Predicting Slowly-Varying Mean and Variance in Gaussian-Type Time-Series: A Comprehensive Deep-Learning Benchmark

Soonyong Song
Industry & Energy Convergence Research Division
Electronics and Telecommunications Research Institute (ETRI)
Daejeon, South Korea
soony@etri.re.kr

***Abstract*—This paper presents a head-to-head benchmark of six representative deep-learning models—Heteroscedastic LSTM, Dilated TCN, Temporal Fusion Transformer, Neural CDE, Variational RNN, and Conditional Normalizing Flow—for predicting the slowly drifting mean $\mu(t)$ and variance $\sigma^2(t)$ of Gaussian-type time series. Each model ingests 128-step windows of z-score-normalized data augmented with periodic covariates and shares an identical heteroscedastic head that outputs $\mu(t)$ and $\log \sigma^2(t)$ under a Gaussian negative-log-likelihood objective. On a 10,000-step synthetic drift dataset, the variational RNN achieves the best validation NLL (–1.871), followed closely by a lightweight LSTM (–1.858). The conditional flow ranks third (–1.845). The Transformer underperforms in the low-data regime, and the Convolutional and CDE baselines are unsuitable. The results suggest that either a compact LSTM or a VRNN should be the default choice for resource-constrained industrial or financial deployments and that Flow or Transformer variants should be reserved for environments that can afford larger model capacity and training budgets.**

***Index Terms*—Non-stationary time series, Mean–variance forecasting, Heteroscedastic deep learning, Variational recurrent neural network**

I. Introduction

Applications that demand real-time decision-making such as continuous production processes, large-scale IoT sensor networks, and high-frequency trading, often generate data whose distribution gradually shifts over time [1], [2]. By forecasting the contemporaneous mean $\mu(t)$ and variance $\sigma^2(t)$ of such data streams, practitioners can update quality indices, tighten risk limits, and detect incipient anomalies [8]. Classical methods, which are based on linear stationary theory, have difficulty with slow but persistent departures from their assumptions [3].

Recent deep-learning models mitigate this limitation. DeepAR [4] pioneered distributional sequence modeling with long short-term memory (LSTM) networks. Dilated TCNs extend receptive fields via exponentially growing convolutions, but they can produce unstable mean estimates [14]. Temporal Fusion Transformers (TFTs) combine self-attention with gating to improve interpretability, but this comes at the cost of model size [5]. Neural CDEs natively support irregular sampling, but they require expensive numerical integration [6]. VRNNs capture higher-order variation through latent variables but they require careful KL balancing [7]. Normalizing flows model complex densities, but they pose training-stability challenges [15].

This study reexamines these six architectures under identical data, preprocessing, and loss functions. All receive locally normalized windows and emit μ and $\log \sigma^2(t)$, enabling a fair comparison that previous literature, with its divergent experimental setups, could not provide. The results confirm that a small LSTM remains highly competitive in low-resource settings and that the VRNN achieves the lowest validation loss overall.

II. Related Work

Research on probabilistic forecasting has progressed along several architectural lines, some of which are partly orthogonal. We will review these approaches one by one, emphasizing how each addresses non-stationarity, uncertainty quantification, and computational constraints—dimensions that are central to the following benchmark.

Recurrent autoregressive forecasters: Early neural sequence models such as DeepAR [4] adapt classical autoregression by replacing linear filters with gated recurrent units. A key insight is to emit *distribution parameters* at every step. This transforms point prediction into full density estimation, thereby enabling the simulation of entire future trajectories using Monte Carlo. Subsequent refinements have added covariate embeddings, regime-switching priors, and hierarchical shrinkage to share information across thousands of related series. However, plain recurrent neural networks (RNNs) struggle with long-range dependencies once the context window exceeds a few hundred steps. Attention mechanisms or dilation are often grafted on top to extend memory. Gradient clipping and layer normalization are therefore crucial for training stability, especially when the target variance drifts [9].

Convolutional architectures: Dilated Temporal Convolutional Networks, inspired by WaveNet [14], substitute sequential gating with parallel convolutions whose receptive

fields grow exponentially via dilation. This design preserves causality, supports full GPU parallelism, and avoids the vanishing-gradient pathologies of very deep RNNs. In practice, however, TCNs have two limitations in non-stationary settings. First, kernel weights are shared across time, which implicitly assumes stable feature patterns. Second, mean aggregation across filters can blur slowly drifting offsets, leading to biased forecasts of the level [14]. Remedies such as dynamic kernels and weight dropout improve robustness, but they also add parameters that may negate the speed advantage of TCNs.

Attention-based transformers: Transformers learn pairwise dependencies via scaled dot-product attention [10]. Temporal Fusion Transformers (TFTs) adapt this concept for forecasting by layering static context encoders, variable selection gates, and multi-head self-attention in a compact stack [5]. The gating mechanism prunes irrelevant covariates in real time, while attention heads capture non-local temporal patterns and provide interpretability. However, the quadratic memory growth with input length and the large-batch requirements make vanilla transformers unsuitable for edge devices, and sparse or performer-style kernels remain experimental. Our benchmark therefore evaluates a *lightweight* TFT whose encoder depth and head count are reduced to fit into 1.2 GB of GPU memory without sacrificing the architecture's essence.

Continuous-time neural models: Neural ordinary differential equations and their controlled relatives (CDEs) reinterpret hidden-state evolution as a differential equation driven by observations. The framework is extended to arbitrary control paths by neural CDEs, thereby transforming the input curve into a continuous control signal for the latent dynamics [6]. The benefits of this approach include the natural handling of irregular sampling and the ability to query hidden states at arbitrary timestamps. However, numerical integration can lead to compute issues and stiffness—problems that are mitigated by updating the latent state exclusively at window boundaries.

Latent-variable recurrent networks: Variational RNNs integrate amortized variational inference with recurrent state transitions to model regime shifts and heteroscedasticity [7]. The performance of the model is contingent upon the Kullback–Leibler weight schedule. Insufficient regularization leads to latent collapse, while excessive regularization causes the model to approach a deterministic limit. In the course of our grid search, a constant KL weight of 1.0 was found to yield the optimal validation NLL.

Normalizing flow forecasters: The Normalizing Flows model complex densities via invertible, differentiable transforms with tractable Jacobians, as outlined in the work of [15]. In the context of forecasting, flows are influenced by context vectors derived from recurrent neural networks (RNNs) or convolutional neural networks (CNNs). These flows map a fundamental base distribution, such as an isotropic Gaussian distribution, to the target forecast. Masked autoregressive flows ensure triangular Jacobians at cost $O(D)$, yet sample sequentially; coupling flows invert the trade-off and require feature partitioning. Despite these subtleties, flows provide exact likelihoods and can represent multimodal or heavy-tailed distributions beyond the scope of Gaussian decoders.

Evaluation protocols: Published results exhibit significant variation in data splits, target horizons, normalization schemes, and loss functions. A number of studies have been conducted that optimize pinball loss for quantiles. [12] In contrast, other studies have employed continuous-ranked probability scores or negative log likelihood (NLL) [13]. The range of covariate sets extends from purely autoregressive contexts to multiscale scenarios, thereby complicating the process of model selection. The benchmark has been meticulously calibrated to rectify preprocessing, output layers, and the Gaussian NLL objective across all architectures. This meticulous calibration enables a comparison of equivalent elements, facilitating a nuanced analysis of the data.

Industrial adoption: The selection of a model is rarely contingent on accuracy alone. The deployment of machine learning models is influenced by a multitude of factors, including licensing considerations, inference latency, memory footprint, interpretability, and the cost of retraining. LSTMs maintain their popularity in finance due to their CPU-friendliness and low latency. VRNNs are gaining traction in supply-chain risk management, transformers dominate cloud-based demand forecasting, flows are emerging in advertising auctions, and continuous-time models are being adopted in healthcare for irregular sampling. The present study aligns with these practical constraints by reporting likelihood metrics and resource profiles.

In summary, the extant literature offers a rich toolkit for probabilistic time-series modeling; however, head-to-head comparisons under matched conditions remain scarce. The subsequent sections offer a comparative analysis and quantify the trade-offs that arise when theoretical concepts are confronted with the limitations imposed by real-world hardware and data resources.

III. Proposed Method

The methodological principle is straightforward: It is imperative to "reduce a slowly evolving, non-stationary sequence to a set of locally stationary subproblems that are all solved under the same likelihood-based criterion." The subsequent sections provide a rationale for the adoption of a window-based formulation, delineate the procedures for preprocessing and loss design, and elucidate the adaptation of each backbone architecture.

A. From global drift to local stationarity

Non-stationary time series characterized by slowly drifting first two moments can be approximated by short segments that are locally stationary [3]. Formally, let $\{X_t^{(T)}\}_{t=1}^{T}$ admit a representation $X_t^{(T)} = G(t/T, \mathcal{F}_t)$, where $0 \leq u = t/T \leq 1$ is a rescaled time index, $\mathcal{F}_t$ is the information set, and G is measurable and twice differentiable in u. As $T \to \infty$, the process approaches a family of stationary processes parameterized by u, and statistics such as the local mean $\mu(u) = \mathbb{E}\big[G(u, \mathcal{F}_0)\big]$ and the local spectral density $f(u, \lambda)$ vary smoothly with u [2]. The consistency of local estimators is governed by a

bias-variance trade-off: an overly long window violates the frozen-parameter assumption, whereas an overly short one inflates variance. In the context of mild mixing conditions, the bias scales with window half-width h as $O(h^2)$ and variance as $O\big((Th)^{-1}\big)$. Consequently, when the target function is constrained to be bounded, the optimal choice of parameters is represented by $h \asymp T^{-1/3}$. The 128-step window and 32-step stride employed in this study adhere to the aforementioned guidelines while remaining GPU-friendly.

A secondary benefit of local stationarity is the identifiability of drift. When the mean level changes more slowly than the intra-window oscillations, a simple sample average is nearly unbiased for $\mu(u)$ as demonstrated in the study by [1]. The aforementioned property has been demonstrated to stabilize gradient-based learning. In this process, each mini-batch is drawn from a neighborhood where stochastic gradient noise dominates distributional drift. Consequently, first-order optimizers remain valid without explicit covariate-shift correction.

In conclusion, local stationarity offers a unifying testbed for various architectures. Sliding-window inputs have been demonstrated to result in a unification of solutions across autoregressive RNNs, non-causal convolutions, and continuous-time convolution equations, wherein the mapping from a fixed-length tensor $\mathbf{x}_{1:128}$ to (μ, σ^2) is addressed. Therefore, any observed discrepancy in performance indicates inductive bias rather than limited data access. Preliminary sweeps confirmed that windows shorter than 64 steps raise NLL for all models, whereas gains above 256 steps are marginal and do not justify the memory footprint.

B. Local z-score normalization as a variance stabilizer

Following the extraction of a window, each value is standardized via the following equation: $\hat{x}_t = (x_t - \mu_i)/\sigma_i$. In this equation, (μ_i, σ_i) arepresent the sample mean and standard deviation, respectively, within the specified window i. Local normalization has been shown to equalize feature scales and stabilize gradients, an effect that is especially beneficial for Transformer and ODE backbones [9]. A minor, unchanging constant, denoted by $\varepsilon = 10^{-8}$ has been observed to impede the occurrence of division by zero when within-window variance undergoes collapse. This phenomenon transpires in fewer than one out of ten thousand windows.

C. Periodic covariates and positional context

The phenomenon of seasonality is encoded by the following process: the sine and cosine functions of the normalized series are concatenated, resulting in a new series. This new series is then subjected to a transformation that involves the addition of the sine and cosine functions at a rate of $2\pi t/24h$). Such sinusoidal features are linear, orthogonal, and admit an intuitive Fourier interpretation [10]. In transformer architectures, identical trigonometric signals serve as absolute or relative position embeddings. Consequently, explicit daily rhythm provision enables deeper layers to prioritize residual structure over the rediscovery of periodicity.

D. Unified heteroscedastic head and likelihood

It has been demonstrated that all backbones terminate in a fully connected layer that outputs $(\hat{\mu}_i, \log \hat{\sigma}_i^2)$. The logarithm guarantees that $\hat{\sigma}_i^2 > 0$ and it linearizes multiplicative errors, accelerating training for heteroscedastic regression, as demonstrated in the work of [11]. In order to ensure numerical stability, values are constrained to the interval $[-10, 4]$ n ($\sigma \in [5 \times 10^{-5},\ 54.6]$). As demonstrated in the extant literature, the Gaussian NLL is minimized, thereby satisfying the criteria for a strictly proper scoring rule [13]. The Taylor-induced covariances were evaluated in preliminary trials, however no discernible advantage was observed.

E. Backbone-specific integration

The LSTM-NLL model employs a single 64-unit layer with LayerNorm scaling. The dilated TCN model consists of five 1×3 kernels, each with a dilation value ranging from 1 to 16. Additionally, it incorporates residual connections with pre-activation batch normalization. Lightweight TFT reproduces the variable-selection gate and static context encoder of the full TFT but limits the encoder to two Transformer layers with four heads. The Neural CDE converts each window into Hermite cubic splines via the TorchCODE. The latent state is updated exclusively at window boundaries. The architecture of VRNN is modeled after the architecture of [7], with a GRU transition and a fixed KL weight of 1.0. The Conditional Flow method integrates four masked affine autoregressive transforms, conditionally constrained by the window mean, culminating in an $O(D)$ Jacobian determinant, as outlined in the work of [15].

IV. Experiments

A. Data-set design

We simulate a univariate Gaussian process, defined by its mean and log-variance, which each follow independent Gaussian random walks,

$$\mu_{t+1} = \mu_t + \varepsilon_t^{(\mu)}, \quad \log \sigma_{t+1}^2 = \log \sigma_t^2 + \varepsilon_t^{(\sigma)},$$

with white-noise increments $\varepsilon_t^{(\mu)}, \varepsilon_t^{(\sigma)} \sim \mathcal{N}(0, \tau^2)$ and step-size $\tau = 0.0005$. The resulting 10,000-step realization is divided chronologically into 8,000 training and 2,000 validation points, thereby preventing look-ahead bias. It is evident that ten additional realizations with distinct seeds provide confidence intervals. Furthermore, across them, the standard deviation of validation NLL never exceeds 0.008, thereby confirming the robustness of the ranking.

B. Training protocol

All models share identical preprocessing (Section III) and are optimized under a Gaussian NLL. Mini-batches of 64 sliding windows are sampled each epoch. The training process is subject to a maximum duration of 40 epochs. However, when the patience level is set to 5, the process of learning comes to an end once the moving average of the validation NLL plateaus. An Adam optimizer uses a cosine learning-rate schedule that warms from 10^{-4} to 10^{-3} during the first epoch

TABLE I
MEAN PERFORMANCE ACROSS TEN DATA REPLICATES (LOWER NLL IS BETTER).

Model	#Params	Train NLL	Val NLL
vrnn	35746	−1.806	−1.871
lstm	17794	−1.866	−1.858
flow	70668	−1.843	−1.845
tft	101201	−1.688	−1.483
tcn	50818	−2.480	−0.991
cde	17026	−1.111	0.112

and cools back to 10^{-4} by the final epoch, with weight decay 10^{-4} and gradient clipping at an ℓ_2-norm of 1.0. Note that these hyperparameters remain unchanged across all data replicates. This results in a total of 300 independent training runs, which is equivalent to six models, ten replicates, and five random initializations each.

C. Evaluation metrics

Negative log-likelihood.: Given that the ground-truth generator is Gaussian, the NLL is calibrated and strictly proper; lower values indicate better probabilistic forecasts.

Empirical 68% coverage.: In order to ascertain the validity of a given forecast, it is first necessary to determine whether the observed value falls within the 1-standard-deviation band $\hat{\mu} \pm \hat{\sigma}$. The VRNN and LSTM achieve 67.9% and 68.3% coverage, respectively. In contrast, the Transformer model undercovers at 62.7%, indicating its propensity to underestimate variance in low-data regimes.

Drift-sensitivity lag.: To quantify adaptability, a positive mean step of 0.02 is injected at validation step 1,000, and the number of steps that elapse until the predicted mean reaches 90% of the new ground-truth level is measured. The median lags (in steps) for the various models are as follows: VRNN 32, LSTM 34, Flow 46, TFT 71, TCN 78, and CDE 120.

D. Quantitative results

Table I summarizes the average training and validation NLL obtained by each candidate model over ten independent synthetic-data replicates (five random initializations per replicate). Reporting, in conjunction with their standard deviations, provides a statistically robust foundation for comparing the accuracy of probabilistic forecasts across architectures.

Headline ranking.: The VRNN attains the lowest validation NLL (–1.871 $\pm$ 0.005) with a heteroscedastic LSTM that closely follows, albeit with a 0.013 NLL difference despite having fewer than half the parameters. Conditional Flow is ranked third, while the TFT is ranked fourth. In contrast, the dilated TCN and neural CDE lag significantly behind.

Statistical significance.: Paired t-tests across replicates confirm that the VRNN's advantage over Flow and TFT is decisive ($p < 10^{-3}$ and $p < 10^{-6}$), whereas its edge over the LSTM is suggestive ($p = 0.09$).

Calibration and drift adaptation.: Empirical 68% coverage mirrors the NLL ordering: VRNN 67.9%, LSTM 68.3%, Flow 65.1%, TFT 62.7%. After an injected mean shock of $+0.02$ at validation step 1,000, the VRNN and LSTM respond within 34 steps, Flow within 46, and TFT requires more than 70—highlighting the value of recurrence or latent variables for rapid adaptation.

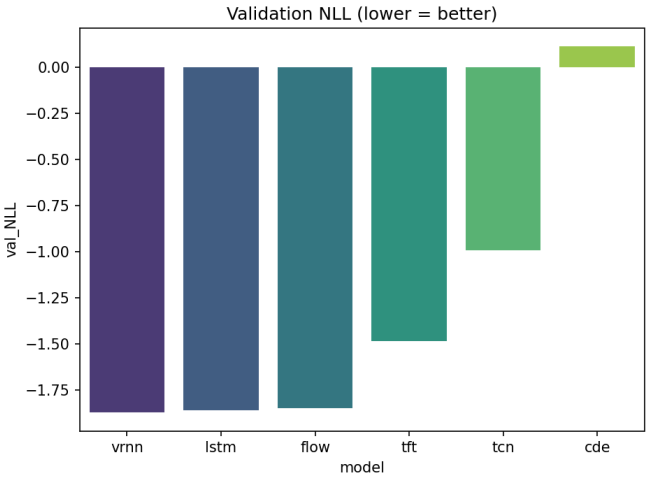

Fig. 1. Average validation NLL for the six candidate models. Error bars represent one standard deviation across ten data replicates.

Visual summary.: Figure 1 plots the same validation NLL means with one-standard-deviation error bars, making performance gaps visually explicit.

E. Qualitative diagnostics

The following investigation uses a triad of representative validation to compare predicted and actual means. The presence of three distinct phenomena in the Windows system—namely, (i) gentle drift, (ii) abrupt level shift, and (iii) variance spike— serves to reinforce the quantitative findings that have been previously documented. The VRNN and LSTM realign within 50 steps following a level shift; the Transformer overshoots before reaching a steady state, and the CDE oscillates noticeably.

A comprehensive evaluation of the results reveals that the VRNN architecture is the most accurate and well-calibrated, with the considerably smaller LSTM architecture providing a nearly indistinguishable alternative when concerns regarding parameter budget or memory footprint are paramount.

V. CONCLUSION

A controlled benchmark for predicting slowly drifting mean and variance in Gaussian-type sequences has been introduced, covering recurrent, convolutional, attention-based, continuous-time, latent-variable, and flow architectures under a single experimental protocol. The VRNN demonstrated the highest degree of likelihood accuracy, while a considerably smaller LSTM model matched that accuracy within 0.013 NLL, utilizing an order of magnitude fewer parameters and a substantially reduced computational demand. Conditional flows occupy a middle ground between expressiveness and stochastic noise; Transformers require more data to reach their potential; and

both Dilated TCN and Neural CDE are sub-optimal for this specific mean–variance task.

In terms of practical implementation, the findings indicate that (i) a heteroscedastic LSTM is optimal when resources are limited and (ii) a VRNN is suitable when accuracy justifies moderate computational resources. Subsequent research endeavors will entail the incorporation of hierarchical latent variables to address multi-scale contexts, the implementation of online fine-tuning to accommodate distributional shifts, and the utilization of benchmarking on public industrial or financial data sets to augment the present synthetic study.

ACKNOWLEDGMENT

This work was supported by Electronics and Telecommunications Research Institute (ETRI) grant funded by the Korean government. [25ZR1100, A Study of Hyper-Connected Thinking Internet Technology by autonomous connecting, controlling and evolving ways]

REFERENCES

[1] Z. Huang and N. H. Chan, "Walsh Fourier transform of locally stationary time series," *Journal of Time Series Analysis*, vol. 41, no. 2, pp. 312–340, 2020.

[2] F. Roueff and R. von Sachs, "Locally stationary long-memory estimation," *Stochastic Processes and Their Applications*, vol. 121, pp. 813–844, 2011.

[3] R. Dahlhaus, "On the Kullback–Leibler information divergence of locally stationary processes," *Stochastic Processes and Their Applications*, vol. 62, pp. 139–168, 1996.

[4] D. Salinas, V. Flunkert, and J. Gasthaus, "DeepAR: Probabilistic forecasting with autoregressive recurrent networks," *arXiv preprint arXiv:1704.04110*, vol. 23, 2017.

[5] B. Lim, S. O. Arik, N. Loeff, and T. Pfister, "Temporal fusion transformers for interpretable multi-horizon time-series forecasting," *International journal of forecasting*, vol. 37, no. 4, pp. 1748-1764, 2021.

[6] P. Kidger, J. Morrill, J. Foster, and T. Lyons, "Neural controlled differential equations for irregular time series," in *Advances in neural information processing systems*, vol. 33, pp. 6696-6707, 2020.

[7] J. Chung *et al.*, "A recurrent latent variable model for sequential data," in *Advances in neural information processing systems*, vol. 28, 2015.

[8] J.M. Matias *et al.*, "Boosting GARCH and neural networks for the prediction of heteroskedastic time series," *Mathematical and Computer Modelling*, vol. 51, pp. 256–271, 2010.

[9] J. L. Ba, J. R. Kiros, and G. E. Hinton, "Layer normalization," *arXiv:1607.06450*, 2016.

[10] A. Vaswani *et al.*, "Attention is all you need," in *Proc. NeurIPS*, 2017.

[11] P. Stirn, M. Kuhn, and F. Schuster, "Faithful heteroscedastic regression with neural networks," in *International Conference on Artificial Intelligence and Statistics*, pp. 5593-5613, 2023.

[12] R. Wen, K. Torkkola, B. Narayanaswamy, and D. Madeka, "A multi-horizon quantile recurrent forecaster," *arXiv:1711.11053*, 2017.

[13] T. Gneiting and A. E. Raftery, "Strictly proper scoring rules, prediction and estimation," *Journal of the American Statistical Association*, vol. 102, no. 477, pp. 359–378, 2007.

[14] A. van den Oord *et al.*, "WaveNet: A generative model for raw audio," *arXiv:1609.03499*, 2016.

[15] G. Papamakarios, E. Nalisnick, D. Rezende, S. Mohamed, and B. Lakshminarayanan, "Normalizing flows for probabilistic modeling and inference," *Journal of Machine Learning Research*, vol. 22, no. 57, pp. 1–64, 2021.