{\text{qry}} = c\}$
$\mathbf{y}$, $\mathbf{y}_c$, $\mathbf{y}_{1:c}$	—	Targets aligned with $\mathcal{Q}$; of channel c; of channels $1, \dots, c$
$\mathbf{y}_S$, $\mathbf{y}_{-S}$	—	Targets that are marginalized out / retained, $S \subseteq [N]$
$p(\mathbf{y} \mid \mathcal{Q}, \mathcal{X})$	$\mathbb{R}^N \to \mathbb{R}_{\geq 0}$	Conditional joint density modeled by CircuITS
$sc(\cdot)$	—	Scope of a circuit node: the subset of targets it is a density over
Indices and hyperparameters		
n, c	$[L]$/$[N]$, $[C]$	Observation / query index; channel index (= recursion step)
k; i, j	$[K]$	Leaf index; component of state $\phi^{(c-1)}$ / of new leaf $p_{c,\cdot}$
m	$[K^2]$	Flattened index of the product node formed by the pair (i, j)
K	$\mathbb{N}$	Number of leaves per channel; $K \in \{2, 3, 4\}$ in our experiments
D, D'	$\mathbb{N}$	Embedding dimension; dimension of one component slice, $D = KD'$
D^{CORR}	$\mathbb{N}$	Dimension of the correlation features (Eq. (8))
L^{DSF}	$\mathbb{N}$	Number of layers of the Deep Sigmoidal Flow (DSF), indexed by l
Encoder (Section 4.1); all quantities depend on $\mathcal{X}$ only, except $\mathbf{e}_n$, which also depends on query point n		
$\phi^{\text{TIME}}(\cdot)$	$\mathbb{R} \to \mathbb{R}^D$	Sinusoidal (Time2Vec) embedding of a continuous timestamp
$\mathbf{E}^{\text{chan}}$, $\mathbf{E}^{\text{proto}}$, $\mathbf{E}^{\text{cw}}$	$\mathbb{R}^{C \times D}$	Learnable channel embeddings, prototypes, and circuit-weight queries
$\mathbf{H}^{\text{obs}}$	$\mathbb{R}^{L \times D}$	Embeddings of the L observation triplets
$\mathbf{M}$	$\{0, 1\}^{C \times L}$	Attention mask; $M_{c,n} = 1$ iff observation n belongs to channel c
$\mathbf{H}^{\text{chan}}$	$\mathbb{R}^{C \times D}$	Per-channel summary of the observations
$\tilde{\mathbf{H}}$, $\tilde{\mathbf{h}}_c$	$\mathbb{R}^{C \times D}$, $\mathbb{R}^D$	Channel states after inter-channel self-attention; row c of $\tilde{\mathbf{H}}$
$\mathbf{e}_n$, $\mathbf{e}_n^k$	$\mathbb{R}^D$, $\mathbb{R}^{D'}$	Embedding of query point n; its k-th slice (parameterizes leaf k)
Lin^{QRY}, Lin^{W}, Lin^{ROOT}	—	Linear projections producing $\mathbf{e}_n$, $\mathbf{W}^{(c)}$, and $\mathbf{w}^{\text{root}}$
Leaves (Section 4.2); each $p_{c,k}$ is a density over all N_c query points of channel c		
$p_{c,k}$	$\mathbb{R}^{N_c} \to \mathbb{R}_{\geq 0}$	k-th candidate density (leaf) of channel c; models one plausible regime
$F(\cdot\,; \theta_i^k)$	$\mathbb{R} \to (0, 1)$	DSF approximating the CDF of query point i under component k
θ_i^k	—	DSF parameters $\{A_l, b_l\}_l$, layers $f_l(x) = \sigma(A_l x + b_l)$, $A_l > 0$
u_i, $\mathbf{z}_i$	$(0, 1)$, $\mathbb{R}^{N_c}$	PIT $u_i = F(y_i; \theta_i^k)$; normal scores $\mathbf{z}_i = \Phi^{-1}(\mathbf{u}_i)$
$\mathbf{v}_i^{c,k}$	$\mathbb{R}^{D^{\text{CORR}}}$	Correlation features of query point i, $\tanh(\text{MLP}_{\text{corr}}(\mathbf{e}_i^k))$
$\dot{R}^{c,k}$, $\mathbf{R}^{c,k}$	$\mathbb{R}^{N_c \times N_c}$	Raw Gram matrix and the resulting correlation matrix (unit diagonal)
$\mathcal{L}^{\text{GC}}$	$\mathbb{R}$	Log-density of the Gaussian copula
Recursive sum-product circuit and objective (Section 4.3); (c) denotes the state after absorbing channels $1, \dots, c$		
$\phi_k^{(c)}$	$\mathbb{R}^{N_{1:c}} \to \mathbb{R}_{\geq 0}$	k-th component density of the joint state over channels $1{:}c$
$\mathbf{\Phi}^{(c)}$	K densities	Vector $[\phi_1^{(c)}, \dots, \phi_K^{(c)}]^\top$ of the current joint state
ψ_m	—	Product node $m = (i, j)$: $\phi_i^{(c-1)} \cdot p_{c,j}$
$\mathbf{W}^{(c)}$, $\mathbf{w}^{\text{root}}$	$\mathbb{R}^{K^2 \times K}$, $\mathbb{R}^K$	Column-stochastic sum weights at step c; root weights
$\mathbf{Z}$, $\mathbf{Z}_c$	$\mathbb{R}^{C \times D}$, $\mathbb{R}^D$	Attention output from which all circuit weights are projected
$M^{(c)}$	$\mathbb{R}^{K \times K}$	Log-semiring transition matrix of the parallel scan (Appendix D.3)
k_c	$[K]$	Active component of channel c in sampling (Appendix D.2)
njNLL	$\mathbb{R}$	Training objective $-\log p(\mathbf{y} \mid \mathcal{Q}, \mathcal{X})/N$

B Proofs

B.1 Marginalization Consistency up to the Leaves of CircuITS

Lemma B.1. *An SPN is a marginalization consistent estimator up to the leaf nodes:*

$$\int p(\mathbf{y} \mid \mathcal{Q}, \mathcal{X}) d\mathbf{y}_c = p(\mathbf{y}_{-c} \mid \mathcal{Q}_{-c}, \mathcal{X}),$$

where $\mathbf{y}_{-c} = \mathbf{y} \setminus \mathbf{y}_c$*, if it fulfills:*

- ***Property 1:*** *Each leaf node in the SPN represents the distribution of variables corresponding to a single input channel*
- ***Property 2:*** *The marginal leaf distributions only depend on the query subset ($\mathcal{Q}_c$) that corresponds to the respective channel*
- ***Property 3:*** *The weights of the sum nodes do not depend on the query set $\mathcal{Q}$*

Proof of Lemma B.1. Let n be a node in the SPN representing the distribution $p_n(\mathbf{y}_{sc(n)} \mid \mathcal{Q}, \mathcal{X})$. We want to prove that for any channel $y_c \subseteq sc(n)$:

$$\int p_n(\mathbf{y}_{sc(n)} \mid \mathcal{Q}, \mathcal{X})\, d\mathbf{y}_c = p_n(\mathbf{y}_{sc(n)\setminus c} \mid \mathcal{Q}_{-c}, \mathcal{X})$$

If n is a Leaf Node. If n is a leaf node $L_c^{(i)}$, its scope is $\mathbf{y}_c$ (Property 1). By the definition of a probability distribution, integrating over its entire domain yields 1:

$$\int p_L^{(i)}(\mathbf{y}_c \mid \mathcal{Q}_c, \mathcal{X})\, d\mathbf{y}_c = 1$$

This echoes the principle of SPNs that the marginalization of a node is by replacing its distribution with 1. The resulting 1 is the identity element for product nodes, effectively *removing* the channel from the joint factorization.

If $sc(n) \cap \mathbf{y}_c = \emptyset$, the leaf is independent of the marginalization variable, and is therefore not affected.

If n is a Product Node. By decomposability, y_c exists in the scope of exactly one child, say child k'.

$$p_n(\mathbf{y}_{sc(n)} \mid \mathcal{Q}, \mathcal{X}) = p_{k'}(\mathbf{y}_{sc(k')} \mid \mathcal{Q}, \mathcal{X}) \cdot \prod_{k \neq k'} p_k(\mathbf{y}_{sc(k)} \mid \mathcal{Q}, \mathcal{X})$$

Integrating over $\mathbf{y}_c$:

$$\int \left(p_{k'}(\mathbf{y}_{c(k')} \mid \mathcal{Q}, \mathcal{X}) \prod_{k \neq k'} p_k(\mathbf{y}_{sc(k)} \mid \mathcal{Q}, \mathcal{X}) \right) d\mathbf{y}_c$$
$$= \left(\int p_{k'}(\mathbf{y}_{sc(k')} \mid \mathcal{Q}, \mathcal{X}) d\mathbf{y}_c \right) \prod_{k \neq k'} p_k(\mathbf{y}_{sc(k)} \mid \mathcal{Q}, \mathcal{X})$$

If $sc(k') = \mathbf{y}_c$, the integral is 1, effectively removing channel c from the product.

If n is a Sum Node. By smoothness of the SPN [Choi et al., 2020], all children share the same scope. By Property 3, weights $w_{n,k}$ are independent of $\mathbf{y}$ and $\mathcal{Q}$.

$$\int \left(\sum_{k \in ch(n)} w_{n,k}(\mathcal{X}) \cdot p_k(\mathbf{y}_{sc(n)} \mid \dots) \right) d\mathbf{y}_c = \sum_{k \in ch(n)} w_{n,k}(\mathcal{X}) \int p_k(\mathbf{y}_{sc(n)} \mid \dots) d\mathbf{y}_c$$

Each integral $\int p_k(\mathbf{y}_{sc(n)}) d\mathbf{y}_c$ yields the marginal $p_k(\mathbf{y}_{sc(n)\setminus c})$. Since the weights $w_{n,k}(\mathcal{X})$ remain unchanged, the node n now represents a valid mixture of the marginalized distributions of its children.

Since the property holds for the leaves and is preserved by both product and sum operations, it holds for the root of the circuit. By repeating this process for any number of channels $c \in \{1, \dots, C\}$, we prove that the SPN is marginalization consistent across all channel-level subsets. □

B.2 CircuITS' Leaves are Marginalization Consistent

Proposition B.2 (Marginalization Consistency of CircuITS' Leaf Distributions)**.** *Let $\mathcal{Q}_c \subseteq \mathcal{Q}$ be the subset of forecasting queries belonging to a specific channel c, and let $\mathbf{y}_c$ be the vector of target variables corresponding to $\mathcal{Q}_c$.*

Consider an arbitrary query index n within this channel's set $\mathcal{Q}_c$. Let $\mathcal{Q}_{c\setminus n}$ denote the query set with the n-th query point removed, and $\mathbf{y}_{c\setminus n}$ denote the target vector without y_n. The leaf distribution for channel c is marginalization consistent if:

$$\int p(\mathbf{y}_c \mid \mathcal{Q}_c, \mathcal{X})\, dy_n = p(\mathbf{y}_{c\setminus n} \mid \mathcal{Q}_{c\setminus n}, \mathcal{X})$$

By induction, satisfying this condition implies consistency for any subset of variables within the channel.

Proof. The leaf distribution for a channel c and mixture component k is defined as a Gaussian Copula combined with univariate Deep Sigmoidal Flows (DSF) [Nelsen, 2006]. It is a standard property of the Gaussian Copula that marginalizing out a single variable y_n yields a Gaussian Copula over the remaining variables $\mathbf{y}_{c\setminus n}$, parameterized by the correlation matrix $R^{c,k}$ with the n-th row and column removed.

Therefore, validity relies on **parameter stability**: the model parameters for the remaining queries $\mathcal{Q}_{c\setminus n}$ must not change when the query point q_n is removed. Specifically, for any two remaining indices $i, j \neq n$, the correlation entry R_{ij} must remain identical.

In CircuITS, the correlation matrix entry for channel c and component k is computed as:

$$R_{ij} = \frac{\mathbf{v}_i^\top \mathbf{v}_j}{D^{\text{CORR}}} + \delta_{ij}(1 - \dots),$$

where $\mathbf{v}_i = \tanh(\text{MLP}(e_i))$. The context embedding e_i is derived via $e_i = \text{Lin}^{\text{QRY}}([\phi^{\text{TIME}}(t_i^{\text{qry}}) \,\|\, \tilde{h}_c])$ (see Equations (8) and (9)).

Crucially, the channel context $\tilde{h}_c$ is computed solely from the observation history $\mathcal{X}$ via the set-based encoder and is independent of the query set $\mathcal{Q}_c$. Thus, e_i and $\mathbf{v}_i$ depend strictly on the specific query point $q_i = (t_i^{\text{qry}}, c)$ and $\mathcal{X}$. Since the calculation of R_{ij} for $i, j \neq n$ does not involve the query point q_n, removing q_n simply corresponds to removing the n-th row and column from $R^{c,k}$. This matches the theoretical marginalization requirement of the Gaussian Copula. □

B.3 Proof of CircuITS' Marginalization Consistency

CircuITS is a marginalization consistent model for conditional joint density estimation of irregular multivariate time series forecasting. Formally, for any subset of variables $\mathbf{y}_A \subset \mathbf{y}$:

$$\int p(\mathbf{y} \mid \mathcal{Q}, \mathcal{X})\, d\mathbf{y}_A = p(\mathbf{y} \setminus \mathbf{y}_A \mid \mathcal{Q} \setminus \mathcal{Q}_A, \mathcal{X})$$

Proof. Let S be an arbitrary set of indices, and let $S_c = S \cap \mathcal{I}_c$ denote the indices belonging to channel c, such that $S = \bigcup_{c=1}^{C} S_c$.

From Lemma B.1, we have established that the circuit structure is marginalization-consistent up to the leaf nodes. The integral operator distributes through the sum and product nodes such that the global integral is determined by the local integrals at the leaves:

$$\int p(\mathbf{y} \mid \mathcal{Q}, \mathcal{X}) d\mathbf{y}_{S_c} = \text{SPN}\left(\dots, \int p_L^{(i)}(\mathbf{y}_c \mid \mathcal{Q}_c, \mathcal{X}) d\mathbf{y}_{S_c}, \dots\right)$$

where $\text{SPN}(\cdot)$ represents the original circuit topology with fixed weights $w_{n,k}(\mathcal{X})$ (Property 3 from Lemma B.1).

By Proposition B.2, each leaf node is itself marginalization consistent over its subset S_c:

$$\int p_L^{(i)}(\mathbf{y}_c \mid \mathcal{Q}_c, \mathcal{X}) d\mathbf{y}_{S_c} = p_L^{(i)}(\mathbf{y}_{c\setminus S_c} \mid \mathcal{Q}_{c\setminus S_c}, \mathcal{X})$$

Substituting this into the circuit, the leaf nodes are replaced by their corresponding marginal distributions over the remaining queries $\mathcal{Q}_{c\setminus S_c}$. By repeating this process for all c, the entire circuit represents the valid marginal distribution $p(\mathbf{y}_{-S} \mid \mathcal{Q}_{-S}, \mathcal{X})$. □

B.4 On the Necessity of Query-Independent Sum Weights

Lemma B.1 states Property 3 as a sufficient condition. Here we discuss to what extent it is also necessary. Expanding the recursion, the joint density of CircuITS is a mixture over the K^C configurations of leaf components (see Appendix C),

$$p(\mathbf{y} \mid \mathcal{Q}, \mathcal{X}) = \sum_{j_1,\dots,j_C} \lambda_{j_1 \dots j_C} \prod_{c=1}^{C} p_{c,j_c}(\mathbf{y}_c),$$

with effective weights $\lambda_{j_1 \dots j_C}$ obtained by multiplying and summing the sum weights along the SPN. Since leaves of different channels have disjoint scopes, the products $\prod_c p_{c,j_c}$ are linearly independent whenever the K leaves of each channel are. Now drop a single query point n from a channel. The set of channels is unchanged and the leaves are marginalization consistent (Proposition B.2), so the marginalized joint and the prediction for $\mathcal{Q}_{-n}$ are mixtures over the same product functions, with weights $\lambda(\mathcal{Q})$ and $\lambda(\mathcal{Q}_{-n})$, respectively. By linear independence, consistency then forces $\lambda(\mathcal{Q}) = \lambda(\mathcal{Q}_{-n})$ exactly. Iterating this argument, the *effective* weights must be independent of the query.

What is not necessary is that each individual sum weight $w_{m,k}^{(c)}$ is query-independent. The map from sum weights to effective weights is many-to-one, so individual weights could in principle vary with $\mathcal{Q}$ while the effective weights $\lambda(\mathcal{Q})$ stay fixed. This is why we state Property 3 as a sufficient rather than a necessary condition. Such a variation, however, leaves every predicted density for every query unchanged: it is merely a reparameterization of the same circuit and therefore adds no expressivity. Moreover, guaranteeing that $\lambda(\mathcal{Q})$ remains constant while the individual weights vary with $\mathcal{Q}$ would be very difficult to enforce architecturally. We quantify the inconsistency that arises from query-dependent sum weights empirically in Appendix L.

C Expressivity of the Recursive Projection

In every recursive step (Section 4.3), the K^2 product nodes are projected back onto K sum nodes. Here we characterize precisely what this projection restricts.

What the projection does. Unrolling the recursion in (12)–(13) gives the closed form

$$p(\mathbf{y} \mid \mathcal{Q}, \mathcal{X}) = \sum_{k_1,\ldots,k_C} A_{k_1\ldots k_C} \prod_{c=1}^{C} p_{c,k_c}(\mathbf{y}_c), \qquad A_{k_1\ldots k_C} = \mathbf{e}_{k_1}^{\top}\, \mathbf{G}^{(2)}[k_2] \cdots \mathbf{G}^{(C)}[k_C]\, \mathbf{w}^{\text{root}}, \tag{18}$$

where $\mathbf{e}_{k_1}$ is the k_1-th unit vector and the per-channel factor $\mathbf{G}^{(c)}[j] \in \mathbb{R}^{K\times K}$ is the matrix whose entry in row i and column r equals the entry $W^{(c)}_{(i,j),r}$ of the sum-layer weights. Hence, CircuITS *is* a mixture over all K^C combinations of channel leaves. No product term is dropped, and every product term keeps a strictly positive weight, since all weights are obtained by a softmax (Equation (17)). What the projection restricts is the *weight tensor* A, which is forced into a nonnegative tensor-train (matrix-product-state) factorization of rank K. Equivalently, this is the hidden Markov model over the channel order described in Appendix D.3, with the K sum nodes as latent states, the channel leaves as emissions, and $\mathbf{W}^{(c)}$ as the joint transition-emission matrix, so that the recursion is exactly its forward algorithm.

Intuition. The K-dimensional message that channels $1,\ldots,c$ pass to channels $c{+}1,\ldots,C$ is $\mathbf{\Phi}^{(c)}$. Conditional on which of the K sum nodes is active at level c, the channels up to c and the channels after c are independent. Each sum node thus acts as a reusable *regime* summarizing the channels seen so far, and $\mathbf{W}^{(c)}$ specifies how leaf j of a new channel pairs with regime i to form regime k. The projection is what buys tractable evaluation and marginalization in $\mathcal{O}(CK^3)$ and keeps the network decomposable and smooth.

What is lost. Any information about channels $1,\ldots,c$ beyond *which of the K regimes is active* cannot influence the remaining channels. Formally, let $A^{(c)} \in \mathbb{R}^{K^c\times K^{C-c}}$ denote the matricization of A that groups $(k_1,\ldots,k_c)$ into rows and $(k_{c+1},\ldots,k_C)$ into columns. By (18), $A^{(c)} = LR$ with nonnegative factors $L \in \mathbb{R}^{K^c\times K}_{\geq 0}$ and $R \in \mathbb{R}^{K\times K^{C-c}}_{\geq 0}$, hence the nonnegative rank of $A^{(c)}$ is at most K, whereas an unconstrained weight tensor admits a nonnegative rank of up to $\min(K^c, K^{C-c})$. This yields the following statements:

- For $C \leq 3$, the projection loses nothing: every weight tensor A is exactly representable.
- For $C \geq 4$, it is a *strict* restriction, first binding at the cut $c = 2$. Exactly representing an arbitrary A would require the number of sum nodes per layer, i.e. the inner dimension of the factorization, to grow to $K^{\lfloor C/2\rfloor}$.

Per-instance restriction. Since the weights $\mathbf{W}^{(c)}$ and $\mathbf{w}^{\text{root}}$ are predicted per instance from $\mathcal{X}$ (Equation (17)), the rank-K constraint binds each *predicted* joint density individually, not the model class as a whole: different instances (e.g. different patients) realize different rank-K weight tensors.

Does the restriction matter in practice? Increasing K raises the tensor-train rank together with the number of leaves per channel and would therefore relax the restriction. Our sensitivity analysis (Appendix M) nevertheless finds $K = 2$ or $K = 3$ to be optimal, outperforming $K = 4$. This suggests that the rank constraint is not the binding limitation on the evaluated datasets.

D Model Details

This section provides additional details on CircuITS: a diagram of the encoder, the sampling procedure of the SPN, and the parallelized computation of the circuit.

D.1 Encoder Architecture Diagram

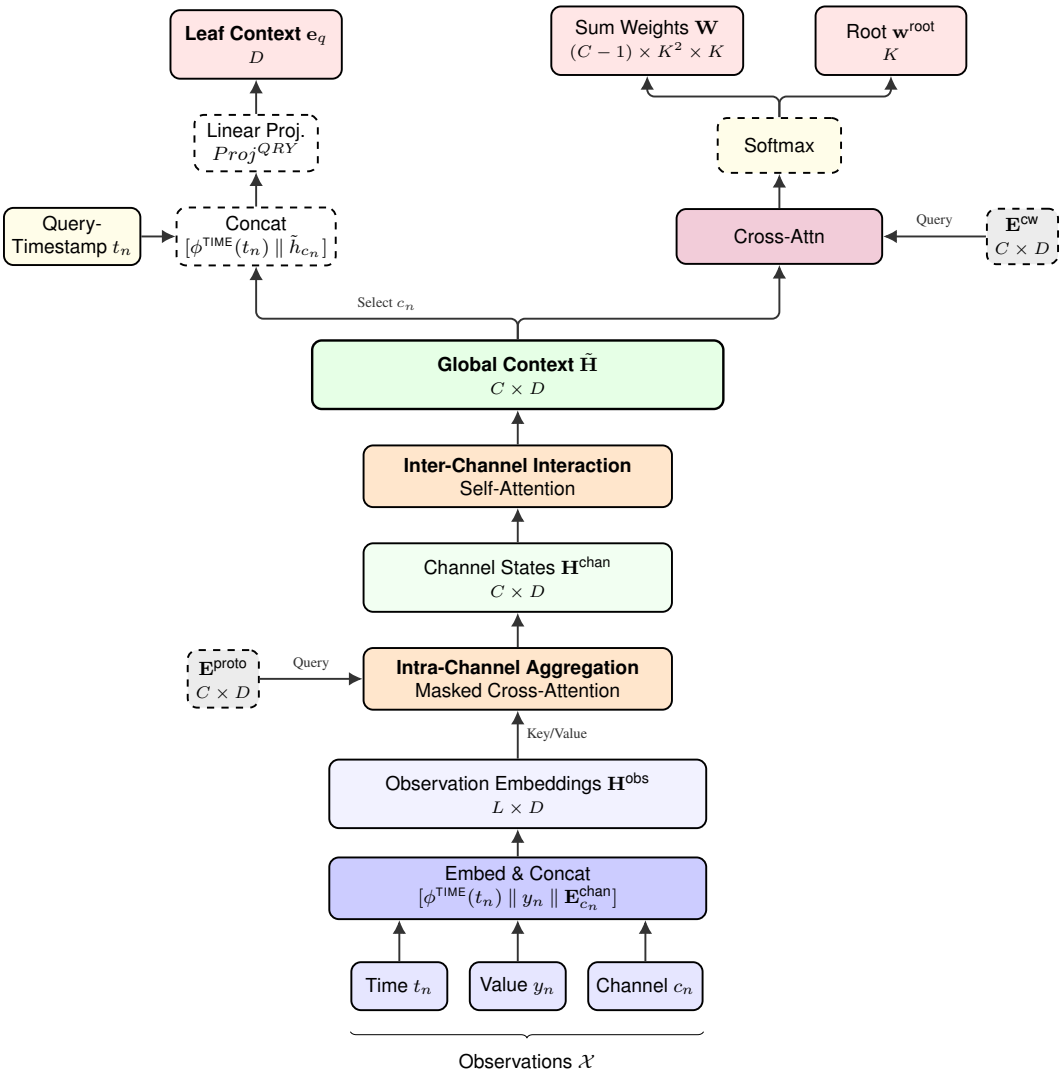

Figure 3: Detailed architecture of the CircuITS Encoder with tensor dimensions. The global context $\tilde{\mathbf{H}}$ (dim $C \times D$) branches to generate leaf contexts (dim D) and circuit weights.

D.2 Sampling with CircuITS' Sum Product Network

Let $k_c \in \{1, \ldots, K\}$ denote the index of the active sum node (latent component) for channel c. The sampling proceeds as follows:

Root Initialization ($c = C$): We begin by sampling the final latent state k_C directly from the distribution defined by the final weights:

$$k_C \sim \text{Categorical}(\mathbf{w}^{\text{root}}) \tag{19}$$

Recursive Backtracking ($c = C, \ldots, 2$): Given the current active state $k_c = k$, we must determine the branch (product node) that generated it. The sum node k aggregates over K^2 product terms

formed by the previous channel state and the current channel leaf. We sample the flattened product index $m \in \{1, \dots, K^2\}$ using the mixing weights $W^{(c)}$ specific to the active node k:

$$m \sim \text{Categorical}(W^{(c)}_{\cdot,k}) \tag{20}$$

The sampled index m uniquely maps to a pair of indices: the latent state of the previous channel k_{c-1} and the specific mixture component j_c for the current channel's leaf distribution. We update the active state for the next step (k_{c-1}) and record the active leaf component index j_c for the current channel.

Termination ($c = 1$): At the final step of the backward pass, the remaining active state k_1 directly indicates the active leaf component j_1 for the first channel.

D.3 Parallelizing the Circuit Computation

The recursive Sum-Product Network (SPN) aggregation described in Section 4.3 inherently suggests a sequential computation: the joint state $\phi^{(c)}$ depends on $\phi^{(c-1)}$, requiring $O(C)$ sequential steps for C channels. However, by analyzing the algebraic structure of the update rule, we can reformulate the recurrence as a sequence of matrix multiplications in the log-semiring. This reformulation allows us to utilize parallel prefix scan algorithms (associative scans), reducing the time complexity from linear $O(C)$ to logarithmic $O(\log C)$ on parallel hardware (e.g., GPUs).

Connection to HMMs. The recursive update in CircuITS is isomorphic to the Forward Algorithm in a Hidden Markov Model [Rabiner, 1989] (HMM) with a time-varying transition matrix. Recall the update rule for the circuit state. Let $\log \phi^{(c-1)} \in \mathbb{R}^K$ be the log-density vector of the joint state after $c-1$ channels. The update for the k-th component of the next channel c is given by aggregating over the previous states i and the current leaf mixture components j:

$$\log \phi^{(c)}_k = \log \sum_{i=1}^{K} \sum_{j=1}^{K} \exp\left(\log \phi^{(c-1)}_i + L_{c,j} + \log W^{(c)}_{(i,j),k}\right)$$

where $L_{c,j}$ is the log-likelihood of the j-th leaf component for channel c, and $W^{(c)}$ describes the mixing weights. We can rearrange the summation to isolate the dependence on the previous state $\phi^{(c-1)}_i$:

$$\log \phi^{(c)}_k = \log \sum_{i=1}^{K} \exp\left(\log \phi^{(c-1)}_i + \underbrace{\log \sum_{j=1}^{K} \exp\left(L_{c,j} + \log W^{(c)}_{(i,j),k}\right)}_{M^{(c)}_{ik}}\right)$$

Here, we define a transition matrix $M^{(c)} \in \mathbb{R}^{K \times K}$ for step c. The entry $M^{(c)}_{ik}$ represents the log-probability of transitioning from joint component i to component k, marginalizing over the leaf distributions of the current channel.

In vector notation, this becomes a matrix-vector multiplication in the log-semiring (where $(\oplus, \otimes) \to (\text{logsumexp}, +)$): $\phi^{(c)} = \phi^{(c-1)} \otimes_{\log} M^{(c)}$.

Consequently, the final joint distribution weights $\phi^{(C)}$ can be expressed as the product of the initial state $\phi^{(1)}$ and a sequence of transition matrices:

$$\phi^{(C)} = \phi^{(1)} \otimes_{\log} M^{(2)} \otimes_{\log} M^{(3)} \cdots \otimes_{\log} M^{(C)}$$

This formulation highlights that CircuITS, while structurally an SPN, shares the computational characteristics of linear recurrent models and HMMs [Särkkä and García-Fernández, 2021].

Parallel Log-Semiring Scan. Since matrix multiplication is associative, the cumulative product of the transition matrices $M^{(c)}$ can be computed in parallel using the Hillis-Steele algorithm [Hillis and Steele, 1986]. Instead of computing the product sequentially $(((M^{(2)}M^{(3)})M^{(4)})\dots)$, a parallel scan computes the products in a tree-like structure. For a sequence of length C, this requires $\mathcal{O}(\log C)$ sequential steps with sufficient parallel threads.

E Details on the Synthetic Bifurcation Experiment

For the toy experiment described in Section 5, we generated IMTS trajectories using 50 evenly spaced time steps. The data points prior to bifurcation are randomly distributed around 0 with a standard deviation of 0.1. After the bifurcation point (at the 18th time step), the channels follow a Brownian motion with a drift of either 0.1 or -0.1 per time step. The standard deviation is still 0.1. This decision is random, independent per channel, and equiprobable for both paths.

The models are tasked with predicting the distribution of the final 75% of time steps, conditioned on the initial 25%. Consequently, the observations provided as input contain no information regarding the future drift direction. To simulate irregularity, we randomly dropped 5% of the observations and query points.

We present samples from ProFITi in Figure 4, along with the final normalized joint negative log-likelihood (njNLL) on the test split for ProFITi, MOSES, and CircuITS. While ProFITi correctly captures the Brownian motion dynamics and approaches CircuITS in modeling the joint likelihood (Table 4), it fails to learn the strict separation of modalities.

Table 4: Test njNLL for the synthetic four channel bifurcation task.

	MOSES-10	MOSES-16	ProFITi	CircuITS
njNLL	-0.340 ±.07	-0.290 ± .04	-1.253 ± .01	**-1.280 ± .02**

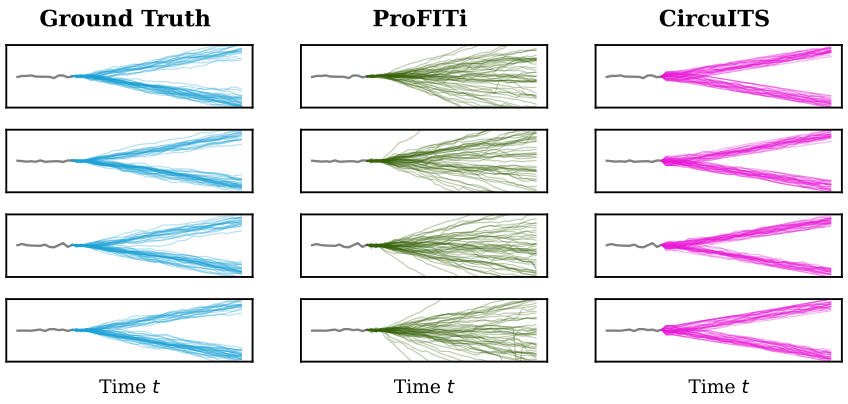

Figure 4: Comparing samples from ProFITi and CircuITS for the four channel bifurcation task.

F Dataset Descriptions

We utilize the following five benchmark datasets to evaluate performance, covering climatological, physiological, and human motion domains. The first four are used in coherence with Yalavarthi et al. [2025b]. The specific characteristics and preprocessing protocols for each are detailed below:

USHCN. Derived from the U.S. Historical Climatology Network [Menne et al., 2006], this dataset tracks climate changes across the USA. It consists of daily measurements of 5 variables (snow precipitation, rain precipitation, snow depth, min. temperature, and max temperature) collected over 150 years from 1,218 meteorological stations. Following the protocol of De Brouwer et al. [2019], we selected a subset of 1,100 stations and an observation window of 4 years (1996-2000). USHCN is transformed into an IMTS by randomly dropping 95% recorded observations.

PhysioNet2012. Sourced from the PhysioNet Computing in Cardiology Challenge 2012 [Silva et al., 2012], this physiological dataset contains medical records of 12,000 patients admitted to the ICU. It tracks 37 vital signs over a 48-hour period. Consistent with Che et al. [2018], the dataset is binned into hourly observations by rounding time stamps to the hour and averaging observations, if a channel is observed multiple times within one hour.

MIMIC-III. The Medical Information Mart for Intensive Care III [Johnson et al., 2016] is a large-scale physiological dataset collected at Beth Israel Deaconess Medical Center. It contains 17,874 clinical instances measuring 96 variables over 48 hours. Following De Brouwer et al. [2019], observations are aggregated and rounded to 30-minute intervals.

MIMIC-IV. This dataset features records from a tertiary academic medical center in Boston. It tracks 102 variables from ICU patients over a 48-hour period [Johnson et al., 2023a,b]. Adopting the standards of Biloš et al. [2021], the observations are rounded to 1-minute intervals. Due to the more fine-grained binning, this dataset records the greatest number of observations and queries.

Human Activity. This dataset [Kaluža et al., 2010, Vidulin et al., 2010] contains 3D position recordings of four body-worn tags (left ankle, right ankle, chest, and belt) from five people, each performing a scripted sequence of activities such as walking, sitting, and lying five times, which yields 25 recordings with 12 channels. Only one tag is read out at a time, so the resulting multivariate series is sporadically observed. Following Rubanova et al. [2019] and Zhang et al. [2024], timestamps are quantized to milliseconds and repeated readings of the same tag within one millisecond are averaged. Each recording is cut into non-overlapping windows of 4 seconds, which take the role of the 48 hours of the medical datasets: in the 36–12 setting, for instance, the models observe the first 3000 ms of a window and forecast the remaining 1000 ms. The splits are made on the level of recordings, so that no window of a test recording leaks into training.

Table 5: Summary of Dataset Characteristics and Preprocessing

Dataset	Samples	Channels	Duration	Binning	Max. Observations
USHCN	1,100 (Stations)	5	4 Years	1 Week	325
PhysioNet2012	12,000 (Patients)	37	48 Hours	1 Hour	552
MIMIC-III	21,000 (Patients)	96	48 Hours	30 Mins	741
MIMIC-IV	18,000 (Patients)	102	48 Hours	1 Min	1386
Human Activity	1,360 (Windows)	12	4 Seconds	1 ms	387

Table 6: Minimum, Average, and Maximum Number of Queries (Grouped by Task)

	36-3			36-12			24-24			12-36		
Dataset	min.	avg.	max.	min.	avg.	max.	min.	avg.	max.	min.	avg.	max.
USHCN	3	3.3	6	5	54.7	63	19	109.4	120	19	164.1	179
PhysioNet2012	1	19.8	53	1	61.8	148	1	139.9	297	1	220.7	459
MIMIC-III	1	10.3	85	1	25.3	264	1	67.2	512	1	109.7	785
MIMIC-IV	1	7.8	79	1	57.2	481	1	128.9	906	1	208.4	1357
Human Activity	3	9.0	9	3	90.3	99	33	180.2	195	21	269.9	291

G Hyperparameters

Each model is trained for a maximum of 2000 epochs using early stopping with patience of 30 epochs. We use the AdamW [Kingma and Ba, 2015, Loshchilov and Hutter, 2019] optimizer with an initial learning rate of 0.001, an L2 regularization of 0.001 and a batch size of 64. Additionally, we use a learning-decay-on-plateau scheduler with a decay rate of 0.5 and 5 epochs patience. For each model we evaluate a maximum of 10 randomly searched hyperparameter setting. We train each setting on single fold and select the setting with the best validation njNLL. We follow Yalavarthi et al. and select the following hyperparameter search spaces for the evaluated models:

MOSES. For the MOSES model [Yalavarthi et al., 2025b], we perform a search over the number of **mixture components** $\in \{1, 2, 5, 7, 10\}$. Additionally, we evaluate different sizes for the **hidden dimension** $\in \{16, 32, 64, 128\}$ and vary the number of **attention heads** $\in \{1, 2, 4\}$.

ProFITi. For ProFITi [Yalavarthi et al., 2025a], we evaluate the number of **flow layers** $\in \{7, 8, 9, 10\}$ and the **latent dimension size** $\in \{32, 64, 128, 256\}$.

Neural Flows. For the Neural Flows model [Biloš et al., 2021], we primarily search over the number of flow layers $\in \{1, 4\}$. We fix the **hidden dimension** to 64, and the number of **hidden layers** is 2, and the **flow model** is set to GRU.

CircuITS. For our model, we vary the number of **(components** K**)** $\in \{2, 3, 4\}$ and the **hidden dimension** $D' \in \{32, 64\}$. For the Deep sigmoidal flow on the univariate marginals we tune the number of **flow layers** $\in \{2, 3\}$, and the **flow hidden dimension** $\in \{10, 20\}$. The MLP that infers the parameters has 2 hidden layers and a hidden dimension of 32. The multi-head attention layers in the encoder are applied by setting the number of **attention heads** to 2. For CircuITS we had to set the batch size to 32 for the MIMIC-IV due to memory overflow.

The remaining hyperparameters of the baseline models are taken from the respective papers.

H Details on Ablation Studies

We provide additional information for the ablation studies presented in Table 2.

w/o SPN. We conduct this experiment by training a model where the number of leaf components K is set to 1. Consequently, there is no weighted sum of different components; instead, channels are combined via a single product distribution.

w/o DSF. We replace the Deep Sigmoidal Flow (DSF), which is applied to the univariate marginals, with normal distributions parameterized by (μ, σ) that are inferred from the embeddings of the forecasting queries. Formally:

$$\mu_i^k = f^{\text{mean}}(e_i^k) \tag{21}$$

$$\sigma_i^k = \exp(f^{\text{var}}(e_i^k)), \tag{22}$$

where f^{mean} and f^{var} are implemented as 2-layer MLPs. We do not change the computation of the correlation matrix R.

w/o GC. For this ablation, we set the correlation matrix $R = I$, thereby treating the univariate marginals of a single component k as independent.

w/ TF. We utilize Tripletformer (TF) [Yalavarthi et al., 2023] as an encoder to compute the circuit weights and the embeddings of the forecasting queries. In contrast to the encoder in CircuITS, Tripletformer does not group observations/queries by channel. Instead, it first performs self-attention among all observation triplets (channel, timestamp, value) and subsequently applies cross-attention from the encoded observation triplets to the forecasting queries.

To infer the weights of the SPN, we include an additional Cross-Attention layer with learnable circuit weight embeddings $\mathbf{E}^{\text{cw}}$, (similar to our encoder but using all encoded observation triplets as attention keys and values).

I Additional Metrics

I.1 Energy Score and CRPS

To evaluate the quality of samples produced by the learned distributions, we employ the Energy Score and the Continuous Ranked Probability Score (CRPS). Since closed-form solutions are often intractable for complex joint distributions modeled by CircuITS and the baselines, we approximate these metrics using Monte Carlo estimation with $S = 1000$ samples.

Energy Score. The Energy Score is a multivariate strictly proper scoring rule that assesses the quality of the joint distribution, capturing correlations between variables in the IMTS. For a ground truth vector $y \in \mathbb{R}^M$ corresponding to the query $\mathcal{Q}$, and S samples, the empirical Energy Score is defined as:

$$ES(y, \mathcal{Q}) = \frac{1}{S}\sum_{s=1}^{S} \|\hat{y}^{(s)} - y\|_2 - \frac{1}{2S^2}\sum_{s=1}^{S}\sum_{j=1}^{S} \|\hat{y}^{(s)} - \hat{y}^{(j)}\|_2 \tag{23}$$

where $\|\cdot\|_2$ denotes the Euclidean norm. The first term measures the distance between the samples and the ground truth (calibration), while the second term measures the diversity of the samples (sharpness).

Continuous Ranked Probability Score (CRPS). While the Energy Score evaluates the joint distribution, the CRPS evaluates the accuracy of the univariate marginal distributions. We report the mean CRPS averaged over all query targets in $\mathcal{Q}$. The sample-based approximation for the CRPS is given by:

$$CRPS(y, \mathcal{Q}) = \frac{1}{N}\sum_{i=1}^{N}\left(\frac{1}{S}\sum_{s=1}^{S}|\hat{y}_i^{(s)} - y_i| - \frac{1}{2S^2}\sum_{s=1}^{S}\sum_{j=1}^{S}|\hat{y}_i^{(s)} - \hat{y}_i^{(j)}|\right) \tag{24}$$

where y_i is the ground truth value for the i-th query target and $\hat{y}_i^{(s)}$ is the corresponding value in the s-th sample. Table 8 and Table 7 compare these scores for NeuralFlows, ProFITi, MOSES, and CircuITS.

Table 7: Energy Score of competing models. Lower values are better. Best results are in **bold**, second best in *italics*. Models are trained and tuned on njNLL.

	Model	USHCN	PhysioNet'12	MIMIC-III	MIMIC-IV	Activity
36–3*	NeuralFlows	0.661 ± 0.06	1.691 ± 0.00	*1.381 ± 0.03*	0.982 ± 0.01	3.801 ± 0.21
	ProFITi	**0.452 ± 0.04**	**0.879 ± 0.30**	1.606 ± 0.17	**0.808 ± 0.00**	**0.793 ± 0.06**
	MOSES	0.552 ± 0.04	*1.599 ± 0.01*	**1.353 ± 0.03**	*0.906 ± 0.03*	0.918 ± 0.11
	CircuITS	*0.468 ± 0.02*	1.613 ± 0.00	1.489 ± 0.04	0.927 ± 0.05	*0.796 ± 0.07*
36–12	NeuralFlows	5.943 ± 0.39	6.107 ± 0.18	8.107 ± 0.83	12.222 ± 0.39	12.225 ± 0.45
	ProFITi	*4.385 ± 1.7*	3.575 ± 0.45	2.546 ± 0.13	5.394 ± 0.15	*3.328 ± 0.15*
	MOSES	**3.252 ± 0.13**	**3.195 ± 0.04**	*2.358 ± 0.10*	*2.869 ± 0.04*	3.332 ± 0.09
	CircuITS	5.535 ± 0.93	*3.213 ± 0.00*	**2.279 ± 0.01**	**2.835 ± 0.09**	**3.174 ± 0.09**
24–24	NeuralFlows	*8.448 ± 0.66*	9.522 ± 0.44	10.290 ± 0.35	16.836 ± 0.71	15.425 ± 1.19
	ProFITi	14.42 ± 7.6	5.742 ± 0.28	4.287 ± 0.38	10.82 ± 1.1	6.171 ± 0.25
	MOSES	**5.330 ± 0.31**	*5.284 ± 0.03*	*3.632 ± 0.05*	**4.268 ± 0.07**	*5.207 ± 0.25*
	CircuITS	10.71 ± 1.9	**5.267 ± 0.01**	**3.543 ± 0.04**	*4.482 ± 0.15*	**5.191 ± 0.16**
12–36	NeuralFlows	**10.79 ± 1.1**	12.170 ± 0.41	14.524 ± 0.95	23.049 ± 0.97	18.529 ± 0.83
	ProFITi	41.61 ± 39	8.439 ± 0.42	6.567 ± 0.53	11.10 ± 0.91	9.274 ± 0.91
	MOSES	21.22 ± 17	*7.685 ± 0.63*	*5.770 ± 0.03*	*6.276 ± 0.03*	*6.980 ± 0.17*
	CircuITS	*12.18 ± 1.8*	**7.241 ± 0.01**	**5.730 ± 0.06**	**5.851 ± 0.05**	**6.808 ± 0.23**

Table 8: CRPS of competing models. Lower values are better. Best results are in **bold**, second best in *italics*. Models are trained and tuned on njNLL.

	Model	USHCN	PhysioNet'12	MIMIC-III	MIMIC-IV	Activity
36–3*	NeuralFlows	0.306 ± 0.03	0.277 ± 0.00	0.308 ± 0.00	0.281 ± 0.00	0.194 ± 0.01
	ProFITi	**0.182 ± 0.01**	0.271 ± 0.00	0.319 ± 0.00	0.279 ± 0.01	**0.185 ± 0.01**
	MOSES	0.220 ± 0.02	*0.260 ± 0.00*	*0.296 ± 0.01*	*0.245 ± 0.01*	0.211 ± 0.02
	CircuITS	**0.182 ± 0.01**	**0.252 ± 0.00**	**0.286 ± 0.01**	**0.221 ± 0.00**	*0.186 ± 0.01*
36–12	NeuralFlows	0.428 ± 0.02	0.320 ± 0.00	0.384 ± 0.02	0.318 ± 0.01	0.239 ± 0.01
	ProFITi	*0.234 ± 0.04*	0.312 ± 0.03	0.383 ± 0.03	0.654 ± 0.08	*0.228 ± 0.01*
	MOSES	**0.211 ± 0.01**	**0.285 ± 0.00**	*0.350 ± 0.01*	*0.262 ± 0.01*	0.235 ± 0.01
	CircuITS	0.359 ± 0.06	*0.286 ± 0.00*	**0.327 ± 0.01**	**0.249 ± 0.01**	**0.226 ± 0.01**
24–24	NeuralFlows	0.433 ± 0.02	0.352 ± 0.00	0.407 ± 0.01	0.333 ± 0.01	0.310 ± 0.02
	ProFITi	*0.355 ± 0.06*	0.336 ± 0.01	0.438 ± 0.07	1.132 ± 0.17	0.313 ± 0.01
	MOSES	**0.239 ± 0.01**	*0.318 ± 0.00*	*0.333 ± 0.00*	**0.279 ± 0.01**	**0.266 ± 0.01**
	CircuITS	0.513 ± 0.11	**0.315 ± 0.00**	**0.323 ± 0.01**	*0.281 ± 0.01*	*0.268 ± 0.01*
12–36	NeuralFlows	**0.437 ± 0.02**	0.396 ± 0.01	0.477 ± 0.00	0.356 ± 0.00	0.344 ± 0.02
	ProFITi	0.593 ± 0.36	0.408 ± 0.03	2.413 ± 1.0	0.823 ± 0.12	0.384 ± 0.04
	MOSES	1.198 ± 1.15	*0.379 ± 0.04*	*0.418 ± 0.00*	*0.332 ± 0.00*	*0.298 ± 0.01*
	CircuITS	*0.456 ± 0.07*	**0.355 ± 0.00**	**0.409 ± 0.00**	**0.292 ± 0.00**	**0.292 ± 0.01**

I.2 Marginal NLL on the Main Benchmark

The marginal Negative Log Likelihood (mNLL) averages the univariate log-likelihoods over the N query targets:

$$\text{mNLL}(y|\mathcal{Q},\mathcal{X}) = \frac{\sum_{n=1}^{N} -\log p(y_n \mid (t_n^{\text{qry}}, c_n^{\text{qry}}), \mathcal{X})}{N}. \tag{25}$$

While the main paper reports the joint njNLL (Table 1), we additionally evaluate the mNLL on the same datasets and forecasting splits. Results are shown in Table 9.

Table 9: Comparison of mNLL for different forecasting splits. Lower values are better. Best results are in **bold**, second best in *italics*. The $\star$ represents that the forecasting horizon includes the next n steps, instead of n-many hours/months.

	Model	USHCN	PhysioNet'12	MIMIC-III	MIMIC-IV	Activity
36–3*	NeuralFlows	0.775 ± 0.18	0.492 ± 0.03	0.866 ± 0.10	0.796 ± 0.05	0.360 ± 0.16
	ProFITi	*-3.324 ± 0.21*	-0.016 ± 0.09	0.408 ± 0.03	*0.500 ± 0.32*	*0.118 ± 0.08*
	MOSES	-3.355 ± 0.16	*-0.271 ± 0.03*	*0.163 ± 0.03*	-0.634 ± 0.02	0.139 ± 0.05
	CircuITS	**-3.717 ± 0.20**	**-0.287 ± 0.04**	**0.095 ± 0.11**	**-0.731 ± 0.06**	**0.060 ± 0.03**
36–12	NeuralFlows	1.344 ± 0.13	0.709 ± 0.05	1.380 ± 0.13	1.073 ± 0.04	0.713 ± 0.08
	ProFITi	**-3.092 ± 0.40**	1.376 ± 1.8	1.158 ± 0.26	0.550 ± 0.15	0.477 ± 0.09
	MOSES	-1.428 ± 0.98	*-0.083 ± 0.03*	*0.624 ± 0.08*	*-0.111 ± 0.06*	*0.313 ± 0.03*
	CircuITS	*-3.072 ± 0.25*	**-0.118 ± 0.03**	**0.140 ± 0.03**	**-0.602 ± 0.04**	**0.311 ± 0.03**
24–24	NeuralFlows	1.436 ± 0.20	1.065 ± 0.08	1.324 ± 0.06	1.171 ± 0.03	1.061 ± 0.12
	ProFITi	-0.626 ± 0.16	0.705 ± 0.18	1.101 ± 0.28	0.609 ± 0.03	1.315 ± 0.36
	MOSES	*-1.917 ± 0.11*	*-0.020 ± 0.06*	*0.700 ± 0.02*	*-0.028 ± 0.02*	*0.506 ± 0.04*
	CircuITS	**-2.593 ± 0.39**	**-0.050 ± 0.01**	**0.150 ± 0.07**	**-0.345 ± 0.10**	**0.500 ± 0.02**
12–36	NeuralFlows	1.458 ± 0.27	1.439 ± 0.20	1.449 ± 0.03	1.195 ± 0.04	1.169 ± 0.10
	ProFITi	-0.116 ± 0.78	2.977 ± 3.0	*0.341 ± 0.25*	0.912 ± 0.14	1.878 ± 0.53
	MOSES	*-1.431 ± 0.71*	**0.040 ± 0.13**	0.850 ± 0.02	*0.218 ± 0.00*	*0.681 ± 0.02*
	CircuITS	**-2.254 ± 0.34**	*0.162 ± 0.05*	**0.016 ± 0.05**	**-0.302 ± 0.02**	**0.518 ± 0.04**

I.3 MSE of the Predicted Means

To assess the point accuracy of the predicted distributions, Table 10 reports the Mean Squared Error (MSE) of the empirical means, calculated with 1000 samples, on the 36–3* task. It includes the additional baselines of Appendix J.

Table 10: Comparing the MSE of the mean estimated by 1000 samples from probabilistic models on the 36–3* setting.

Model	USHCN	PhysioNet'12	MIMIC-III	MIMIC-IV
ProFITi	0.308 ± .06	0.305 ± .01	0.548 ± .06	0.389 ± .02
GRU-ODE	0.410 ± .11	0.329 ± .00	*0.479 ± .04*	0.365 ± .01
NeuralFlows	0.424 ± .11	0.331 ± .01	*0.479 ± .05*	0.374 ± .02
CRU	**0.290 ± .06**	0.475 ± .02	0.725 ± .04	OOM
GPR	0.597 ± .11	0.575 ± .06	0.862 ± .02	0.609 ± .01
GMM	*0.294 ± .08*	*0.293 ± .01*	0.535 ± .06	*0.332 ± .02*
MOSES	0.411 ± .10	0.307 ± .01	0.517 ± .06	0.342 ± .03
CircuITS	0.306 ± .03	**0.292 ± .00**	**0.475 ± .05**	**0.285 ± .00**

J Additional Baselines

In addition to the baselines presented in Section 5, we compare CircuITS to some additional baselines. Similar to NeuralFlows [Biloš et al., 2021], Continuous Recurrent Units (CRU) [Schirmer et al., 2022] and GRU-ODE-Bayes [De Brouwer et al., 2019] learn to forecasting univariate marginals of an IMTS.

Furthermore, we compare with Gaussian Process Regression [Dürichen et al., 2015] and Gaussian Mixture Model [Yalavarthi et al., 2025b], which predicts a mixture of multivariate Gaussian in the same way that MOSES [Yalavarthi et al., 2025b], however without applying normalizing flows. The njNLL and mNLL on the 36–3* task are shown in Table 11; the MSE of the predicted means is reported in Table 10.

Table 11: Comparison of njNLL and mNLL on the 36-3* task. Lower values are better. Best results are in **bold**, second best in *italics*. Results are taken from [Yalavarthi et al., 2025b]. OOM stands for out-of-memory.

	Model	USHCN	PhysioNet'12	MIMIC-III	MIMIC-IV
njNLL	ProFITi	-3.226 ± .23	**-0.647 ± .08**	*-0.377 ± .03*	*-1.777 ± .07*
	GRU-ODE	0.766 ± .16	0.501 ± .00	0.961 ± .06	0.823 ± .32
	NeuralFlows	0.775 ± .15	0.496 ± .00	0.998 ± .18	0.689 ± .09
	CRU	0.761 ± .19	1.057 ± .01	1.234 ± .08	OOM
	GPR	2.011 ± 1.4	1.367 ± .07	3.146 ± .36	2.789 ± .05
	GMM	1.050 ± .03	1.063 ± .00	1.160 ± .02	1.076 ± .00
	MOSES	*-3.357 ± .18*	-0.491 ± .04	-0.305 ± .03	-1.668 ± .10
	CircuITS	**-3.789 ± .22**	*-0.550 ± .01*	**-0.574 ± .08**	**-2.113 ± .04**
mNLL	ProFITi	*-3.324 ± .21*	*-0.016 ± .09*	0.408 ± .03	*0.500 ± .32*
	GRU-ODE	0.776 ± .17	0.504 ± .06	0.839 ± .03	0.876 ± .59
	NeuralFlows	0.775 ± .18	0.492 ± .03	0.866 ± .10	0.796 ± .05
	CRU	0.762 ± .18	0.931 ± .02	1.209 ± .04	OOM
	GPR	1.235 ± .10	1.161 ± .07	1.341 ± .01	1.161 ± .01
	GMM	1.042 ± .02	1.069 ± .00	1.124 ± .01	1.075 ± .01
	MOSES	-3.355 ± .16	-0.271 ± .03	*0.163 ± .03*	-0.634 ± .02
	CircuITS	**-3.717 ± .20**	**-0.287 ± .04**	**0.095 ± .11**	**-0.731 ± .06**

K Importance of Channel Ordering

To quantify the importance of channel ordering, we repeated the PhysioNet'12 evaluation on the 36–12 task using five randomly sampled channel permutations. The resulting test njNLL values are provided in Table 12. While the results indicate that ordering has a statistically discernible impact, the variation is negligible relative to the performance gaps between the competing models. Consequently, the choice of permutation does not alter the qualitative conclusions of the experiment.

Table 12: Permuting the channels order in the recursive SPN aggregation for the Physionet'12 36–12 task.

	Original	Perm. 1	Perm. 2	Perm. 3	Perm. 4
njNLL	-0.536 ± .04	-0.540 ± .02	-0.574 ± .03	-0.526 ± .02	-0.547 ± .03

L Quantifying the Effect of Query-Dependent SPN Weights

Appendix B.4 shows that query-independent effective mixture weights are necessary for marginalization consistency. To quantify how strongly a violation shows on real data, we construct an inconsistent variant of CircuITS in which the SPN weights depend on the query $\mathcal{Q}$, thereby violating Property 3 of Lemma B.1. In this variant, the learnable channel-wise queries $\mathbf{E}^{\text{cw}}$ that produce the sum-layer weights $\mathbf{W}^{(c)}$ (Equation (17)) attend over the (linearly projected) query embeddings $[\phi^{\text{TIME}}(t_n^{\text{qry}}) \,\|\, \tilde{\mathbf{h}}_{c_n}]$ of all N query points instead of over the channel states $\tilde{\mathbf{H}}$. The root weights and all other components remain unchanged. We compare its marginalization consistency with that of the original CircuITS, measured on samples.

On 100 test instances we build a synthetic query with a single query point in every channel, whose timestamps are drawn uniformly from the range of the observed timestamps of the instance; only the observations $\mathcal{X}$ are taken from the data. We then pick two distinct channels (a, b) at random and draw $n = 1000$ samples of their bivariate marginal in two ways: once from the full all-channel query and once from a query containing only a and b. A marginalization consistent model must yield the same bivariate distribution in both cases. We measure the discrepancy by the 2-Wasserstein distance between the two sample sets $\{x_i\}_{i=1}^n$ and $\{y_i\}_{i=1}^n$:

$$W_2 = \left(\min_{\sigma \in S_n} \frac{1}{n} \sum_{i=1}^{n} \|x_i - y_{\sigma(i)}\|_2^2 \right)^{1/2}. \tag{26}$$

Since 1000 samples never reproduce a distribution exactly, we report the ratio of the mean discrepancy to a noise floor, the mean W_2 distance between two sets of 1000 samples of the same two-channel query drawn with different seeds. Values near 1 indicate consistency. Table 13 shows that the ratio of CircuITS is close to 1 on all datasets, whereas the query-dependent variant exceeds the noise floor by a factor of up to 5.5. The ratio of 0.80 on PhysioNet'12 is explained by the high variance of the sampling error on this dataset (W_2 between marginalized and two-channel query: 0.075 ± 0.13; W_2 between resampled two-channel queries: 0.093 ± 0.32). Hence, the inconsistency induced by query-dependent SPN weights is clearly measurable on real data.

Table 13: Ratio of the W_2 discrepancy between the marginalized and the directly queried bivariate distribution to the resampling noise floor. Values near 1 indicate marginalization consistency.

Model	USHCN	PhysioNet'12	MIMIC-III	MIMIC-IV
CircuITS	**0.99**	**0.80**	**1.00**	**1.01**
CircuITS (query-dep. weights)	1.25	4.72	5.54	4.99

M Sensitivity Analysis for the Number of Components

We evaluated the sensitivity of CircuITS to the hyperparameter K, representing the number of leaf nodes per channel (mixture components). As shown in Figure 5, the model exhibits moderate robustness to variations in K. Our results suggest that selecting K from the set $\{2, 3, 4\}$ is sufficient, as the optimal configuration was found to be $K = 2$ or $K = 3$ for the evaluated task (36–12).

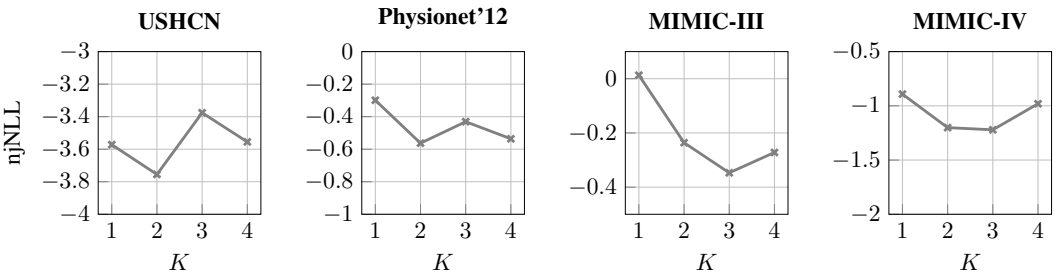

Figure 5: Sensitivity Analysis on the 36-12 task for the number of circuit components K.

N Memory Consumption and Epoch Times

We conduct an experiment to compare CircuITS' memory consumption during training and epoch times with those of ProFITi and MOSES. To do this, we use toy datasets based on Brownian motion, in which we can vary the number of channels C and the number of time steps L. The models are tasked with predicting the second half of time steps based on the first half, similar to the 24–24 task in our main experiment (Section 5). To simulate an IMTS, we randomly drop 10% of all observations and forecasting queries. Hence, the expected number of query points N is $\frac{0.9LC}{2}$.

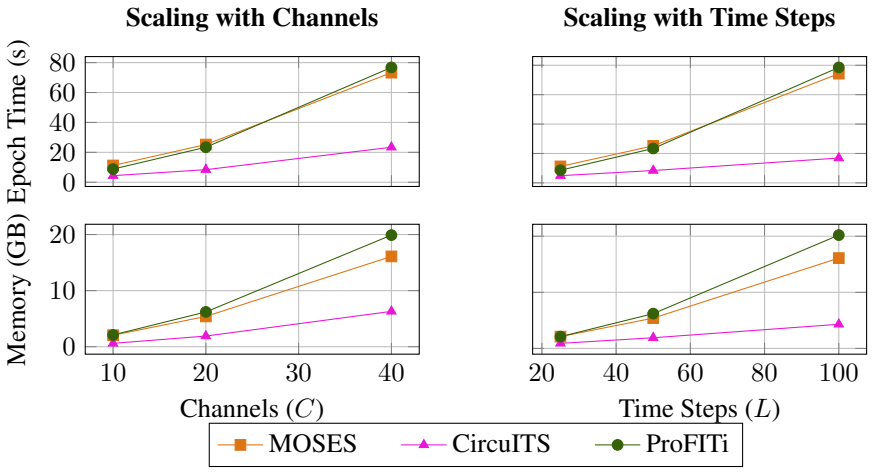

Figure 6: Comparing memory consumption and epoch times during training on Toy datasets.

For MOSES, we set the number of mixture components to 3, the hidden dimension to 128, and the number of attention heads to 4. ProFITi is applied using 7 flow layers and a latent dimension of 64. For CircuITS, we set the number of components K to 3, the number of flow layers to 2, and the hidden dimension to 32. We run these experiments on a single NVIDIA A40 with a batch size of 64. Figure 6 presents the results. In this toy setup, where the channels all have a similar number of queries, CircuITS is significantly faster than MOSES and ProFITi, while using less memory to compute the gradients and update the weights.

As stated in Section 4.5 CircuITS reduces the complexity compared to a model that computes full covariance over all query points by factor C^2 under the assumption that the forecasting targets are distributed evenly among channels.

However, in real world IMTS datasets this is typically not the case and some channels are observed more frequently than others. At least in our implementation the speed-up that our model generates by grouping the forecasting targets by channels decreases, when forecasting queries are concentrating on single channels. As an example we show the distribution of queries over channels for MIMIC-III in the 12–36 setting in Figure 7. For such uneven IMTS CircuITS is actually slower than MOSES and ProFITi as we show in Table 14.

Table 14: Epoch times on MIMIC-III, run on a single A100 GPU.

Model	36–12	24–24	12–36
ProFITi	13s	14s	15s
MOSES	13s	12s	13s
CircuITS	27s	24s	40s

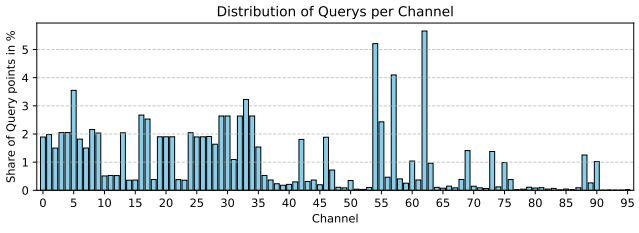

Figure 7: Query distribution for MIMIC-III 12–36.

O Broader Impacts

CircuITS improves the calibration and consistency of probabilistic forecasts for irregular multivariate time series, which can support better-informed decision-making in domains such as climate monitoring and clinical care — two of the application areas represented in our benchmarks, alongside human motion tracking. Reliable uncertainty quantification is particularly valuable in healthcare settings, where overconfident point predictions can mislead clinicians. At the same time, more accurate-looking probabilistic forecasts can also induce automation bias if practitioners trust them beyond what calibration warrants, and the model's guarantees concern marginalization consistency rather than the correctness of the underlying data or domain shift at deployment time. We therefore view CircuITS as a tool to be used alongside, not in place of, domain expertise. We do not release new datasets and use only existing public benchmarks (USHCN, PhysioNet 2012, MIMIC-III, MIMIC-IV, Human Activity), with the clinical datasets de-identified by their creators, under their respective licenses, so we do not see additional privacy or dual-use concerns introduced by this work.

NeurIPS Paper Checklist

1. **Claims**

   Question: Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope?

   Answer: [Yes]

   Justification: The contributions stated in the introduction (a marginalization-consistent SPN architecture for IMTS, a query encoder, and superior joint density estimation on five datasets) are directly supported by Appendices B and B.2, the architecture in Section 5, and the results in Table 1.

   Guidelines:

   - The answer [N/A] means that the abstract and introduction do not include the claims made in the paper.
   - The abstract and/or introduction should clearly state the claims made, including the contributions made in the paper and important assumptions and limitations. A [No] or [N/A] answer to this question will not be perceived well by the reviewers.
   - The claims made should match theoretical and experimental results, and reflect how much the results can be expected to generalize to other settings.
   - It is fine to include aspirational goals as motivation as long as it is clear that these goals are not attained by the paper.

2. **Limitations**

   Question: Does the paper discuss the limitations of the work performed by the authors?

   Answer: [Yes]

   Justification: A dedicated Limitations section discusses channel-order dependence of the sequential SPN construction, the inability to compactly represent smooth inter-channel dependencies, and that the $\mathcal{O}(N^3/C^2)$ complexity assumes evenly distributed queries (with longer epoch times when queries concentrate on few channels).

   Guidelines:

   - The answer [N/A] means that the paper has no limitation while the answer [No] means that the paper has limitations, but those are not discussed in the paper.
   - The authors are encouraged to create a separate "Limitations" section in their paper.
   - The paper should point out any strong assumptions and how robust the results are to violations of these assumptions (e.g., independence assumptions, noiseless settings, model well-specification, asymptotic approximations only holding locally). The authors should reflect on how these assumptions might be violated in practice and what the implications would be.
   - The authors should reflect on the scope of the claims made, e.g., if the approach was only tested on a few datasets or with a few runs. In general, empirical results often depend on implicit assumptions, which should be articulated.
   - The authors should reflect on the factors that influence the performance of the approach. For example, a facial recognition algorithm may perform poorly when image resolution is low or images are taken in low lighting. Or a speech-to-text system might not be used reliably to provide closed captions for online lectures because it fails to handle technical jargon.
   - The authors should discuss the computational efficiency of the proposed algorithms and how they scale with dataset size.
   - If applicable, the authors should discuss possible limitations of their approach to address problems of privacy and fairness.
   - While the authors might fear that complete honesty about limitations might be used by reviewers as grounds for rejection, a worse outcome might be that reviewers discover limitations that aren't acknowledged in the paper. The authors should use their best judgment and recognize that individual actions in favor of transparency play an important role in developing norms that preserve the integrity of the community. Reviewers will be specifically instructed to not penalize honesty concerning limitations.

3. **Theory assumptions and proofs**

   Question: For each theoretical result, does the paper provide the full set of assumptions and a complete (and correct) proof?

   Answer: [Yes]

   Justification: The marginalization-consistency results (Lemma in Appendix B and Proposition in Appendix B.2) are formally stated with their assumptions and accompanied by complete proofs in the appendix.

   Guidelines:

   - The answer [N/A] means that the paper does not include theoretical results.
   - All the theorems, formulas, and proofs in the paper should be numbered and cross-referenced.
   - All assumptions should be clearly stated or referenced in the statement of any theorems.
   - The proofs can either appear in the main paper or the supplemental material, but if they appear in the supplemental material, the authors are encouraged to provide a short proof sketch to provide intuition.
   - Inversely, any informal proof provided in the core of the paper should be complemented by formal proofs provided in appendix or supplemental material.
   - Theorems and Lemmas that the proof relies upon should be properly referenced.

4. **Experimental result reproducibility**

   Question: Does the paper fully disclose all the information needed to reproduce the main experimental results of the paper to the extent that it affects the main claims and/or conclusions of the paper (regardless of whether the code and data are provided or not)?

   Answer: [Yes]

   Justification: The architecture is fully specified in Section 5 and the appendix, datasets and the preprocessing/splitting protocol of Yalavarthi et al. [2025b] are described in Appendix F, hyperparameters and search spaces in Appendix G, and the code is publicly available (linked from the introduction).

   Guidelines:

   - The answer [N/A] means that the paper does not include experiments.
   - If the paper includes experiments, a [No] answer to this question will not be perceived well by the reviewers: Making the paper reproducible is important, regardless of whether the code and data are provided or not.
   - If the contribution is a dataset and/or model, the authors should describe the steps taken to make their results reproducible or verifiable.
   - Depending on the contribution, reproducibility can be accomplished in various ways. For example, if the contribution is a novel architecture, describing the architecture fully might suffice, or if the contribution is a specific model and empirical evaluation, it may be necessary to either make it possible for others to replicate the model with the same dataset, or provide access to the model. In general, releasing code and data is often one good way to accomplish this, but reproducibility can also be provided via detailed instructions for how to replicate the results, access to a hosted model (e.g., in the case of a large language model), releasing of a model checkpoint, or other means that are appropriate to the research performed.
   - While NeurIPS does not require releasing code, the conference does require all submissions to provide some reasonable avenue for reproducibility, which may depend on the nature of the contribution. For example
     (a) If the contribution is primarily a new algorithm, the paper should make it clear how to reproduce that algorithm.
     (b) If the contribution is primarily a new model architecture, the paper should describe the architecture clearly and fully.
     (c) If the contribution is a new model (e.g., a large language model), then there should either be a way to access this model for reproducing the results or a way to reproduce the model (e.g., with an open-source dataset or instructions for how to construct the dataset).

(d) We recognize that reproducibility may be tricky in some cases, in which case authors are welcome to describe the particular way they provide for reproducibility. In the case of closed-source models, it may be that access to the model is limited in some way (e.g., to registered users), but it should be possible for other researchers to have some path to reproducing or verifying the results.

5. **Open access to data and code**

   Question: Does the paper provide open access to the data and code, with sufficient instructions to faithfully reproduce the main experimental results, as described in supplemental material?

   Answer: [Yes]

   Justification: We provide a public code repository (linked in the footnote of the introduction) containing the model implementation and training scripts. Datasets (USHCN, PhysioNet'12, MIMIC-III, MIMIC-IV, Human Activity) are public benchmarks; for the first four we follow the preprocessing and splitting protocol of Yalavarthi et al. [2025b], and we describe the preprocessing of all datasets in Appendix F.

   Guidelines:

   - The answer [N/A] means that paper does not include experiments requiring code.
   - Please see the NeurIPS code and data submission guidelines (`https://neurips.cc/public/guides/CodeSubmissionPolicy`) for more details.
   - While we encourage the release of code and data, we understand that this might not be possible, so [No] is an acceptable answer. Papers cannot be rejected simply for not including code, unless this is central to the contribution (e.g., for a new open-source benchmark).
   - The instructions should contain the exact command and environment needed to run to reproduce the results. See the NeurIPS code and data submission guidelines (`https://neurips.cc/public/guides/CodeSubmissionPolicy`) for more details.
   - The authors should provide instructions on data access and preparation, including how to access the raw data, preprocessed data, intermediate data, and generated data, etc.
   - The authors should provide scripts to reproduce all experimental results for the new proposed method and baselines. If only a subset of experiments are reproducible, they should state which ones are omitted from the script and why.
   - At submission time, to preserve anonymity, the authors should release anonymized versions (if applicable).
   - Providing as much information as possible in supplemental material (appended to the paper) is recommended, but including URLs to data and code is permitted.

6. **Experimental setting/details**

   Question: Does the paper specify all the training and test details (e.g., data splits, hyperparameters, how they were chosen, type of optimizer) necessary to understand the results?

   Answer: [Yes]

   Justification: Appendix G specifies the optimizer (AdamW), learning rate, weight decay, batch size, scheduler, early-stopping criterion, and the random hyperparameter search spaces for CircuITS and all baselines; data splits follow the protocol of Yalavarthi et al. [2025b] (Appendix F).

   Guidelines:

   - The answer [N/A] means that the paper does not include experiments.
   - The experimental setting should be presented in the core of the paper to a level of detail that is necessary to appreciate the results and make sense of them.
   - The full details can be provided either with the code, in appendix, or as supplemental material.

7. **Experiment statistical significance**

   Question: Does the paper report error bars suitably and correctly defined or other appropriate information about the statistical significance of the experiments?

Answer: [Yes]

Justification: All result tables report mean ± standard deviation across five paired train/validation/test splits, with the same five splits used for every method (Section 5, Table 1).

Guidelines:

- The answer [N/A] means that the paper does not include experiments.
- The authors should answer [Yes] if the results are accompanied by error bars, confidence intervals, or statistical significance tests, at least for the experiments that support the main claims of the paper.
- The factors of variability that the error bars are capturing should be clearly stated (for example, train/test split, initialization, random drawing of some parameter, or overall run with given experimental conditions).
- The method for calculating the error bars should be explained (closed form formula, call to a library function, bootstrap, etc.)
- The assumptions made should be given (e.g., Normally distributed errors).
- It should be clear whether the error bar is the standard deviation or the standard error of the mean.
- It is OK to report 1-sigma error bars, but one should state it. The authors should preferably report a 2-sigma error bar than state that they have a 96% CI, if the hypothesis of Normality of errors is not verified.
- For asymmetric distributions, the authors should be careful not to show in tables or figures symmetric error bars that would yield results that are out of range (e.g., negative error rates).
- If error bars are reported in tables or plots, the authors should explain in the text how they were calculated and reference the corresponding figures or tables in the text.

8. **Experiments compute resources**

   Question: For each experiment, does the paper provide sufficient information on the computer resources (type of compute workers, memory, time of execution) needed to reproduce the experiments?

   Answer: [Yes]

   Justification: The compute hardware (NVIDIA A100/A40 GPUs) is reported in Appendix N, and per-epoch runtimes for CircuITS and baselines are given in Table 14.

   Guidelines:

   - The answer [N/A] means that the paper does not include experiments.
   - The paper should indicate the type of compute workers CPU or GPU, internal cluster, or cloud provider, including relevant memory and storage.
   - The paper should provide the amount of compute required for each of the individual experimental runs as well as estimate the total compute.
   - The paper should disclose whether the full research project required more compute than the experiments reported in the paper (e.g., preliminary or failed experiments that didn't make it into the paper).

9. **Code of ethics**

   Question: Does the research conducted in the paper conform, in every respect, with the NeurIPS Code of Ethics `https://neurips.cc/public/EthicsGuidelines`?

   Answer: [Yes]

   Justification: The work uses only existing, de-identified public benchmarks under their respective licenses, releases no new data or high-risk models; we have reviewed and conform to the NeurIPS Code of Ethics.

   Guidelines:

   - The answer [N/A] means that the authors have not reviewed the NeurIPS Code of Ethics.

- If the authors answer [No], they should explain the special circumstances that require a deviation from the Code of Ethics.
- The authors should make sure to preserve anonymity (e.g., if there is a special consideration due to laws or regulations in their jurisdiction).

10. **Broader impacts**

    Question: Does the paper discuss both potential positive societal impacts and negative societal impacts of the work performed?

    Answer: [Yes]

    Justification: Appendix O discusses positive impacts (better-calibrated probabilistic forecasts for clinical and climate applications) and negative impacts (automation bias, the gap between marginalization consistency and downstream correctness under domain shift).

    Guidelines:

    - The answer [N/A] means that there is no societal impact of the work performed.
    - If the authors answer [N/A] or [No], they should explain why their work has no societal impact or why the paper does not address societal impact.
    - Examples of negative societal impacts include potential malicious or unintended uses (e.g., disinformation, generating fake profiles, surveillance), fairness considerations (e.g., deployment of technologies that could make decisions that unfairly impact specific groups), privacy considerations, and security considerations.
    - The conference expects that many papers will be foundational research and not tied to particular applications, let alone deployments. However, if there is a direct path to any negative applications, the authors should point it out. For example, it is legitimate to point out that an improvement in the quality of generative models could be used to generate Deepfakes for disinformation. On the other hand, it is not needed to point out that a generic algorithm for optimizing neural networks could enable people to train models that generate Deepfakes faster.
    - The authors should consider possible harms that could arise when the technology is being used as intended and functioning correctly, harms that could arise when the technology is being used as intended but gives incorrect results, and harms following from (intentional or unintentional) misuse of the technology.
    - If there are negative societal impacts, the authors could also discuss possible mitigation strategies (e.g., gated release of models, providing defenses in addition to attacks, mechanisms for monitoring misuse, mechanisms to monitor how a system learns from feedback over time, improving the efficiency and accessibility of ML).

11. **Safeguards**

    Question: Does the paper describe safeguards that have been put in place for responsible release of data or models that have a high risk for misuse (e.g., pre-trained language models, image generators, or scraped datasets)?

    Answer: [N/A]

    Justification: We release neither scraped datasets nor generative models with significant misuse potential; the released artifact is a forecasting model trained on existing public benchmarks.

    Guidelines:

    - The answer [N/A] means that the paper poses no such risks.
    - Released models that have a high risk for misuse or dual-use should be released with necessary safeguards to allow for controlled use of the model, for example by requiring that users adhere to usage guidelines or restrictions to access the model or implementing safety filters.
    - Datasets that have been scraped from the Internet could pose safety risks. The authors should describe how they avoided releasing unsafe images.
    - We recognize that providing effective safeguards is challenging, and many papers do not require this, but we encourage authors to take this into account and make a best faith effort.

12. **Licenses for existing assets**

    Question: Are the creators or original owners of assets (e.g., code, data, models), used in the paper, properly credited and are the license and terms of use explicitly mentioned and properly respected?

    Answer: [Yes]

    Justification: All baselines (ProFITi, MOSES, NeuralFlows, Tripletformer) and datasets (USHCN, PhysioNet'12, MIMIC-III, MIMIC-IV, Human Activity) are cited to their original sources in Appendix F and the main text, and used in accordance with their respective licenses/data-use agreements; Human Activity is distributed by the UCI Machine Learning Repository under CC BY 4.0.

    Guidelines:

    - The answer [N/A] means that the paper does not use existing assets.
    - The authors should cite the original paper that produced the code package or dataset.
    - The authors should state which version of the asset is used and, if possible, include a URL.
    - The name of the license (e.g., CC-BY 4.0) should be included for each asset.
    - For scraped data from a particular source (e.g., website), the copyright and terms of service of that source should be provided.
    - If assets are released, the license, copyright information, and terms of use in the package should be provided. For popular datasets, `paperswithcode.com/datasets` has curated licenses for some datasets. Their licensing guide can help determine the license of a dataset.
    - For existing datasets that are re-packaged, both the original license and the license of the derived asset (if it has changed) should be provided.
    - If this information is not available online, the authors are encouraged to reach out to the asset's creators.

13. **New assets**

    Question: Are new assets introduced in the paper well documented and is the documentation provided alongside the assets?

    Answer: [Yes]

    Justification: The only new asset is the CircuITS code, released publicly on GitHub (link in the introduction footnote) with documentation of training and evaluation; no new datasets or models are released.

    Guidelines:

    - The answer [N/A] means that the paper does not release new assets.
    - Researchers should communicate the details of the dataset/code/model as part of their submissions via structured templates. This includes details about training, license, limitations, etc.
    - The paper should discuss whether and how consent was obtained from people whose asset is used.
    - At submission time, remember to anonymize your assets (if applicable). You can either create an anonymized URL or include an anonymized zip file.

14. **Crowdsourcing and research with human subjects**

    Question: For crowdsourcing experiments and research with human subjects, does the paper include the full text of instructions given to participants and screenshots, if applicable, as well as details about compensation (if any)?

    Answer: [N/A]

    Justification: The paper does not involve crowdsourcing or research with human subjects; we use only existing, de-identified public benchmark datasets.

    Guidelines:

    - The answer [N/A] means that the paper does not involve crowdsourcing nor research with human subjects.

- Including this information in the supplemental material is fine, but if the main contribution of the paper involves human subjects, then as much detail as possible should be included in the main paper.
- According to the NeurIPS Code of Ethics, workers involved in data collection, curation, or other labor should be paid at least the minimum wage in the country of the data collector.

15. **Institutional review board (IRB) approvals or equivalent for research with human subjects**

    Question: Does the paper describe potential risks incurred by study participants, whether such risks were disclosed to the subjects, and whether Institutional Review Board (IRB) approvals (or an equivalent approval/review based on the requirements of your country or institution) were obtained?

    Answer: [N/A]

    Justification: The paper does not conduct new research with human subjects; the MIMIC-III/IV and PhysioNet datasets used were collected and de-identified by their original creators under their own IRB approvals.

    Guidelines:

    - The answer [N/A] means that the paper does not involve crowdsourcing nor research with human subjects.
    - Depending on the country in which research is conducted, IRB approval (or equivalent) may be required for any human subjects research. If you obtained IRB approval, you should clearly state this in the paper.
    - We recognize that the procedures for this may vary significantly between institutions and locations, and we expect authors to adhere to the NeurIPS Code of Ethics and the guidelines for their institution.
    - For initial submissions, do not include any information that would break anonymity (if applicable), such as the institution conducting the review.

16. **Declaration of LLM usage**

    Question: Does the paper describe the usage of LLMs if it is an important, original, or non-standard component of the core methods in this research? Note that if the LLM is used only for writing, editing, or formatting purposes and does *not* impact the core methodology, scientific rigor, or originality of the research, declaration is not required.

    Answer: [N/A]

    Justification: LLMs are not part of the core methodology of this paper; any use was limited to writing/editing assistance, which does not require declaration under the policy.

    Guidelines:

    - The answer [N/A] means that the core method development in this research does not involve LLMs as any important, original, or non-standard components.
    - Please refer to our LLM policy in the NeurIPS handbook for what should or should not be described.