Adaptive Quantization in GFN for Probabilistic Sequential Prediction

Nadhir Hassen, Zhen Zheng, Johan W.Verjans

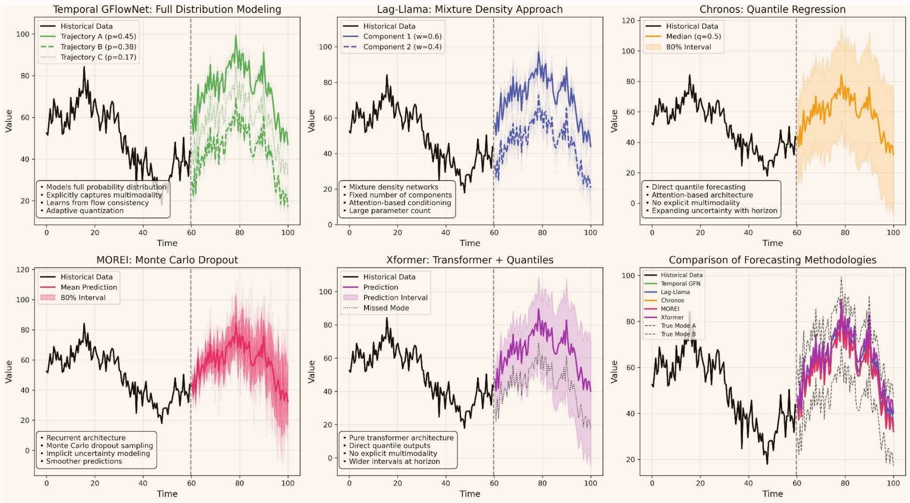
Mila

THE UNIVERSITY of ADELAIDE

Australian Institute for Machine Learning

Overview of Temporal GFN

1. Methodology
2. Core Innovation
3. Key Takeaway
4. Experiments and results



What Happens During Training?

Temporal GFN learns to generate **sequences of forecasted values** by taking actions (predicting quantized values) step by step. The key goals are:

1. **Sample high-reward future trajectories** (matching the ground truth),
2. **Adapt the quantization resolution** over time based on model performance,
3. **Train a forward policy** (and optionally backward policy) using **Trajectory Balance loss** with gradients propagated through **Straight-Through Estimators (STE)**.

GFN for Time Series: A forecast Cursor

Goal: Construct initial GFN state from the time series history (context window).

- Given a dataset with time series samples $x_{1:T}$, we split each into:
  - **Context window** $x_{1:C}$ → used as input ("observed history").
  - **Forecast horizon** $x_{C+1:C+H}$ → ground truth to evaluate the prediction.
- The **initial state** s_0 is constructed as:

$$s_0 = \mathrm{Encoder}(x_{1:C})$$

  where the encoder could be an LSTM, Transformer encoder, or sliding-window patch embedding layer. This produces a fixed-length latent representation that the GFN policy can condition on.
- The state representation includes:
  - Latent embedding of the history window,
  - A **forecast cursor** (step counter $t = 0$),
  - Possibly a cache of previously sampled outputs if needed for autoregression.

Discrete Action Space via Quantization

Goal: Convert continuous forecast space into discrete tokens/actions.

- The normalized range of values $[v_{\min}, v_{\max}]$ is partitioned into K bins:

$$\{q_1, q_2, \dots, q_K\}, \quad q_k = v_{\min} + (k - 0.5) \cdot \Delta, \quad \Delta = \frac{v_{\max} - v_{\min}}{K}$$

- These **quantized centers** q_k act as tokens. At each forecasting step t, the GFN **chooses one token** q_k as the predicted value for x_{C+t+1}.
- **Each GFN action** is selecting one of these bins:

$$a_t \in \{q_1, q_2, \dots, q_K\}$$

- The quantized value is appended to the state, updating the forecast:

$$s_{t+1} = s_t \cup \{a_t\}$$

Trajectory Construction via Forward Policy

Goal: Build a full trajectory $\tau = (a_0, \ldots, a_{H-1})$ of quantized forecasts using a learned policy.

At each time step $t \in [0, H-1]$:

1. **Forward Policy Sampling:**
   - Compute logits $\mathbf{z}_t = f_\theta(s_t) \in \mathbb{R}^K$
   - Convert to probabilities over actions:

$$P_F(a_t = q_k | s_t) = \mathrm{softmax}(z_{t,k})$$

2. **Action Selection via STE:**
   - **Hard forward action:** sample or take argmax over P_F → yields discrete token $a_t^{\mathrm{hard}} = q_k$
   - **Soft value for gradient:** compute expected value:

$$a_t^{\mathrm{soft}} = \sum_k q_k \cdot P_F(a_t = q_k \mid s_t)$$

3. **State Update:**
   - Append discrete a_t^{hard} to current sequence.
   - Update s_{t+1} with new time step's embedding (can use a positional encoding or RNN hidden state update).
   - Repeat for H steps to get complete trajectory $\tau = (a_0, \ldots, a_{H-1})$

Reward Computation

Goal: Evaluate trajectory quality by comparing with true future values.

Let true future be $y = x_{C+1:C+H}$, predicted quantized values $z = (a_0, ..., a_{H-1})$

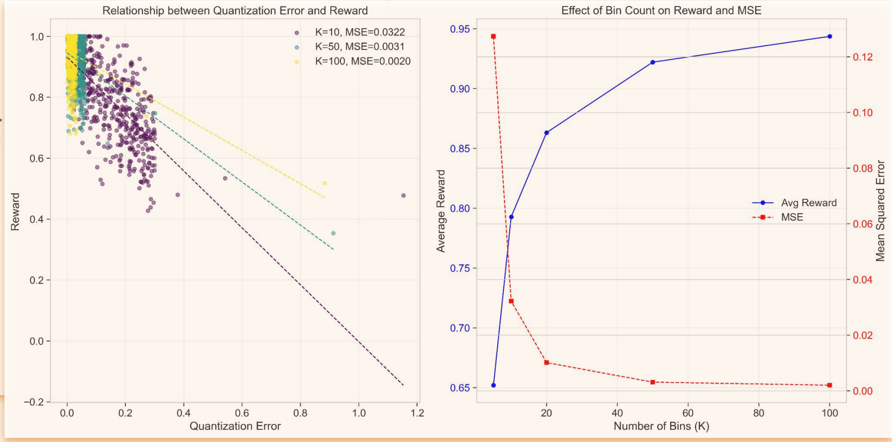
- Define **normalized squared error**:

$$\mathrm{NSE}(z, y) = \frac{1}{H} \sum_{t=1}^{H} \frac{(z_t - y_t)^2}{(v_{\max} - v_{\min})^2}$$

- Define **exponential reward**:

$$R(\tau) = \exp(-\beta \cdot \mathrm{NSE}(z, y))$$

where β is a scaling constant (e.g., 5–20).



Adaptive Quantization Update

Goal: Adjust number of bins K based on model learning progress.

After each epoch:

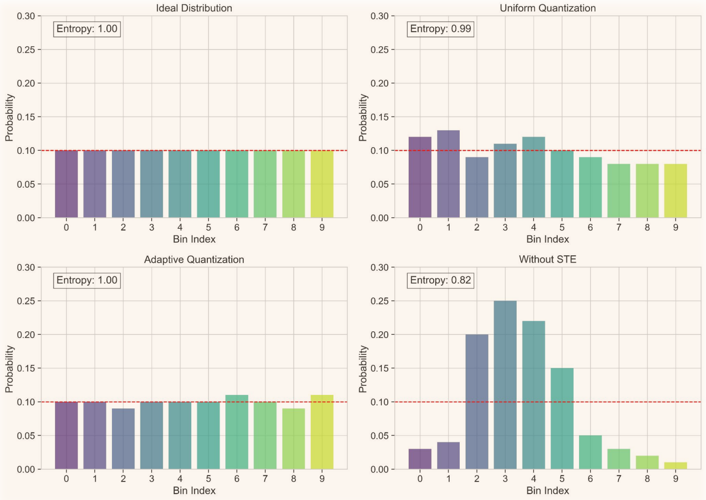
1. **Track:**
   - Recent average reward improvement ΔR_e
   - Policy entropy $H_e = \mathbb{E}_t[H(P_F(a_t|s_t))] \in [0, 1]$
2. **Update factor:**

$$\eta_e = 1 + \lambda \left(\frac{\max(0, \epsilon - \Delta R_e)}{\epsilon} + (1 - H_e) \right)$$

3. **New bin count:**

$$K_e = \min(K_{\max}, \lfloor K_{e-1} \cdot \eta_e \rfloor)$$

4. **Re-quantize bins:**
   - Uniformly repartition $[v_{\min}, v_{\max}]$ into new K_e bins.
   - Optionally interpolate soft logits from old K to new K if needed.



Loss and training Objective

Goal: Train forward policy (and optionally backward policy) to match desired sampling distribution $P^*(\tau) \propto R(\tau)$

Trajectory Balance Loss:

For trajectory $\tau = (s_0 \to s_1 \to \cdots \to s_H)$:

$$\mathcal{L}_{TB} = \left(\log Z + \sum_{t=0}^{H-1} \log P_F(a_t \mid s_t) - \sum_{t=1}^{H} \log P_B(s_{t-1} \mid s_t) - \log R(\tau) \right)^2$$

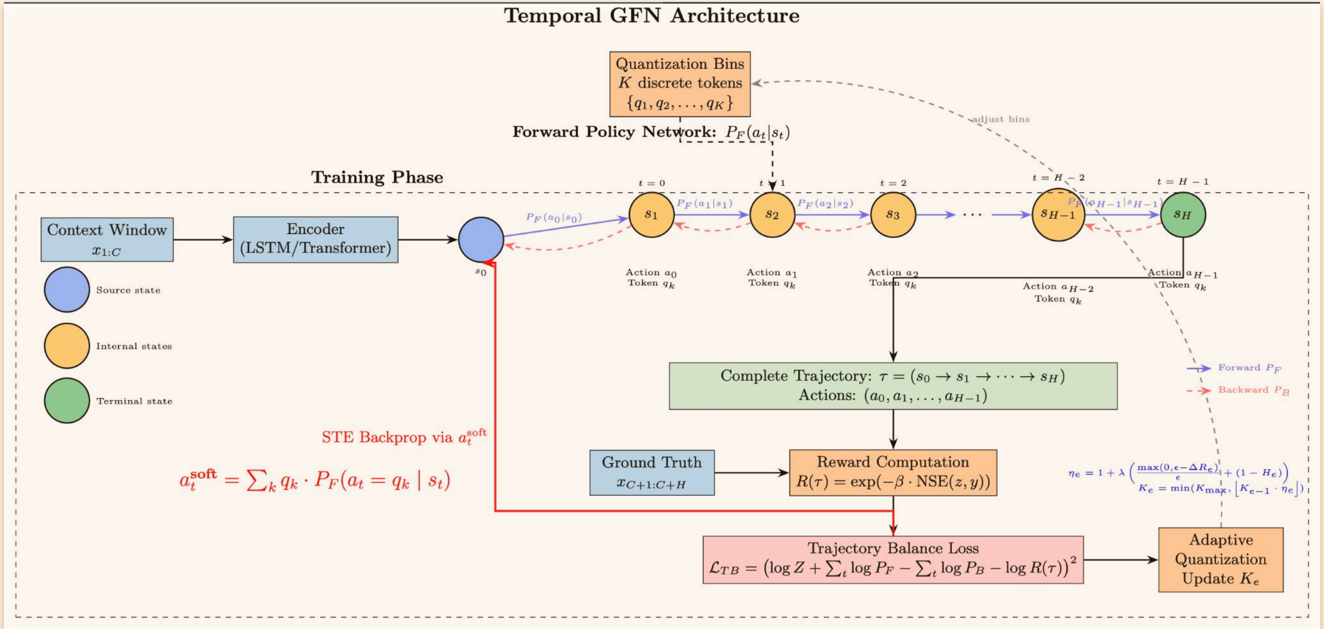
- **Optional entropy regularization:**

$$\mathcal{L}_{\text{entropy}} = -\lambda_H \sum_{t=0}^{H-1} H(P_F(\cdot|s_t))$$

- **Total loss:**

$$\mathcal{L}_{\text{total}} = \mathcal{L}_{TB} + \mathcal{L}_{\text{entropy}}$$

GFN Architecture



Experiments and results

Table 1: Aggregated Relative Performance vs. RL/MCMC on Benchmark I.

Method	CRPS (↓)	WQL (↓)	MASE (↓)	Avg. Improv. (%)
Temporal GFN (Adaptive K20)	0.1674	0.2273	**0.8938**	-
Temporal GFN (Fixed K20)	0.1781	0.2401	0.9363	-5.9%
Temporal GFN (Learned Pol. Adapt K10)	**0.1453**	**0.2137**	1.0345	+6.0%*
PPO	0.2299	0.3013	1.1759	-25.1%
SAC	0.2423	0.3227	1.2491	-29.5%
MCMC	0.2673	0.3522	1.3523	-35.5%

*Avg. Improv. for Learned Policy calculated based on CRPS/WQL vs Adaptive K20.

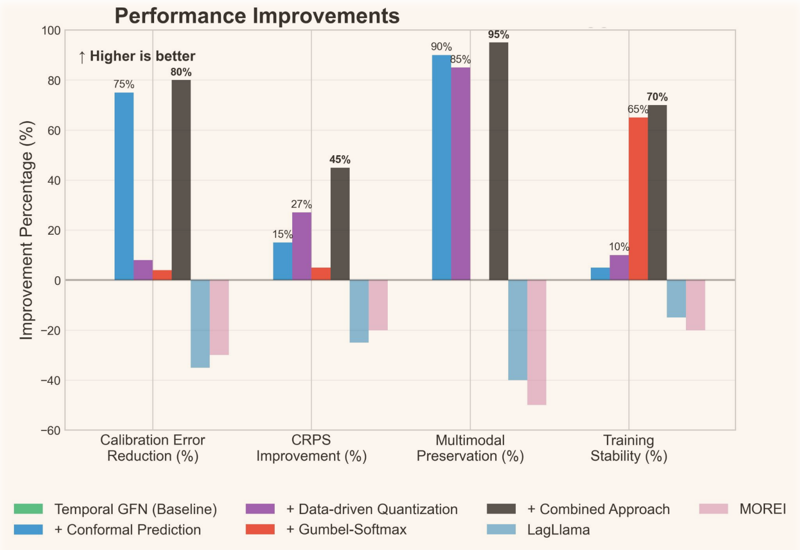
Table 2: Aggregated Relative Performance vs. SOTA Models (In-Domain / Zero-Shot).

Model	Eval Setting	CRPS (↓)	WQL (↓)	MASE (↓)	Calib. Err. (↓)	Multimod. (↑)
Temporal GFN	Zero-Shot / In-Domain	**0.1542**	**0.2158**	**0.9378**	**0.0348**	**0.8762**
Lag-Llama	Zero-Shot / Pretrained	0.1675	0.2401	0.9532	0.0623	0.7234
Chronos (Base)	Zero-Shot / Pretrained	0.1731	0.2534	1.0273	0.0571	0.6957
MOREI (Base)	Task-Specific Trained	0.1859	0.2678	1.0871	0.0745	0.6182
TimeFM	Task-Specific Trained	0.2076	0.2846	1.1258	0.1183	0.3012
Temporal GFN (Learned Pol.)	In-Domain	0.1453	0.2137	1.0345	-	-

Main Temporal GFN results reflect strong performance across settings. Calib. Err. & Multimod. primarily from healthcare/extended benchmarks. Task-Specific models trained per dataset. Learned Policy results from Table 3.

Table 3: Ablation Study Configurations and Main Results.

Experiment Config	Quantization	Start K	Policy Type	CRPS	WQL	MASE
Fixed K=10	fixed	10	uniform	0.1546	0.2346	1.1946
Fixed K=20	fixed	20	uniform	0.1921	0.2521	0.9721
Adaptive K=10	adaptive	10	uniform	0.1688	0.2408	1.1048
Adaptive K=20	adaptive	20	uniform	0.1845	0.2385	**0.9532**
Learned Policy (Adapt K10)	adaptive	10	learned	**0.1453**	**0.2137**	1.0345



Thank You !