SIGNAL OR NOISE? UNDERSTANDING GENERATIVE MODELS FOR REAL-WORLD SENSOR TIME SERIES

Anonymous authors
Paper under double-blind review

ABSTRACT

Generative models have changed how machine learning represents complex data distributions, especially in language and vision, yet many real-world systems are observed instead as continuous, high-dimensional, and noisy *sensor time series*. Existing generative modeling of sensor data, however, remains fragmented across modalities, datasets, and task formulations, limiting a systematic understanding of *when*, *how*, and *why* generative models succeed or fail in real-world settings. To address this gap, we introduce `SensorGen`, a large-scale study of sensor-signal generation spanning 14 settings across 4 domains, 7 datasets, and 12 signal modalities. Leveraging `SensorGen`, we systematically evaluate generative models from five major families and uncover three key findings: (1) flow-matching models provide strong overall performance across most settings; (2) signal properties matter, with demographic covariates improving longitudinal generation and time-frequency modeling improving high-frequency signal generation; and (3) generated signals have practical utility beyond visual realism, with scaling improving generation quality and synthetic data improving downstream performance. Together, `SensorGen` establishes a broader understanding of design choices, evaluation protocols, and failure modes in real-world sensor data generation.

1 INTRODUCTION

Generative models have reshaped how machine learning imagines data: learning not only to recognize patterns, but to sample, complete, and represent complex distributions (Peebles & Xie, 2023; Ho et al., 2020). Yet, much of the real world is not expressed as language or images, but as continuous, noisy, high-dimensional ***sensor time series*** that record physiology (Xu et al., 2026), behavior (Chan et al., 2024), environments (Nguyen et al., 2023), and machines (Khazatsky et al., 2024). This makes sensor time series a central data regime for modern machine learning, but extending generative modeling advances to sensor signals is non-trivial. Unlike language or natural images, sensor signals are collected across diverse settings and modalities, and vary widely in sampling frequency, time span, sequence length, channel structure, and physical semantics.

Existing studies for sensor time series, however, remain highly *fragmented* across modalities, datasets, and task formulations (Lai et al., 2025; Ding & Wu, 2024; Ibtehaz et al., 2022). Current sensor generation methods are often designed for specific applications, with modeling paradigms, architectures, and evaluation protocols tailored to a particular signal type or task. While effective in their target settings, these dedicated designs provide limited evidence about whether the modeling choices generalize across heterogeneous sensor generative problems. As a result, the field lacks a systematic understanding of the factors that shape generation quality across sensor signals. This motivates a central question:

When, how, and why do generative models succeed or fail across diverse sensor time series generation settings?

To answer this question, we introduce `SensorGen`, a fully open and large-scale exploration of generative modeling for real-world sensor time series. Rather than introducing another task-specific generator, `SensorGen` is designed to disentangle how model families, signal properties, task formulations, scale, and evaluation protocols jointly shape generation quality (Fig. 1): ❶ **Task coverage.** We construct practically meaningful generation tasks from real-world sensor datasets and standardize them into a unified pipeline for data processing, task construction, model training, and evaluation.

Reviewers: please read the Reviewer Guidelines (iclr.cc/Conferences/2027/ReviewerGuidelines) and the AI Policy for Reviewers (iclr.cc/Conferences/2027/AIPolicyForReviewers). If you used AI to expand, edit, or polish your review, please provide the input text to the LLM. Better still, consider skipping the LLM and submitting your original text: we, and the authors, are much more interested in your unedited thoughts than in what an LLM has to say. AI-assisted or not, you are putting your name and reputation behind your review: LLM-generated falsehoods, hallucinations or misrepresentations are subject to disciplinary action, which may include desk-rejecting all papers you have authored.

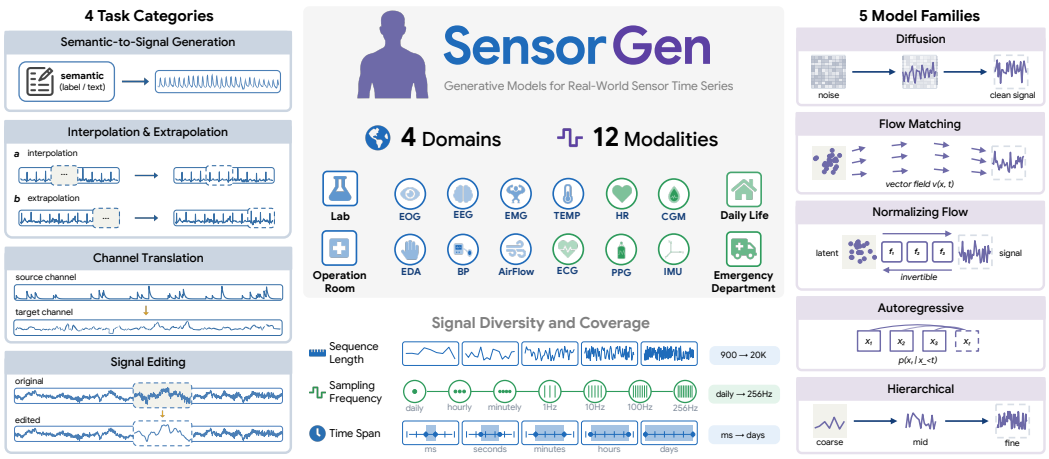

Figure 1: **Overview of `SensorGen`.** We present a large-scale study of real-world sensor time-series generation, spanning 14 settings across 4 domains, 7 datasets, and 12 signal modalities. `SensorGen` establishes to date the broadest coverage of sequence length, frequency, and time span. More details are in Appendix B.1.

❷ Sensor diversity. `SensorGen` spans 14 generation settings across 4 domains, 7 datasets, and 12 signal modalities, covering diverse sampling frequencies, sequence lengths, and time spans. **❸ Generation settings.** We organize these settings into 4 major categories: *semantic-to-signal generation*, *sensor interpolation and extrapolation*, *channel translation*, and *signal editing*. **❹ Model families.** We evaluate six representative models from five major generative families, including diffusion models, flow-matching models, autoregressive models, normalizing-flow models, and hierarchical models.

Through `SensorGen`, we reveal intriguing lessons for building generative models for sensor signals. First, *model family matters*: flow-matching models consistently provide strong performance across most settings, suggesting a generally strong baseline. Second, *sensor-property-aware design matters*: demographic covariates improve longitudinal generation, while time-frequency modeling improves high-resolution signal generation. Third, *generated signals are useful beyond realism*: scaling generally improves generation quality, and synthetic data improves downstream task performance under data scarcity. Beyond these findings, we further analyze design choices, evaluation approaches, and failure modes, providing a broader understanding of key gaps and directions for future research. Our contributions are as follows:

- We introduce `SensorGen`, the first open large-scale study of sensor time-series generation, spanning 14 settings across 4 domains, 7 datasets, and 12 signal modalities, with broad coverage of sequence length, sampling frequency, and time span.
- We establish a unified exploration framework that consolidates fragmented sensor-generation tasks, modeling paradigms, and evaluation protocols, covering 4 generation categories and five representative generative model families.
- `SensorGen` provides a comprehensive guideline for developing and evaluating generative models for real-world sensor time series. Through extensive controlled analyses of design choices, evaluation protocols, and failure modes, we uncover practical principles for sensor generation: flow matching provides a strong baseline; signal-aware designs improve challenging regimes; and scaling and synthetic data improve generation quality and downstream utility.

2 Related Work

Modern Generative Modeling. Generative models have become a central approach for learning and sampling complex data distributions. Recent advances span several representative families, including diffusion models (Peebles & Xie, 2023; Ho et al., 2020), flow-matching models (Ma et al., 2024; Esser et al., 2024), autoregressive models (Li et al., 2024; Sun et al., 2024), normalizing flows (Zhai et al., 2024), and hierarchical generative models (Saharia et al., 2022; Li et al., 2025b). These families have achieved strong performance in high-fidelity image synthesis (Dhariwal & Nichol, 2021) and language generation (Brown et al., 2020), making them natural candidates for studying generation beyond language and vision. However, existing evidence is largely concentrated in these domains, leaving open which model families are most effective for sensor data generation. Unlike language or

Table 1: **Comparison of studies on generative modeling of sensor time series.**

Study	# Domain	# Dataset	# Modality	Model Family					Task Type			
				Diff	FM	NF	AR	Hier	S2S	IntExt	Trans	Edit
Vetter *et al.* Vetter et al. (2024)	1	4	3	✓	✗	✗	✗	✗	✓	✓	✗	✗
Oppel *et al.* Oppel & Munz (2025)	1	1	1	✓	✗	✗	✗	✗	✓	✗	✗	✗
Bondar *et al.* Bondar et al. (2025)	1	1	1	✗	✓	✗	✗	✗	✓	✗	✗	✗
Fang *et al.* Fang et al. (2025)	1	4	2	✗	✓	✗	✗	✗	✗	✗	✓	✗
Neifar *et al.* Neifar et al. (2024)	1	1	1	✓	✗	✗	✗	✗	✓	✓	✗	✗
Chen *et al.* Chen et al. (2026)	1	4	3	✓	✗	✗	✗	✗	✗	✓	✓	✓
`SensorGen` **(ours)**	**4**	**7**	**12**	✓	✓	✓	✓	✓	✓	✓	✓	✓

`Diff`: Diffusion. `FM`: Flow-Matching. `NF`: Normalizing Flow. `AR`: Autoregressive. `Hier`: Hierarchical Modeling.
`S2S`: Semantic-to-Signal. `IntExt`: Interpolation & Extrapolation. `Trans`: Translation. `Edit`: Signal Editing / Inverse Problems.

natural images, sensor signals are continuous, noisy, temporally structured, and highly heterogeneous in sampling frequency, time span, sequence length, channel structure, and physical semantics (Li et al., 2026), making direct adaptation to sensor time series non-trivial.

Generative Models for Sensor Time Series. Recent work has leveraged generative models for practical sensor-signal tasks, including synthesizing ECG signals to improve sample diversity or augment rare conditions (Lai et al., 2025; Hu et al., 2024; Bondar et al., 2025; Neifar et al., 2024), translating PPG signals into ECG or blood-pressure waveforms (Ding & Wu, 2024; Shome et al., 2024; Fang et al., 2025), generating IMU signals (Oppel & Munz, 2025), forecasting vital signs and wearable signals (Ibtehaz et al., 2022; Chen et al., 2026), synthesizing realistic neurophysiological time series (Vetter et al., 2024), imputing missing measurements (Kazijevs & Samad, 2023), and editing or denoising physiological signals (Jang et al., 2026). However, as shown in Table 1, existing works are often developed for specific modalities, datasets, or applications, with task-specific designs, conditioning strategies, preprocessing pipelines, and evaluation protocols. In contrast, `SensorGen` presents an in-depth study of when, how, and why generative models succeed or fail across sensor-signal generation settings, unifying diverse tasks and representative model families under a shared protocol to analyze how model choices, signal properties, and task formulations shape generation quality.

3 SensorGen

`SensorGen` bridges the fragmented landscape of sensor signal generation into a unified foundation. Instead of building a single best generative model, we study sensor generation as a joint problem over *task formulations*, *sensing domains*, *signal properties*, and *model families*. This is essential as generation quality can vary substantially across modalities with different sampling frequencies, temporal spans, sequence lengths, channel structures, and physical semantics (Li et al., 2026).

3.1 Data Regimes

Dataset diversity and domain coverage. A central design of `SensorGen` is to organize sensor generation around the major real-world regimes in which sensor data are collected and used, rather than simply aggregating datasets by availability. We therefore define four broad, distinct, and representative collection scenarios: *emergency-department* monitoring, *daily-life* sensing, *laboratory* studies, and *operation-room* monitoring. We then instantiate these scenarios with 7 widely used public datasets, spanning applications in everyday human behavioral sensing, health monitoring, and clinical care, and capturing signals related to physiological, emotional, behavioral, and metabolic states.

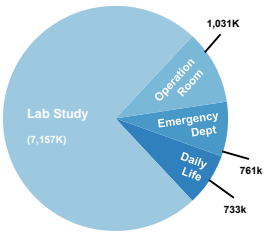

Figure 2: **Domain coverage of `SensorGen`.**

- ***Emergency Department***: Emergency-department ECG datasets, including MIMIC-IV ECG (Gow et al., 2023), provide clinical 12-lead electrocardiogram recordings paired with diagnostic reports.
- ***Daily Life***: Daily-life sensing datasets span CAPTURE-24 (Chan et al., 2024), PPG-DaLiA (Reiss et al., 2019), and Metabonet (Wolff et al., 2026), provide free-living wearable physiological measurements such as PPG, ECG, and accelerometry, and longitudinal metabolic traces such as CGM, insulin, and carbohydrate intake.

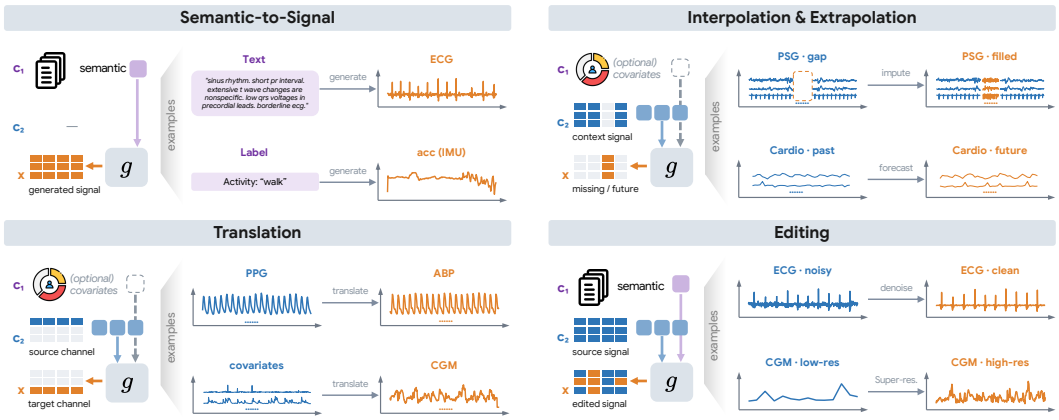

Figure 4: **Illustration of task categories in** `SensorGen`**.** For each category, we show the task-level data flow (left) and representative generation settings with example channels (right).

- ***Lab Study***: Signals collected from laboratory monitoring environments and sleep-study datasets, including PhyMER (Pant et al., 2023) and SHHS (Zhang et al., 2018; Quan et al., 1997), provide controlled affective-state recordings and overnight polysomnography, covering EEG, ECG, EMG, EOG, and peripheral physiological signals.
- ***Operation Room***: VitalDB (Lee et al., 2022) provides intra-operative physiological waveforms, medication-rate records, arterial blood pressure, PPG, ECG, and NIBP-derived measurements.

Broad coverage of sensor modalities. `SensorGen` includes 12 signal modalities spanning bioelectrical recordings, optical and hemodynamic measurements, wearable motion signals, intra-operative waveforms, intervention records, and longitudinal metabolic traces. These modalities differ not only in physical origin, channel structure, and physiological semantics, but also in how they are sampled, measured, and observed in practice. As a result, `SensorGen` spans diverse signal properties, including *sequence length*, *sampling frequency*, and *time span*. Across tasks, sequence lengths range from hundreds to tens of thousands of steps, sampling frequencies span from one sample every five minutes to 256 Hz, and observation windows range from seconds to a week.

3.2 Task Taxonomy

Design Principles and Formulation. We design `SensorGen` around real sensing bottlenecks where generative modeling is practically useful. Each setting is grounded in clinical practice, domain knowledge, or prior empirical findings; addresses a practical data bottleneck or deployment need; and contains sufficient temporal resolution and structure for meaningful generation.

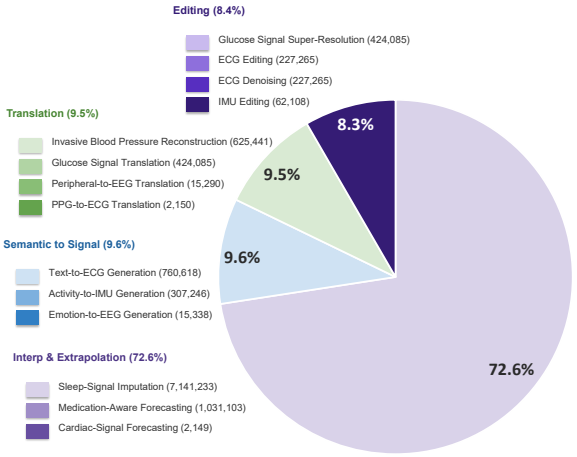

Figure 3: **Task coverage of** `SensorGen`**.**

Following these principles, we construct 4 categories of generation: ❶ *Semantic-to-Signal*, addressing the scarcity of diverse and semantically annotated signal samples; ❷ *Interpolation & Extrapolation*, targeting future prediction and reconstruction of temporally incomplete signals; ❸ *Translation*, reconstructing missing or hard-to-acquire channels from accessible sources; and ❹ *Editing*, enabling constrained restoration or counterfactual modification of existing recordings. Fig. 3 summarizes its coverage breadth.

We formulate all settings as conditional generation. Let $x \in \mathbb{R}^{C \times T}$ denote the target signal, c_1 semantic or metadata conditions, and c_2 observed signal context. Each task takes the form $\widehat{x} \sim p_\theta(x \mid \mathcal{C})$, where $\mathcal{C} \subseteq \{c_1, c_2\}$. Semantic-to-signal uses c_1, interpolation/extrapolation and translation

Table 2: Results on *Translation*.

Model	Translation		
	MSE↓	MAE↓	SMSE↓
DiT	0.224	0.343	0.044
SiT	**0.214**	**0.336**	**0.041**
FractalGen	0.357	0.454	0.099
TarFlow	0.382	0.482	0.097
MAR	0.387	0.483	0.096

Table 3: Results on *Interpolation & Extrapolation*.

Model	Interpolation & Extrapolation		
	MSE↓	MAE↓	SMSE↓
DiT	0.335	0.448	0.096
SiT	**0.252**	**0.384**	**0.064**
FractalGen	0.345	0.413	0.070
TarFlow	0.353	0.450	0.083
MAR	0.351	0.431	0.101

primarily use c_2, optionally with c_1, and editing conditions on an observed signal with optional semantic guidance. Detailed task formulations and specific settings are provided in Appendix B.1.

Semantic-to-Signal synthesizes signals from semantic descriptions c_1, including *Text-to-Signal Generation* from free-form text and *Label-Conditioned Generation* from discrete labels.

Interpolation & Extrapolation reconstructs unobserved portions of a signal using temporal context c_2. This includes *Future Forecasting*, *Intervention-Conditioned Forecasting*, and *Temporal Imputation*. Optional auxiliary intervention information is provided through c_1,

Translation. Given source channels or proxy measurements c_2 and optional auxiliary conditions c_1, models reconstruct target channels through *signal translation* or *proxy-to-target reconstruction*.

Editing starts from an observed signal c_2 and modifies or restores it toward a desired target, with optional guidance c_1. We consider *Super-Resolution*, *Denoising*, and *Semantic Signal Editing*.

3.3 Model Families

Our goal is to compare mainstream generative model families under a unified sensor-generation setting rather than develop task-specific models for individual datasets or applications. We therefore select modern, representative families that are widely used for high-fidelity generation, span distinct generative mechanisms, and are applicable across our task taxonomy (Dhariwal & Nichol, 2021; Peebles & Xie, 2023; Song et al., 2020; Saharia et al., 2022). We summarize these families below and provide implementation details in Appendix F.

- **Diffusion Models.** Diffusion models generate samples through iterative denoising from noise to data. We instantiate this family with DiT (Ho et al., 2020; Peebles & Xie, 2023).
- **Flow Matching.** Flow-matching models learn a continuous transport process from noise to data. We instantiate this family with SiT (Ma et al., 2024).
- **Autoregressive Models.** Autoregressive models generate tokens sequentially by conditioning each step on previously generated tokens. We instantiate this family with MAR (Li et al., 2024).
- **Normalizing Flows.** Normalizing flows model data through invertible likelihood-based transformations. We instantiate this family with TarFlow (Zhai et al., 2024).
- **Hierarchical Models.** Hierarchical models generate samples through multi-scale or coarse-to-fine synthesis. This family is instantiated with FractalGen (Li et al., 2025b) and Imagen (Saharia et al., 2022).

4 Main Results

Data setup. We use a unified preprocessing pipeline that standardizes heterogeneous sensor datasets into a shared data representation, enabling consistent task construction, model training, and evaluation across domains and modalities. Unless otherwise specified, each loaded window is normalized independently with min–max scaling to $[-1, 1]$, which stabilizes training for generative modeling. Detailed preprocessing and data-loading procedures are provided in Appendix A.

Training Setup. We benchmark representative generative model families for sensor time series. All methods share unified task definitions, preprocessing pipelines, and data splits, and are trained from scratch for each task setting with matched batch sizes, training budgets, and comparable model capacities when applicable. We train each run for up to 50K steps or until convergence, using a

Table 4: Results on *Semantic-to-Signal*.

Model	Semantic-to-Signal		
	FID↓	Precision↑	Recall↑
DiT	**2.05**	0.887	0.363
SiT	2.23	**0.915**	**0.413**
FractalGen	7.27	0.632	0.073
TarFlow	4.26	0.617	0.111
MAR	7.76	0.747	0.158
Imagen	2.51	0.734	0.346

Table 5: Results on *Signal Editing*.

Model	Editing		
	MSE↓	MAE↓	SMSE↓
DiT	0.059	0.151	0.032
SiT	0.134	0.260	0.088
FractalGen	0.127	0.256	0.042
TarFlow	0.034	0.144	0.008
MAR	0.266	0.393	0.121
Imagen	**0.008**	**0.064**	**0.002**

standardized training protocol without extensive method-specific hyperparameter tuning. Additional implementation details are provided in Appendix E.1.

Evaluation Protocol. For each method and setting, we randomly generate 1,024 samples from held-out test conditions and report task-specific metrics (see Appendix B.2). For semantic-to-signal generation, we compute distribution-level metrics in the feature space of MIRA (Li et al., 2025a), a representation model pre-trained on diverse healthcare time-series data. For paired generation tasks, we report sample-level metrics, including MSE, MAE, and spectrogram-domain MSE (SMSE). We follow (Shome et al., 2024) to calculate these metrics based on the normalized signal.

4.1 Which generative paradigm transfers best to sensor time series?

Model family matters, with flow matching emerging as the strongest default across tasks. We first compare representative generative models across all task categories. For each metric, we aggregate results within each task category and report category-level averages. Overall, SiT provides the strongest performance, although no single method dominates every setting. As shown in Table 2 and 3, SiT achieves the best aggregate performance on both Translation and Interpolation & Extrapolation tasks, consistently outperforming other methods across the reported metrics.

For Semantic-to-Signal, Table 4 shows that SiT also achieves better overall distribution-level fidelity. In contrast, Editing tasks are more setting-dependent: Table 5 shows that Imagen remains competitive on training-required editing, while Table 6 confirms SiT performs best in training-free editing settings. Full setting-level results are provided in Appendix C.

These results suggest that flow matching is the most reliable default choice for sensor-signal generation in our study. At the same time, the remaining methods show task-dependent strengths, indicating that modern generative paradigms do not transfer uniformly to sensor signals. Their inductive biases may align differently with sensor-specific requirements, including temporal dynamics, spectral structure, and conditional consistency. We next analyze how these signal properties shape model behavior.

Table 6: Results on training-free *Signal Editing*.

Model	Editing (Training-Free)		
	MSE↓	MAE↓	SMSE↓
DiT	0.160	0.266	0.044
SiT	**0.155**	**0.250**	**0.043**

Takeaway 1: Flow-matching models provide a strong baseline for sensor time series generation.

4.2 What context matters for generating long-horizon sensor data?

Subject context helps longitudinal generation only when encoded carefully. We next examine which factors matter for generating longitudinal sensor signals over extended time spans. Motivated by strong subject-level heterogeneity in long-term physiological patterns (Yang et al., 2022; Zhang et al., 2025), we study whether demographic covariates provide useful global context for generation.

As shown in Table 7, directly using raw demographic attributes substantially degrades performance compared with using no demographic information. In contrast, dataset-level normalization makes demographic conditioning beneficial, outperforming the no-demographic baseline. These results suggest that subject-level metadata can improve longitudinal generation, but only when encoded in a properly normalized form.

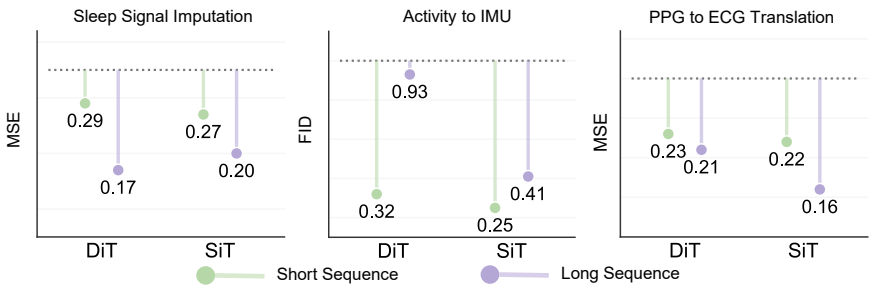

Figure 5: **Comparison of long-sequence generation.** Models struggle with long signals in semantic-to-signal generation, but outperform on imputation and translation when sensor context is available.

Takeaway 2: Normalized demographic covariates improve longitudinal sensor-signal generation.

4.3 Do generative models scale to longer sensor sequences?

Generative models can handle long sensor sequences when target-signal context is available. We next examine how sequence length affects generation quality, since long sensor sequences require models to maintain temporal consistency over extended horizons. To this end, we evaluate paired short- and long-sequence settings: *SHHS* sleep-signal imputation and *Capture-24* activity-conditioned IMU generation use sequence lengths of 1,920 *vs.* 19,200 and 900 *vs.* 9,000, while *PPG-to-ECG translation* uses sequence lengths of 1,000 *vs.* 6,000. For comparable evaluation, different from settings in the main table, we split each generated longer signal into slices matching the short-sequence duration, compute metrics on each slice, and report the average.

As shown in Fig. 5, generation quality degrades as sequence length increases for semantic-to-signal generation, but remains stable in the channel translation and imputation setting. This contrast suggests that long-sequence generation is more tractable when target-signal context is available: surrounding observed segments can anchor the missing region and reduce the burden of generating long-range structure from condition information alone.

Takeaway 3: Target-signal context reduces degradation in long-sequence generation.

4.4 How should models capture high-frequency signal semantics?

High-frequency signal semantics are difficult to capture, while time-frequency modeling improves fidelity. We next examine high-frequency signal generation, where semantic fidelity depends not only on waveform morphology but also on spectral structure. As shown in Table 15, for high-frequency signals such as EEG, all models achieve relatively low recall despite high precision and realistic morphology in qualitative examples, as shown in Appendix D. This suggests that current generative models can produce plausible high-frequency waveforms, but still struggle to cover the semantic and spectral variability of EEG signals.

To test whether explicit time-frequency structure can mitigate this limitation, we conduct an ablation on the emotion-to-EEG setting. We extend the input with the spectrogram to verify whether additional frequency-domain guidance leads to performance improvement. Specifically, we transform raw EEG waveforms into spectrograms using the short-time Fourier transform (STFT), and use the spectrogram as a time-frequency conditioning signal. As shown in Fig. 8, incorporating this representation improves the fidelity of high-frequency channel generation.

Table 7: **Comparison of covariate encoding strategies.** We report glucose signal translation results.

Model	Encoding	MSE↓	MAE↓	SMSE↓
DiT	Raw	0.767	0.744	0.578
DiT	Z-score	**0.097**	**0.223**	**0.032**
DiT	No demo	0.106	0.230	0.032
SiT	Raw	0.272	0.420	0.158
SiT	Z-score	**0.091**	**0.216**	**0.030**
SiT	No demo	0.101	0.228	0.032

Takeaway 4: High-frequency signal generation benefits from time-frequency modeling.

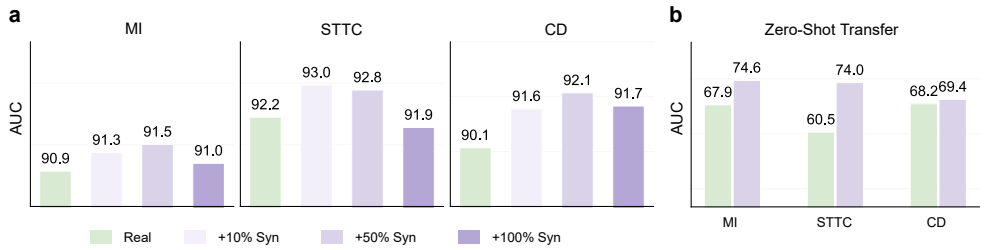

Figure 6: **Downstream utility of generated signals.** With ECGs synthesized by a text-to-ECG generator, we evaluate their utility in two downstream settings. **(a)** Moderate synthetic augmentation improves disease classification performance, whereas excessive synthetic data degrades it. **(b)** Adding synthetic data improves zero-shot transfer in ECG-text representation learning.

Qualitative Results. Fig. 7 further shows that the generated signals preserve meaningful physiological structure rather than merely resembling ECG waveforms. For example, PVC-conditioned generations contain predominantly regular sinus beats interrupted by premature, wider complexes; RBBB generations exhibit consistently broadened QRS complexes; and AF generations show irregular RR intervals with reduced organized P-wave activity. More visualizations are provided in Appendix D across other settings, demonstrating that generated samples capture recognizable signal structure rather than degenerating into noise. These examples qualitatively complement our quantitative results, demonstrating that modern generative models can capture diverse and condition-relevant structure in real-world sensor signals.

5 Analysis

Are generated sensor signals useful beyond visual realism? We evaluate whether generated signals provide utility beyond sample realism by using them to augment self-supervised representation learning and supervised disease prediction. For self-supervised learning, we follow the ECG-text pre-training setting in (Wang et al., 2025), comparing encoders pretrained with real data alone versus a mixture of real and synthetic data. We transfer the resulting models to PTB-XL (Wagner et al., 2020) and evaluate zero-shot classification. As shown in Fig. 6, adding synthetic data improves the downstream performance of Myocardial Infarction (MI) , ST/T Change (STTC) , and Conduction Disturbance (CD) detection tasks , indicating that generated signals can augment pre-training data and improve representation quality.



Figure 7: **Qualitative examples of Semantic-to-Signal generation.** ECG signals generated by DiT and SiT from clinical descriptions of premature ventricular contractions (PVCs), right bundle branch block (RBBB), and atrial fibrillation (AF). The examples illustrate diverse waveform morphology and rhythm characteristics captured by generative models.

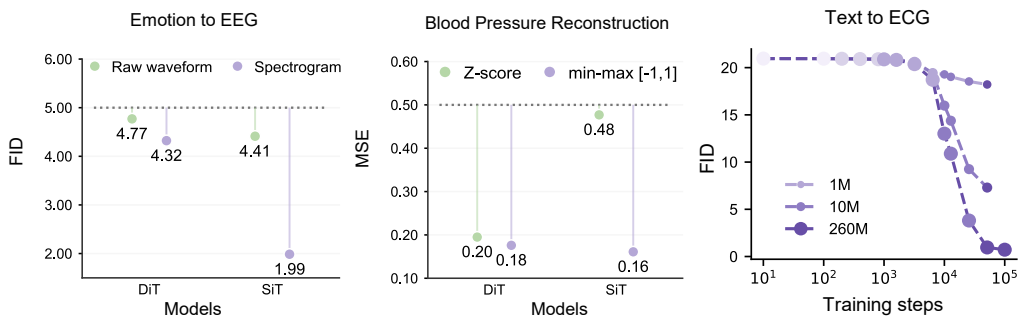

Figure 8: **Left: High-frequency channel modeling approaches.** Spectrogram conditioning improves high-frequency channel generation. **Middle: Signal normalization strategies.** Range-based normalization outperforms z-score normalization for sensor-signal generation. **Right: Scaling behavior.** Increasing training steps and model capacity improves generation quality.

For supervised learning, we train ResNet classifiers on PTB-XL disease diagnosis tasks. We augment the real training set with labeled synthetic ECG signals at three ratios, corresponding to 10%, 50%, and 100% of the real-data size. As shown in Fig. 6, 50% and 10% synthetic augmentation achieve better performance; adding more synthetic data degrades performance, but still often outperforms the real-data-only baseline. This suggests that moderate synthetic augmentation helps address data scarcity, whereas excessive synthetic data may amplify repeated patterns or generator artifacts.

Finding 1: Synthetic sensor signals provide downstream utility through data augmentation.

How do design choices affect generated signal quality? We observe that z-score normalization can introduce undesirable generation bias, such as upward-shifted target signals in the VitalDB proxy-to-target translation task (see Appendix B.1). We therefore compare z-score normalization with range-based $[-1, 1]$ min-max normalization on this setting. As shown in Fig. 8, range-based normalization consistently improves most metrics across models. While z-score normalization is commonly effective for sensor representation learning (Narayanswamy et al., 2024; Shuai et al., 2026), our results suggest that sensor generation may require different normalization choices, with fixed-range scaling providing a more stable target for generative training.

Finding 2: Fixed-range normalization provides a more stable target for sensor generation.

Can sensor generative models scale? We further study whether increasing training computation and model capacity improves sensor-signal generation. We report results on the text-to-ECG generation setting, where we evaluate SiT with different training steps and model sizes. As shown in Fig. 8, generation quality improves consistently with longer training, as reflected by decreasing FID scores across all models. Increasing model size further improves signal fidelity, suggesting that sensor generative models benefit from both compute and capacity scaling.

Finding 3: Sensor generative models scale with training compute and model capacity.

6 Conclusion

We present `SensorGen`, a fully open, large-scale study of generative models for real-world sensor time series. Using 14 generation settings across 4 domains, 7 datasets, 12 signal modalities, and five representative generative model families, `SensorGen` enables controlled analysis of how task formulations, signal properties, model choices, and evaluation protocols shape generation quality. Our results show that flow-matching models serve as strong baselines across diverse settings, signal-aware designs improve generation in challenging regimes, and generated signals are useful beyond visual realism. Together, these findings provide practical lessons for developing and evaluating generative models for sensor time series, and establish `SensorGen` as an open foundation for future research on real-world sensor data generation.

AI USE STATEMENT

We used generative AI tools to assist with drafting and editing parts of this paper, finding related work, and writing and debugging code for data preparation, model implementation, evaluation, and figure and table preparation. We reviewed and revised AI-assisted material and did not treat generated output as scientific evidence. The authors take responsibility for the final study design, analyses, claims, results, and artifacts in this paper.

REFERENCES

Vitalii Bondar, Serhii Semenov, Vira Babenko, and Dmytro Holovniak. Flowecg: Using flow matching to create a more efficient ecg signal generator. *arXiv preprint arXiv:2509.10491*, 2025.

Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, et al. Language models are few-shot learners. *Advances in neural information processing systems*, 33:1877–1901, 2020.

Shing Chan, Yuan Hang, Catherine Tong, Aidan Acquah, Abram Schonfeldt, Jonathan Gershuny, and Aiden Doherty. Capture-24: A large dataset of wrist-worn activity tracker data collected in the wild for human activity recognition. *Scientific Data*, 11(1):1135, 2024.

Zehua Chen, Yuyang Miao, Liyuan Wang, Luyun Fan, Danilo P Mandic, and Jun Zhu. Versatile cardiovascular signal generation with a unified diffusion transformer. *Nature Machine Intelligence*, 8(1):6–19, 2026.

Prafulla Dhariwal and Alexander Nichol. Diffusion models beat gans on image synthesis. *Advances in neural information processing systems*, 34:8780–8794, 2021.

Cheng Ding and Chenwei Wu. Self-supervised learning for biomedical signal processing: A systematic review on ecg and ppg signals. *medRxiv*, pp. 2024–09, 2024.

Patrick Esser, Sumith Kulal, Andreas Blattmann, Rahim Entezari, Jonas Müller, Harry Saini, Yam Levi, Dominik Lorenz, Axel Sauer, Frederic Boesel, et al. Scaling rectified flow transformers for high-resolution image synthesis. In *Forty-first international conference on machine learning*, 2024.

Xiaocheng Fang, Jiarui Jin, Haoyu Wang, Che Liu, Jieyi Cai, Yujie Xiao, Guangkun Nie, Bo Liu, Shun Huang, Hongyan Li, et al. Ppgflowecg: Latent rectified flow with cross-modal encoding for ppg-guided ecg generation and cardiovascular disease detection. *arXiv preprint arXiv:2509.19774*, 2025.

Brian Gow, Tom Pollard, Larry A Nathanson, Alistair Johnson, Benjamin Moody, Chrystinne Fernandes, Nathaniel Greenbaum, Jonathan W Waks, Parastou Eslami, Tanner Carbonati, et al. Mimic-iv-ecg: Diagnostic electrocardiogram matched subset. *Type: dataset*, 2023.

Jonathan Ho, Ajay Jain, and Pieter Abbeel. Denoising diffusion probabilistic models. *Advances in neural information processing systems*, 33:6840–6851, 2020.

Yaojun Hu, Jintai Chen, Lianting Hu, Dantong Li, Jiahuan Yan, Haochao Ying, Huiying Liang, and Jian Wu. Personalized heart disease detection via ecg digital twin generation. *arXiv preprint arXiv:2404.11171*, 2024.

Nabil Ibtehaz, Sakib Mahmud, Muhammad EH Chowdhury, Amith Khandakar, Muhammad Salman Khan, Mohamed Arselene Ayari, Anas M Tahir, and M Sohel Rahman. Ppg2abp: Translating photoplethysmogram (ppg) signals to arterial blood pressure (abp) waveforms. *Bioengineering*, 9(11):692, 2022.

Jong-Hwan Jang, Junho Song, and Yong-Yeon Jo. Cofe: A framework generating counterfactual ecg for explainable cardiac ai-diagnostics. In *ICASSP 2026-2026 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pp. 3471–3475. IEEE, 2026.

Maksims Kazijevs and Manar D Samad. Deep imputation of missing values in time series health data: A review with benchmarking. *Journal of biomedical informatics*, 144:104440, 2023.

Alexander Khazatsky, Karl Pertsch, Suraj Nair, Ashwin Balakrishna, Sudeep Dasari, Siddharth Karamcheti, Soroush Nasiriany, Mohan Kumar Srirama, Lawrence Yunliang Chen, Kirsty Ellis, et al. Droid: A large-scale in-the-wild robot manipulation dataset. *arXiv preprint arXiv:2403.12945*, 2024.

Yongfan Lai, Jiabo Chen, Qinghao Zhao, Deyun Zhang, Yue Wang, Shijia Geng, Hongyan Li, and Shenda Hong. Diffusets: 12-lead ecg generation conditioned on clinical text reports and patient-specific information. *Patterns*, 6(10), 2025.

Hyung-Chul Lee, Yoonsang Park, Soo Bin Yoon, Seong Mi Yang, Dongnyeok Park, and Chul-Woo Jung. Vitaldb, a high-fidelity multi-parameter vital signs database in surgical patients. *Scientific Data*, 9(1):279, 2022.

Hao Li, Bowen Deng, Chang Xu, Zhiyuan Feng, Viktor Schlegel, Yu-Hao Huang, Yizheng Sun, Jingyuan Sun, Kailai Yang, Yiyao Yu, et al. Mira: Medical time series foundation model for real-world health data. *arXiv preprint arXiv:2506.07584*, 2025a.

Sirui Li, Shuhan Xiao, Mihir Joshi, Ahmed Metwally, Daniel McDuff, Wei Wang, and Yuzhe Yang. HEARTS: Benchmarking llm reasoning on health time series. *arXiv preprint arXiv:2603.06638*, 2026.

Tianhong Li, Yonglong Tian, He Li, Mingyang Deng, and Kaiming He. Autoregressive image generation without vector quantization. *Advances in Neural Information Processing Systems*, 37: 56424–56445, 2024.

Tianhong Li, Qinyi Sun, Lijie Fan, and Kaiming He. Fractal generative models. *arXiv preprint arXiv:2502.17437*, 2025b.

Nanye Ma, Mark Goldstein, Michael S Albergo, Nicholas M Boffi, Eric Vanden-Eijnden, and Saining Xie. Sit: Exploring flow and diffusion-based generative models with scalable interpolant transformers. In *European Conference on Computer Vision*, pp. 23–40. Springer, 2024.

Girish Narayanswamy, Xin Liu, Kumar Ayush, Yuzhe Yang, Xuhai Xu, Shun Liao, Jake Garrison, Shyam Tailor, Jake Sunshine, Yun Liu, et al. Scaling wearable foundation models. *arXiv preprint arXiv:2410.13638*, 2024.

Nour Neifar, Achraf Ben-Hamadou, Afef Mdhaffar, and Mohamed Jmaiel. Diffecg: A versatile probabilistic diffusion model for ecg signals synthesis. In *2024 IEEE/ACIS 22nd International Conference on Software Engineering Research, Management and Applications (SERA)*, pp. 182–188. IEEE, 2024.

Tung Nguyen, Johannes Brandstetter, Ashish Kapoor, Jayesh K Gupta, and Aditya Grover. Climax: A foundation model for weather and climate. *arXiv preprint arXiv:2301.10343*, 2023.

Heiko Oppel and Michael Munz. A diffusion model for inertial based time series generation on scarce data availability to improve human activity recognition. *Scientific Reports*, 15(1):16841, 2025.

Sudarshan Pant, Hyung-Jeong Yang, Eunchae Lim, Soo-Hyung Kim, and Seok-Bong Yoo. Phymer: physiological dataset for multimodal emotion recognition with personality as a context. *IEEE Access*, 11:107638–107656, 2023.

William Peebles and Saining Xie. Scalable diffusion models with transformers. In *Proceedings of the IEEE/CVF international conference on computer vision*, pp. 4195–4205, 2023.

Stuart F Quan, Barbara V Howard, Conrad Iber, James P Kiley, F Javier Nieto, George T O'Connor, David M Rapoport, Susan Redline, John Robbins, Jonathan M Samet, et al. The sleep heart health study: design, rationale, and methods. *Sleep*, 20(12):1077–1085, 1997.

Attila Reiss, Ina Indlekofer, Philip Schmidt, and Kristof Van Laerhoven. Deep ppg: Large-scale heart rate estimation with convolutional neural networks. *Sensors*, 19(14):3079, 2019.

Chitwan Saharia, William Chan, Saurabh Saxena, Lala Li, Jay Whang, Emily L Denton, Kamyar Ghasemipour, Raphael Gontijo Lopes, Burcu Karagol Ayan, Tim Salimans, et al. Photorealistic text-to-image diffusion models with deep language understanding. *Advances in neural information processing systems*, 35:36479–36494, 2022.

Debaditya Shome, Pritam Sarkar, and Ali Etemad. Region-disentangled diffusion model for high-fidelity ppg-to-ecg translation. In *Proceedings of the AAAI conference on artificial intelligence*, volume 38, pp. 15009–15019, 2024.

Zitao Shuai, Zongzhe Xu, David Yang, Wei Wang, and Yuzhe Yang. OSF: On pre-training and scaling of sleep foundation models. *arXiv preprint arXiv:2603.00190*, 2026.

Yang Song, Jascha Sohl-Dickstein, Diederik P Kingma, Abhishek Kumar, Stefano Ermon, and Ben Poole. Score-based generative modeling through stochastic differential equations. *arXiv preprint arXiv:2011.13456*, 2020.

Peize Sun, Yi Jiang, Shoufa Chen, Shilong Zhang, Bingyue Peng, Ping Luo, and Zehuan Yuan. Autoregressive model beats diffusion: Llama for scalable image generation. *arXiv preprint arXiv:2406.06525*, 2024.

Julius Vetter, Jakob H Macke, and Richard Gao. Generating realistic neurophysiological time series with denoising diffusion probabilistic models. *Patterns*, 5(9), 2024.

Patrick Wagner, Nils Strodthoff, Ralf-Dieter Bousseljot, Dieter Kreiseler, Fatima I Lunze, Wojciech Samek, and Tobias Schaeffter. Ptb-xl, a large publicly available electrocardiography dataset. *Scientific data*, 7(1):154, 2020.

Fuying Wang, Jiacheng Xu, and Lequan Yu. From token to rhythm: A multi-scale approach for ecg-language pretraining. *arXiv preprint arXiv:2506.21803*, 2025.

Zhou Wang, Alan C Bovik, Hamid R Sheikh, and Eero P Simoncelli. Image quality assessment: from error visibility to structural similarity. *IEEE transactions on image processing*, 13(4):600–612, 2004.

Miriam Kopperstad Wolff, Peter Calhoun, Eleonora Maria Aiello, Yao Qin, and Sam Fox Royston. Metabonet: The largest publicly available consolidated data set for type 1 diabetes management. *Journal of Diabetes Science and Technology*, pp. 19322968261441637, 2026.

Zongzhe Xu, Zitao Shuai, Eideen Mozaffari, Ravi S Aysola, Rajesh Kumar, and Yuzhe Yang. SleepLM: Natural-language intelligence for human sleep. *arXiv preprint arXiv:2602.23605*, 2026.

Yuzhe Yang, Yuan Yuan, Guo Zhang, Hao Wang, Ying-Cong Chen, Yingcheng Liu, Christopher G Tarolli, Daniel Crepeau, Jan Bukartyk, Mithri R Junna, et al. Artificial intelligence-enabled detection and assessment of parkinson's disease using nocturnal breathing signals. *Nature Medicine*, 28(10):2207–2215, 2022.

Yuzhe Yang, Xin Liu, Jiang Wu, Silviu Borac, Dina Katabi, Ming-Zher Poh, and Daniel McDuff. Simper: Simple self-supervised learning of periodic targets. In *International Conference on Learning Representations (ICLR)*, 2023.

Shuangfei Zhai, Ruixiang Zhang, Preetum Nakkiran, David Berthelot, Jiatao Gu, Huangjie Zheng, Tianrong Chen, Miguel Angel Bautista, Navdeep Jaitly, and Josh Susskind. Normalizing flows are capable generative models. *arXiv preprint arXiv:2412.06329*, 2024.

Guo-Qiang Zhang, Licong Cui, Remo Mueller, Shiqiang Tao, Matthew Kim, Michael Rueschman, Sara Mariani, Daniel Mobley, and Susan Redline. The national sleep research resource: towards a sleep data commons. *Journal of the American Medical Informatics Association*, 25(10):1351–1358, 2018.

Yuwei Zhang, Kumar Ayush, Siyuan Qiao, A Ali Heydari, Girish Narayanswamy, Maxwell A Xu, Ahmed A Metwally, Shawn Xu, Jake Garrison, Xuhai Xu, et al. Sensorlm: Learning the language of wearable sensors. *arXiv preprint arXiv:2506.09108*, 2025.

Table 8: **Overview of datasets used in `SensorGen`.**

Dataset	Domain	Modality	Patients
MIMIC-IV ECG	Emergency Department	electrocardiography (ECG)	154,075
PPG-DaLiA	Daily Life	photoplethysmography (PPG)	15
		electrocardiography (ECG)	
		heart rate (HR)	
SHHS	Lab Study	electrocardiography (ECG)	6,663
		electroencephalography (EEG)	
		electromyography (EMG)	
		electrooculography (EOG)	
PhyMER	Lab Study	electroencephalography (EEG)	30
		photoplethysmography (PPG)	
		electrodermal activity (EDA)	
		skin temperature (TEMP)	
CAPTURE-24	Daily Life	inertial measurement unit (IMU)	151
VitalDB	Operation Room	electrocardiology (ECG)	3,073
		photoplethysmography (PPG)	
		arterial blood pressure (ABP)	
		non-invasive blood pressure (NIBP)	
		airway pressure waveform (AWP)	
		end-tidal capnography (Airflow)	
		electroencephalography (EEG)	
		medication volume	
Metabonet	Daily Life	glucose monitoring (CGM)	1,211
		insulin delivery - basal	
		insulin delivery - bolus	
		carbohydrate intake	

Table 9: **Data configurations used in `SensorGen`.** The segmented windows in each generation setting.

Dataset	Frequency	Timespan	Train windows	Test windows
MIMIC-IV ECG	100 Hz	10 s	533.3k	227.3k
PPG-DaLiA	100 Hz	1 min	1.4k	0.7k
SHHS	64 Hz	30 s	5,550.9K	1,590.4K
PhyMER	256 Hz	5 s	12.2k	3.1k
CAPTURE-24	30 Hz	30 s	245.1k	62.1k
VitalDB	50/100 Hz	30 s	826.4k	204.7k
Metabonet	12 per Hour	7 days	360.3k	63.8k

A Data Details

A.1 Unified Data Infrastructure

In this section, we present the unified data preprocessing and preparation pipeline used in `SensorGen`. Rather than being a purely engineering component, this pipeline is a necessary foundation for our controlled study. It standardizes data collection, preprocessing, task construction, and data loading across heterogeneous sensor datasets, enabling consistent evaluation across domains, modalities, and task categories. The pipeline supports signals with diverse sampling frequencies, time spans, sequence lengths, and channel structures, allowing different generative model families to be trained and evaluated under the same framework. It enables the integration of isolated datasets and generation

tasks, and thus enables revealing cross-setting findings in sensor-signal generation. Furthermore, our pipeline is also designed to be extensible, so that future datasets and task formulations can be incorporated with minimal modification. We believe this infrastructure can help support broader research on generative modeling for sensor time series.

Preprocessing. The preprocessed data are stored in a shared HDF5 format. Each signal is saved as a continuous array with shape (C,T), together with metadata such as sampling rate, channel names, subject identifier, recording/group identifier, and optional labels. Task-specific resampling, windowing, and post-processing are performed at loading time, so the same stored files can support different task definitions without regeneration. For datasets with heterogeneous sampling rates or modalities, signals from the same recording are aligned at the group level and stored as sibling entries in the shared HDF5 files. In the main experiments, each requested window loaded from the HDF5 files is normalized independently using per-channel min–max scaling to $[-1, 1]$.

Data organization and loading. We use a unified loading interface to construct task-specific examples from the shared HDF5 files. Instead of materializing all windows offline, the loader builds lightweight indices and reads only the requested windows during training to reduce I/O overhead. Specific periods of signals are sliced and loaded by time-stamp, resampled on the fly, concatenated across selected channels, and normalized when required. All tasks are represented by a common batch format, `UnifiedBatch=`$(x, c_1, c_2, \mathrm{meta})$, where x is the target signal, c_1 contains sparse or global conditions such as text, labels, static covariates, or auxiliary channels, and c_2 contains dense temporal conditions such as past context or cross-channel signals. This abstraction covers all task categories in `SensorGen`, with light-weight model-specific adapters converting `UnifiedBatch` into the input format required by each generative model family.

A.2 Data Usage Details

Here we provide detailed statistics for the datasets used in our paper. Table 8 summarizes the number of subjects, domains, and modalities covered by each dataset, while Table 9 reports the number of training and testing windows, sampling frequency, and time span of the preprocessed signals. As shown in the tables, our datasets span diverse real-world sensing domains and cover a broad range of modalities, resulting in signals with heterogeneous sampling frequencies and time spans. These diversities support our in-depth analysis upon generative modeling on sensor signals.

B TASK DETAILS

B.1 TASK FORMULATION DETAILS

Notation. We formulate all benchmark settings as conditional sensor-signal generation. Let $\mathbf{x} \in \mathbb{R}^{C \times T}$ denote the target signal, where C is the number of channels and T is the sequence length. We use $c_1 \in \mathbb{R}^D$ to denote the tokenized semantic condition, such as a text report, class label, or other non-signal metadata, and $c_2 \in \mathbb{R}^{N \times D}$ to denote the tokenized signal-side condition derived from observed signals, such as historical context, source channels, temporal boundaries, low-resolution observations, noisy observations, or an observed signal to be edited. Here, D is the embedding dimension and N is the number of signal tokens.

Under this notation, each task defines a conditional distribution over target signals:

$$\widehat{\mathbf{x}} \sim p_\theta\left(\mathbf{x} \mid \mathcal{C}\right), \qquad \mathcal{C} \subseteq \{c_1, c_2\},$$

where $\mathcal{C}$ denotes the set of available conditions for a specific task category. For example, semantic-to-signal tasks use only semantic conditions, while translation and temporal reconstruction tasks primarily use signal-side conditions.

Semantic-to-Signal Generation. Semantic-to-signal generation synthesizes target signals from semantic conditions alone. Under our notation, the available condition set is $\mathcal{C} = \{c_1\}$, and the model generates

$$\widehat{\mathbf{x}} \sim p_\theta\left(\mathbf{x} \mid c_1\right).$$

The generated signal should be both realistic as a sensor signal and consistent with the semantic content specified by c_1. We therefore assess the realism and diversity of generated samples using distribution-level metrics, including FID, precision, and recall. This category includes two forms, depending on whether c_1 is represented by free-form text or a discrete label.

Interpolation & Extrapolation. Interpolation and extrapolation generate unobserved signal segments from observed temporal context. In all settings under this category, the model is given a signal-side condition c_2 that provides partial temporal observations, such as historical context or surrounding boundary segments. Some settings additionally include a compressed token as condition c_1, such as intervention variables or medication trajectories. The available condition set is therefore $\mathcal{C} = \{c_2\}$ or $\mathcal{C} = \{c_1, c_2\}$, and the model generates

$$\widehat{\mathbf{x}} \sim p_\theta\left(\mathbf{x} \mid c_2\right) \quad \text{or} \quad \widehat{\mathbf{x}} \sim p_\theta\left(\mathbf{x} \mid c_1, c_2\right).$$

Here, forecasting-based tasks use past observations as c_2 and generate future segments, while temporal imputation uses surrounding observations as c_2 and generates the missing middle segment. We use sample-level evaluation metrics to measure the quality of generated signals.

Translation. Signal translation generates target modalities from observed signals. In this category, the model is given a signal-side condition c_2 corresponding to available channels or proxy measurements. Some tasks additionally include a compressed token as condition c_1, such as demographic covariates. The available condition set is therefore $\mathcal{C} = \{c_2\}$ or $\mathcal{C} = \{c_1, c_2\}$, and the model generates

$$\widehat{\mathbf{x}} \sim p_\theta\left(\mathbf{x} \mid c_2\right) \quad \text{or} \quad \widehat{\mathbf{x}} \sim p_\theta\left(\mathbf{x} \mid c_1, c_2\right).$$

Here, c_2 and $\mathbf{x}$ correspond to different channels or modalities at the same timestamps. These tasks require models to recover target signals that are consistent with the observed source measurements. Therefore, we use sample-level metrics to evaluate reconstruction quality.

Editing. Given an observed signal c_2, the model generates a desired target signal x while preserving non-targeted semantics whenever possible. The observed signal-side input may correspond to a low-resolution signal, a noisy signal, a partially observed signal, or a source signal to be edited. Some settings have additional signal-side conditions, such as correlated signals, which will also be treated as c_2. Semantic conditions c_1, including text instructions, target semantic conditions, or auxiliary covariates, are also provided in some tasks. Thus, $\mathcal{C} = \{c_2\}$ or $\mathcal{C} = \{c_1, c_2\}$, and

$$\widehat{\mathbf{x}} \sim p_\theta\left(\mathbf{x} \mid c_2\right) \quad \text{or} \quad \widehat{\mathbf{x}} \sim p_\theta\left(\mathbf{x} \mid c_1, c_2\right).$$

Note that observed signals can play different roles across tasks. For example, in glucose signal super-resolution, the low-resolution signal serves as a signal-side condition, whereas in ECG denoising, the noisy input signal is directly provided as the target to be restored by the generation model. The generated signal should follow the desired editing or restoration target, while preserving signal attributes that are not intended to change. Thus, we use sample-level metrics to assess generation fidelity.

B.2 Task Setting Details

Here we provide details of our task settings organized by the datasets used in our study.

B.2.1 MIMIC-IV ECG

Semantic-to-Signal – Text-to-Signal Generation

- Text-to-ECG Generation
  Inputs: Free-text ECG report | Output: 12-lead ECG signal
  Sequence Length: 1,000 | Time Span: 10 sec | Frequency: 100 Hz
  Metric: FID, Precision, Recall

Editing – Semantic Signal Editing

- ECG Editing
  Inputs: Source 12-lead ECG signal x_a | Source report text$_a$ | Target report text$_b$
  Output: Edited 12-lead ECG signal $\widehat{x}_b$ aligned with the target report
  Sequence Length: 1,000 | Time Span: 10 sec | Frequency: 100 Hz
  Metric: MSE, MAE, SMSE

Editing – Denoising

- ECG Denoising
  Inputs: Noisy 12-lead ECG ($x_{\text{noisy}} = x_{\text{clean}} + \sigma \cdot \varepsilon, \varepsilon \sim \mathcal{N}(0, I)$) | Free-text ECG report
  Output: Denoised 12-lead ECG signal
  Sequence Length: 1,000 | Time Span: 10 sec | Frequency: 100 Hz
  Metric: MSE, MAE, SMSE

B.2.2 PPG-DaLiA

Interpolation & Extrapolation – Forecasting

- Cardiac Signal Forecasting
  Inputs: historical HR, ECG, and BVP signals | Output: future HR, ECG, and BVP signals
  Sequence Length: 6,000 (4,000 context points; 2,000 target points) | Time Span: 60 sec (40 sec context; 20 sec target) | Frequency: 100 Hz
  Metric: MSE, MAE, SMSE

Translation – Channel Translation

- PPG-to-ECG Translation
  Inputs: PPG signal | Output: ECG signal
  Sequence Length: 6,000 | Time Span: 60 sec | Frequency: 100 Hz
  Metric: MSE, MAE, SMSE

B.2.3 SHHS

Interpolation & Extrapolation – Temporal Imputation

- Sleep Signal Imputation
  Inputs: left 10 sec and right 10 sec context from 2 EEG, 2 EOG, ECG, EMG | Output: masked middle 10 sec of the same channels
  Sequence Length: 1,920 (1,280 context points; 640 target points) | Time Span: 30 sec (20 sec

context; 10 sec target) | Frequency: 64 Hz
Metric: MSE, MAE, SMSE

B.2.4 PHYMER

Translation – Channel Translation

• Peripheral-to-EEG Translation
Inputs: peripheral temperature, EDA, and BVP signals | Output: 14-channel EEG signal
Sequence Length: 1,280 | Time Span: 5 sec | Frequency: 256 Hz
Metric: MSE, MAE, SMSE

Semantic-to-Signal – Label-to-Signal Generation

• Emotion-to-EEG Generation
Inputs: emotion label | Output: 14-channel EEG signal
Sequence Length: 1,280 | Time Span: 5 sec | Frequency: 256 Hz
Metric: FID, Precision, Recall

B.2.5 CAPTURE-24

Semantic-to-Signal – Label-to-Signal Generation

• Activity-to-IMU Generation
Inputs: activity labels | Output: 3-axis accelerometer signal
Sequence Length: 900 | Time Span: 30 sec | Frequency: 30 Hz
Metric: FID, Precision, Recall

Editing – Semantic Signal Editing

• IMU Editing
Inputs: Source 3-axis accelerometer signal x_a | Source label y_a | Target report y_b
Output: Edited 3-axis accelerometer signal $\widehat{x}_b$ aligned with the target report
Sequence Length: 900 | Time Span: 30 sec | Frequency: 30 Hz
Metric: MSE, MAE, SMSE

B.2.6 VITALDB

Interpolation & Extrapolation – Intervention-Conditioned Forecasting

• Medication-Aware Forecasting
Inputs: historical ECG, PPG, breathing, EEG, blood pressure, and medications | Output: future ECG, PPG, breathing, EEG, and blood pressure
Sequence Length: 3,000 (2,000 context points; 1,000 target points) | Time Span: 30 sec (20 sec context; 10 sec target) | Frequency: 100 Hz
Metric: MSE, MAE, SMSE

Translation – Proxy-to-Target Reconstruction

• Invasive Blood Pressure Reconstruction
Inputs: PPG and non-invasive blood pressure | Output: arterial blood pressure
Sequence Length: 1,500 | Time Span: 30 sec | Frequency: 50 Hz
Metric: MSE, MAE, SMSE

B.2.7 METABONET

Translation – Channel Translation

• Glucose Signal Translation
Inputs: insulin, carbohydrate histories, and demographic covariates | Output: continuous glucose monitoring (CGM) signal
Sequence Length: 2,016 | Time Span: 7 days | Frequency: 1 sample / 5 min
Metric: MSE, MAE, SMSE

Editing – Super-Resolution

- Glucose Signal Super-Resolution
  Inputs: low-resolution 1-hour CGM signal, insulin, carbohydrate histories, and demographic covariates | Output: high-resolution 5-min CGM signal
  Sequence Length: 2,016 | Time Span: 7 days | Frequency: 1 sample / 5 min
  Metric: MSE, MAE, SMSE

Overall, the benchmark covers 14 task settings across four generation families: Translation, Semantic-to-Signal, Interpolation & Extrapolation, and Editing.

Table 10: **Performance comparison on Translation.** WaveGAN performs better on PPG-to-ECG translation while FractalGen is strong on Peripheral-to-EEG translation.

Model	PPG-to-ECG Translation			Peripheral-to-EEG Translation		
	MSE↓	MAE↓	SMSE↓	MSE↓	MAE↓	SMSE↓
DiT	0.257	0.370	0.043	0.348	0.475	0.053
SiT	0.244	0.363	0.041	0.352	0.479	0.053
FractalGen	0.381	0.472	0.173	**0.328**	**0.462**	0.065
TarFlow	0.281	0.398	0.052	0.354	0.481	**0.048**
MAR	0.267	0.378	0.049	0.411	0.520	0.053
WaveGAN	**0.155**	**0.305**	**0.037**	–	–	–

Table 11: **Forecasting and intervention-conditioned forecasting results.** SiT performs best on medication-aware forecasting and remains competitive with MAR on PPG-DaLIA forecasting.

Model	Medication-Aware Forecasting			Cardiac Signal Forecasting		
	MSE↓	MAE↓	SMSE↓	MSE↓	MAE↓	SMSE↓
DiT	0.494	0.582	0.163	0.224	0.372	0.046
SiT	**0.282**	**0.426**	0.076	0.203	0.345	0.039
FractalGen	0.532	0.509	**0.068**	0.206	0.344	0.049
TarFlow	0.547	0.556	0.098	0.223	0.370	0.052
MAR	0.570	0.556	0.169	0.198	0.340	0.043
WaveGAN	–	–	–	**0.175**	**0.298**	**0.032**

C Detailed Results

C.1 Detailed Results for Main Experiments

In this section, we present the detailed setting-level results of the generation fidelity of representative generative model families across all task categories. We additionally include WaveGAN as an extra baseline for reference on six representative settings, following the same evaluation protocol. Overall, SiT provides the strongest overall baseline across the task suite, though no single family dominates every setting. Specifically, in Table 15, DiT and SiT achieve the strongest distribution-level fidelity in Semantic-to-Signal settings, while SiT performs better in Interpolation & Extrapolation settings as shown in Table 11 and Table 12. In Translation settings, Table 10 and Table 14 suggest that SiT remains competitive and performs best on PPG-to-ECG, invasive blood pressure reconstruction, and glucose translation. And as marked in Table 16 and Table 13, SiT performs best on 2 of 4 Editing tasks. Beyond SiT's consistent advantage, the relative rankings of compared generative models vary across task categories. This suggests that modern generative paradigms do not naturally transfer uniformly to sensor-signal generation, and their inductive biases may align differently with sensor-specific requirements such as temporal dynamics, spectral structure, and conditional consistency. These findings have motivated us to analyze how these signal properties shape model preferences.

C.2 Additional Quantitative Results

In this section, we show the detailed results of our analysis studies.

C.2.1 Synthetic Augmentation Enhances Downstream Sensor Signal Representations

Building on the experimental settings detailed in the main text, we extend our evaluation to assess the practical downstream utility of synthetic sensor signals. Specifically, we investigate whether generated signals can effectively augment both self-supervised representation learning and supervised disease prediction. The motivation for this analysis is to determine whether generative models can transition from merely producing samples that look realistic to serving actual downstream clinical values.

Table 12: **Performance comparison on Interpolation & Extrapolation.** SiT performs better on Sleep-Signal Imputation.

Model	Sleep Signal Imputation		
	MSE↓	MAE↓	SMSE↓
DiT	0.286	0.389	0.079
SiT	**0.270**	**0.381**	**0.077**
FractalGen	0.296	0.385	0.092
TarFlow	0.290	0.424	0.100
MAR	0.286	0.398	0.091

Table 13: **Performance comparison on Editing.** Imagen performs better on CGM super-resolution.

Model	Glucose Signal Super-Resolution		
	MSE↓	MAE↓	SMSE↓
DiT	0.059	0.151	0.032
SiT	0.134	0.260	0.088
FractalGen	0.127	0.256	0.042
TarFlow	0.034	0.144	0.008
MAR	0.266	0.393	0.121
Imagen	**0.008**	**0.064**	**0.002**

Table 14: **Performance comparison on Translation.** SiT performs better on all these settings.

Model	Glucose Signal Translation			Invasive Blood Pressure Reconstruction		
	MSE↓	MAE↓	SMSE↓	MSE↓	MAE↓	SMSE↓
DiT	0.097	0.223	0.032	0.176	0.283	0.045
SiT	**0.091**	**0.216**	**0.030**	**0.161**	**0.274**	**0.038**
FractalGen	0.149	0.284	0.044	0.571	0.598	0.114
TarFlow	0.307	0.438	0.063	0.584	0.611	0.226
MAR	0.298	0.432	0.063	0.571	0.601	0.218
WaveGAN	0.272	0.412	0.064	0.774	0.666	0.120

Table 15: **Performance comparison on Semantic-to-Signal.** SiT achieves overall better performance on activity-to-IMU and emotion-to-EEG generation, and Imagen achieves better performance on text-to-ECG task setting.

Model	Text-to-ECG			Activity-to-IMU			Emotion-to-EEG		
	FID↓	Precision↑	Recall↑	FID↓	Precision↑	Recall↑	FID↓	Precision↑	Recall↑
DiT	1.07	0.982	0.373	0.32	0.890	0.618	4.77	0.788	0.098
SiT	2.03	**0.992**	0.325	**0.25**	0.893	**0.683**	**4.41**	**0.861**	**0.232**
FractalGen	11.89	0.963	0.059	1.10	0.884	0.151	8.83	0.049	0.008
TarFlow	4.66	0.971	0.107	0.79	0.857	0.225	7.34	0.024	0.000
MAR	14.48	0.702	0.001	0.47	**0.903**	0.463	8.33	0.637	0.011
Imagen	**1.07**	0.976	**0.478**	0.54	0.830	0.550	5.93	0.395	0.010
WaveGAN	3.94	0.922	0.084	0.58	0.829	0.420	–	–	–

Table 16: **Performance comparison on training-free Editing.** SiT achieves the best performance on ECG editing and ECG denoising, and DiT is the strongest on IMU editing task.

Model	Sample Level		
	MSE↓	MAE↓	SMSE↓
ECG Editing			
DiT	0.138	0.253	0.045
SiT	**0.129**	**0.236**	**0.043**
ECG Denoising			
DiT	0.034	0.099	0.012
SiT	**0.006**	**0.051**	**0.002**
IMU Editing			
DiT	**0.308**	**0.447**	**0.074**
SiT	0.329	0.463	0.085

For self-supervised learning, we evaluated downstream finetuning on PTB-XL using the encoders pre-trained on a mixture of real and synthetic data, as well as a baseline model pretrained only on the real data. We provided the detailed raw results in Table 18 as a supplementary to our main paper.

Table 17: **Downstream utility of generated signals.** With ECGs synthesized by a text-to-ECG generator, we evaluate their utility in supervised learning, and find moderate synthetic augmentation improves disease classification performance, whereas excessive synthetic data degrades it.

Training Data	Ratio	MI		STTC		CD	
		AUC↑	AUPRC↑	AUC↑	AUPRC↑	AUC↑	AUPRC↑
REAL	-	90.86	70.62	92.17	73.37	90.14	74.64
REAL + SYN	10%	91.31	**74.25**	**92.95**	**74.33**	91.56	76.45
	50%	**91.51**	73.11	92.83	72.29	**92.14**	77.14
	100%	91.05	70.28	91.92	67.79	91.66	**77.79**

Table 18: **Downstream utility of generated signals.** With ECGs synthesized by a text-to-ECG generator, we evaluate their utility in self-supervised learning. We found that adding synthetic data improves zero-shot transferability in ECG-text representation learning.

Setting	+ Syn. Data	MI		STTC		CD	
		AUC↑	AUPRC↑	AUC↑	AUPRC↑	AUC↑	AUPRC↑
Zero-shot	✗	67.90	46.03	60.45	33.10	68.18	**48.97**
	✓	**74.59**	**56.56**	**73.97**	**50.89**	**69.40**	43.24
Finetune	✗	84.47	**68.65**	90.64	75.96	86.13	**76.41**
	✓	**84.56**	68.05	**90.86**	**76.44**	**86.81**	76.37

Table 19: **Downstream utility of generated signals.** We further evaluate synthetic augmentation across age groups on the held-out PTB-XL test set. Bold indicates the higher AUROC within each disease–age pair.

Age group	Proportion (%)	MI		STTC		CD	
		Real	Real + Syn	Real	Real + Syn	Real	Real + Syn
< 40	15.8	**89.1**	86.1	83.9	**91.5**	82.8	**87.7**
40–59	33.4	91.9	**92.3**	**92.2**	91.9	88.1	**90.3**
60–79	40.5	89.1	**89.3**	92.8	**92.9**	90.6	**91.3**
≥ 80	10.2	85.3	**87.5**	89.6	**90.2**	94.0	**95.1**

We observe that a 50% synthetic augmentation yields the optimal performance boost, effectively mitigating data scarcity as intended. Interestingly, aggressively scaling the synthetic augmentation to 100% degrades performance relative to this peak, though it generally remains superior to using no synthetic data at all. The existence of an optimal mixing ratio is consistent with the intuition that while synthetic augmentation provides an immediate performance uplift, over-saturating the training pool likely forces the model to overfit to subtle generator artifacts or redundant, collapsed patterns. Consequently, this implies that future applications of generative biosignal models may benefit more from careful filtering, where only high-confidence or highly diverse synthetic samples are retained, rather than naive, brute-force dataset scaling.

Additionally, for the encoders pretrained with a mixture of real and synthetic data, and the baseline model pretrained with the real data, we test the zero-shot classification performance and report in Table 18. By adding synthetic data during SSL, as demonstrated in Table 18, there are notable gains in the three zero-shot tasks, which again confirms the quality of the diverse and realistic signals generated by our model.

We further evaluate the real-only and real-plus synthetic classifiers across four age groups in the held-out PTB-XL test set. As shown in Table 19, synthetic augmentation improves AUROC in 10 of 12 disease-by-age comparisons, with gains observed in every age group. In particular, all three diseases benefit in the subgroup aged 80 years and older.

Table 20: **Comparison on normalization strategies of input signals.** We report results on invasive blood pressure reconstruction. Range based normalization provides the best performance.

Model	Normalization	Sample Level		
		MSE$^{\downarrow}$	MAE$^{\downarrow}$	SMSE$^{\downarrow}$
DiT	z-score	0.195	0.311	**0.043**
	min-max $[-1, 1]$	**0.176**	**0.283**	0.045
SiT	z-score	0.477	0.545	0.112
	min-max $[-1, 1]$	**0.161**	**0.274**	**0.038**

C.2.2 Min-Max Normalization Mitigates Generative Vulnerability to Sensor Artifacts

Sensor signals are inherently prone to noise originating from environmental factors, shifting subject conditions, and variable hardware quality. For example, the physical tightness of a respiratory band introduces substantial variance for identical breathing patterns, sudden motion during recordings causes extreme electrical spikes, and poor electrode contact frequently induces baseline wander.

While generative models excel at step-wise denoising (e.g., diffusion and flow-matching architectures), they are uniquely vulnerable to these extreme physiological artifacts. Because these models rely on mapping data to a bounded prior distribution, unbounded extreme values can destabilize the training objective. We hypothesized that bounding the signal amplitude using min-max normalization would provide a more stable learning target for these models compared to standard z-score normalization, which preserves highly extreme values and exposes the network to disproportionate gradient updates.

Following the analysis in the main paper, we provide the raw results here in Table 20. The findings confirm our hypothesis: min-max normalization consistently matches or outperforms z-score normalization across all evaluated generative metrics.

These results carry an important practical implication for generative scaling in the sensor domain. They suggest that strictly bounding signal amplitudes is a more critical preprocessing step than preserving the global statistical variance of the raw dataset. Because z-score normalization preserves destructive, high-amplitude artifacts, bounded normalization strategies like min-max should be considered the default approach to ensure training stability and higher-fidelity generation in noise-prone physiological datasets.

C.2.3 Generative models maintain long-sequence stability when signal-side context is available, but struggle in purely semantic-to-signal generation.

We next examine whether introducing signal-side context can help mitigate quality degradation in long-sequence generation, since maintaining temporal consistency over extended horizons remains a primary challenge for many generative models. To this end, we evaluate paired short- and long-sequence settings across three distinct generation paradigms: semantic-to-signal generation, channel translation, and signal imputation. For comparable evaluation, we split each generated long signal into slices matching the short-sequence duration, compute metrics on each slice, and report the average.

Supplementary to the main paper, we provide the raw results in Table 21 & 22. In case of CAPTURE-24, generation quality severely degrades as sequence length increases, where the model must synthesize long-range accelerometry entirely from abstract activity type. Conversely, performance remains significantly more stable in imputation and translation settings. This contrast suggests that long-sequence generation is highly tractable when target-signal context is available, suggesting that the surrounding or parallel observed segments act as temporal anchors, effectively reducing the generative burden compared to unconstrained semantic-to-signal generation.

Table 21: **Comparison of long-sequence generation.** Models perform better on imputation and translation tasks when signal-side context is available.

Model	Context Length	Sleep Signal Imputation			PPG-to-ECG		
		MSE↓	MAE↓	RMSE↓	MSE↓	MAE↓	RMSE↓
DiT	Short	0.286	0.389	0.512	0.228	0.363	0.478
	Long	**0.174**	**0.260**	**0.358**	0.209	0.359	0.439
SiT	Short	0.270	0.381	0.502	0.216	0.352	0.448
	Long	0.196	0.278	0.381	**0.157**	**0.301**	**0.382**

Table 22: **Comparison of long-sequence generation.** Models struggle with long signals in semantic-to-signal generation.

Model	Context	Distribution Metric		
		FID↓	Precision↑	Recall↑
DiT	Activity-to-IMU 30s	0.32	0.890	0.618
	Activity-to-IMU 300s	0.93	0.813	**0.657**
SiT	Activity-to-IMU 30s	**0.25**	**0.893**	**0.683**
	Activity-to-IMU 300s	**0.41**	**0.856**	0.641

Table 23: **Effect of noise level on ECG denoising.** We compare DiT and SiT under two noise levels. Bold indicates the best performance at each noise level.

Noise Level (σ)	Model	Sample-level Metric		
		MSE↓	MAE↓	SMSE↓
0.1	DiT	0.0340	0.0990	0.0120
	SiT	**0.0060**	**0.0510**	**0.0020**
0.5	DiT	0.0815	0.1764	0.0309
	SiT	**0.0203**	**0.1014**	**0.0061**

Table 24: **Effect of conditioning channels on glucose signal translation.** We ablate the carbohydrate-intake channel from the input conditions. Bold indicates the better performance within each model.

Model	Carbohydrate-Intake Channel	Sample-level Metric		
		MSE↓	MAE↓	SMSE↓
DiT	✓	**0.1157**	**0.2431**	**0.0349**
	✗	0.1235	0.2529	0.0376
SiT	✓	**0.0986**	**0.2259**	**0.0316**
	✗	0.1106	0.2449	0.0347

C.2.4 Effect of Noise Level on ECG Denoising

We further study how noise severity affects generation performance by varying the noise level in ECG denoising. Noisy inputs are constructed as $x_{\text{noisy}} = x_{\text{clean}} + \sigma \cdot \epsilon$, where $\epsilon \sim \mathcal{N}(0, I)$, and we evaluate two noise levels, $\sigma \in \{0.1, 0.5\}$. As shown in Table 23, SiT consistently outperforms DiT across both noise levels and all metrics. The relative ranking remains unchanged as noise severity increases, showing that the model-family comparison is robust to noise level in ECG denoising.

Table 25: **Confidence intervals on glucose signal translation.** We report point estimates with 95% confidence intervals for representative models. Bold indicates the best point estimate for each metric.

Model	Glucose Signal Translation		
	MSE↓	MAE↓	SMSE↓
DiT	0.102 [0.097, 0.107]	0.230 [0.224, 0.236]	0.033 [0.032, 0.034]
SiT	**0.092** [0.088, 0.097]	**0.218** [0.212, 0.224]	**0.030** [0.029, 0.031]
TarFlow	0.306 [0.300, 0.312]	0.438 [0.433, 0.442]	0.064 [0.063, 0.065]

Table 26: **Comparisons of different levels of metrics.** Feature space metrics are more related to signal fidelity. Here we report the cardiac-signal forecasting results.

Model	Raw Waveform Level		Feature Space Level		Signal Fidelity
	MSE↓	MAE↓	MSE↓	MAE↓	FID↓
DiT	0.224	0.372	0.0085	**0.0637**	**4.14**
SiT	0.203	0.345	**0.0080**	0.0661	5.00
FractalGen	0.206	0.344	0.0216	0.1010	15.96
TarFlow	0.223	0.370	0.0111	0.0732	7.16
MAR	0.198	0.340	0.0093	0.0681	6.05
MEAN	**0.193**	**0.299**	0.0242	0.1032	18.90

C.2.5 EFFECT OF CONDITIONING CHANNELS ON CGM TRANSLATION

We further examine how cross-channel dependency affects signal translation by removing the carbohydrate-intake channel from the glucose signal translation input. As shown in Table 24, retaining the carbohydrate-intake channel consistently improves performance across all metrics for both DiT and SiT. These results show that stronger cross-channel dependency facilitates signal translation.

C.2.6 CONFIDENCE INTERVALS FOR CGM TRANSLATION

We further evaluate the statistical reliability of the model comparison by reporting confidence intervals for representative models on the glucose signal translation task. Specifically, we report the point estimate and 95% confidence interval for each evaluation metric. As shown in Table 25, SiT achieves the best point estimate across all three metrics, followed by DiT and TarFlow. The reported confidence intervals further support that the observed inter-model differences are stable across the evaluated metrics.

C.3 FAILURE CASE ANALYSIS

Modeling high-frequency signal is hard. We further examine scenarios where generative models fail to capture diverse signal semantics. As shown in Table 15, all model families struggle on the emotion-to-EEG generation task. Although their precision scores remain relatively high, their recall scores are substantially lower. This suggests that current generative models can synthesize realistic-looking EEG signals, but tend to cover only a limited subset of semantic and spectral patterns, failing to capture the full spectrum of EEG semantics. This motivates us to study which design choices can improve generation quality in such challenging high-frequency signal settings.

Raw-waveform metrics can favor low-information predictions that are numerically close but dynamically unrealistic. To examine whether common reconstruction metrics fully reflect generation

Table 27: **Relationship between generation metrics and ECG morphology.** Spearman's ρ between each generation metric and ECG morphological distance in text-to-ECG generation.

Feature Space	Spearman's ρ		
	FID	Precision	Recall
MIRA	+0.91	−0.55	−0.62
Raw Signal	+0.82	−0.68	+0.19

quality, we compare learned generators with deterministic historical-mean baselines on cardiac-signal forecasting, as an instantiation of the Interpolation & Extrapolation tasks categories. The baseline summarizes the observed context by its channel-wise mean and repeats the resulting value across the target horizon. As shown in Table 26, although such predictions contain little temporal information, they can achieve competitive MSE and MAE by matching the average signal level. In contrast, learned generative models produce signals with more realistic temporal dynamics and better distributional fidelity, even when their raw-waveform errors are not always lower. This discrepancy shows that low time-domain error does not necessarily imply realistic sensor-signal generation.

Feature-space metrics better align with signal realism. We further evaluate generated signals in a learned representation space by extracting embeddings for both generated and ground-truth signals and computing MSE and MAE between them. As shown in Table 26, compared with raw-waveform errors, the trends of feature-space errors are more consistent with FID scores, suggesting that representation-level evaluation may better capture morphology and realistic signal dynamics. Together, these results show that sensor-generation quality cannot be assessed by a single metric family.

Finding 4: Point-wise time-domain metrics alone are insufficient; robust signal-generation evaluation requires the combination of frequency, distribution, and representation level metrics.

C.4 MIRA Metrics and ECG Morphology

We further examine whether MIRA-based generation metrics track clinically meaningful ECG morphology. Using text-to-ECG generation as a representative setting, we compute Spearman's ρ between FID, Precision, and Recall and the distributional distance in standard clinical ECG morphology features. As a reference, we perform the same analysis in the raw signal space. As shown in Table 27, all MIRA-based metrics follow the expected correlation direction. FID is positively correlated with morphological distance, while Precision and Recall are negatively correlated. In contrast, raw-signal Recall is positively correlated with morphological distance. These results support using MIRA features to evaluate ECG generation.

D Detailed Qualitative Results

In this section, we show additional qualitative examples of samples generated by our trained models. Here we show a few examples of the generated signals of representative generation settings. In general, we found ECG, blood pressure, glucose signals, and breathing waveforms are easier to model, while high-frequency channels like EOG, EMG, and EEG channels are harder to model directly.

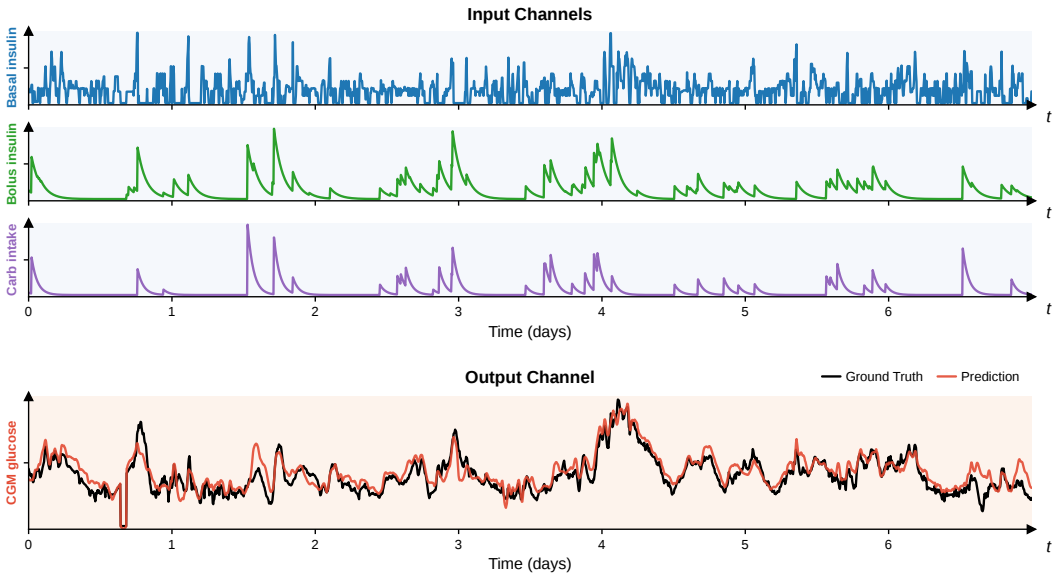

Figure 9: **Visualization of generated signals.** Example from the glucose signal translation task. **Top:** Input signals. **Bottom:** Ground truth and the generated target signal. More examples are in Appendix D.

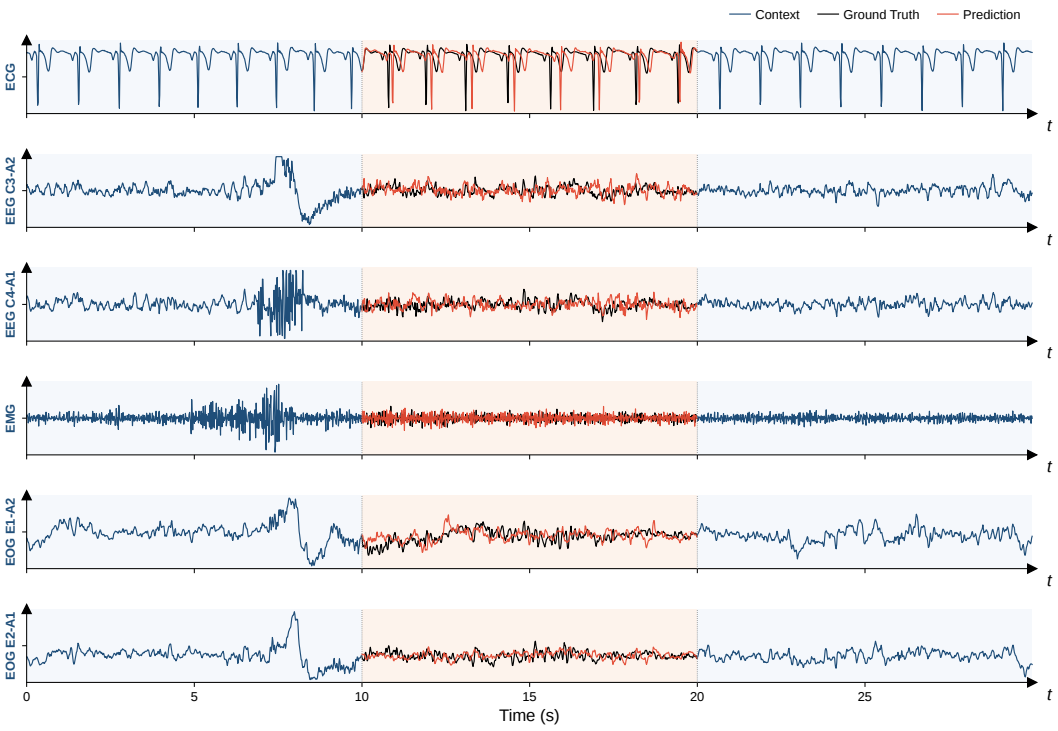

Figure 10: **Visualization of sleep signal imputation task.**

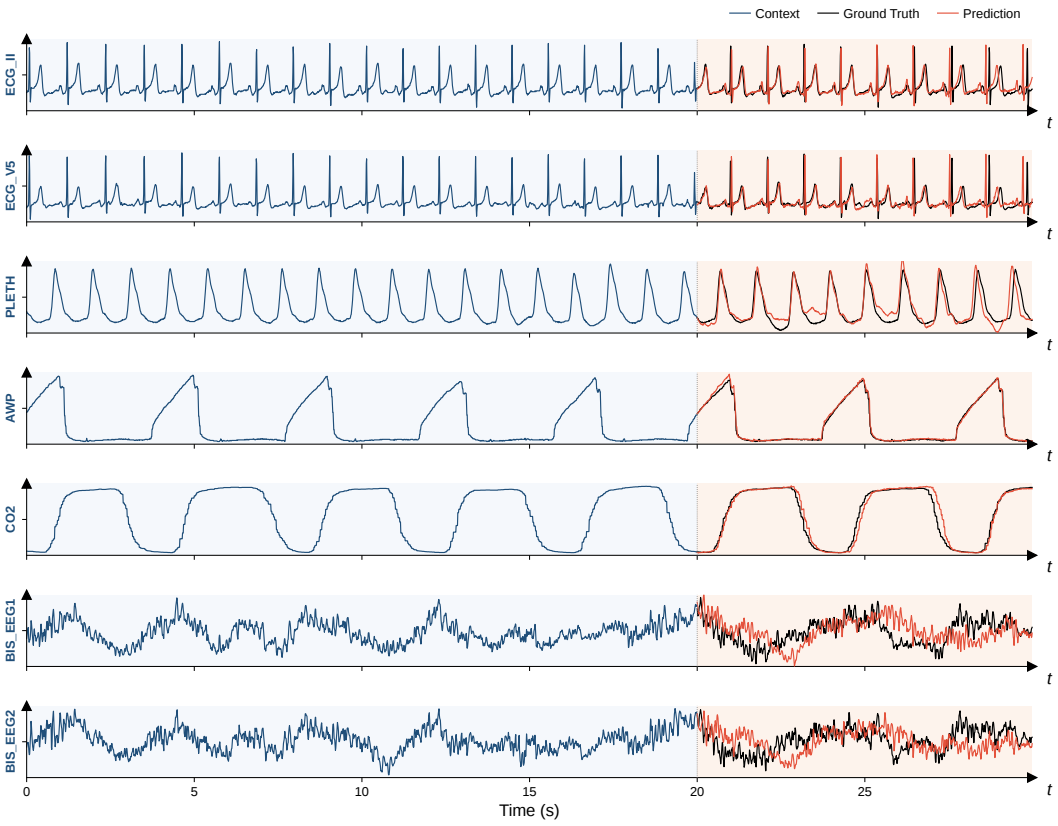

Figure 11: **Visualization of medication-aware forecasting task.**

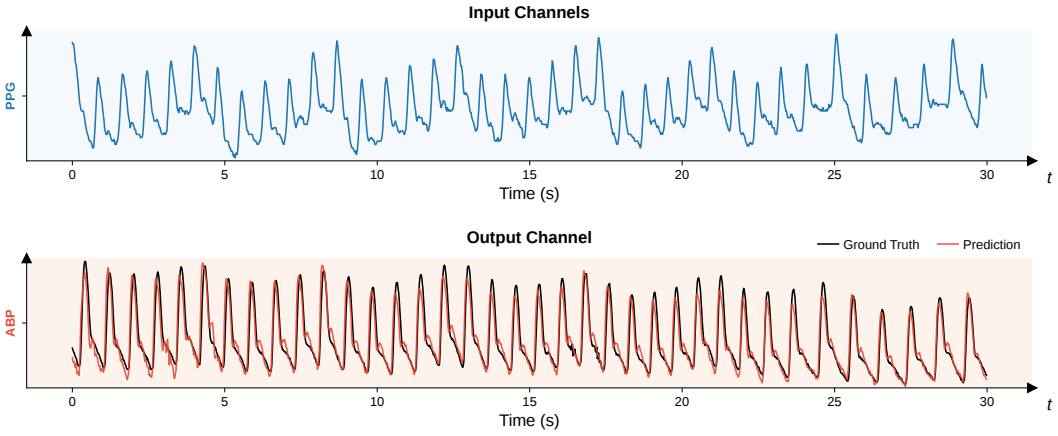

Figure 12: **Visualization of invasive blood pressure reconstruction task.**

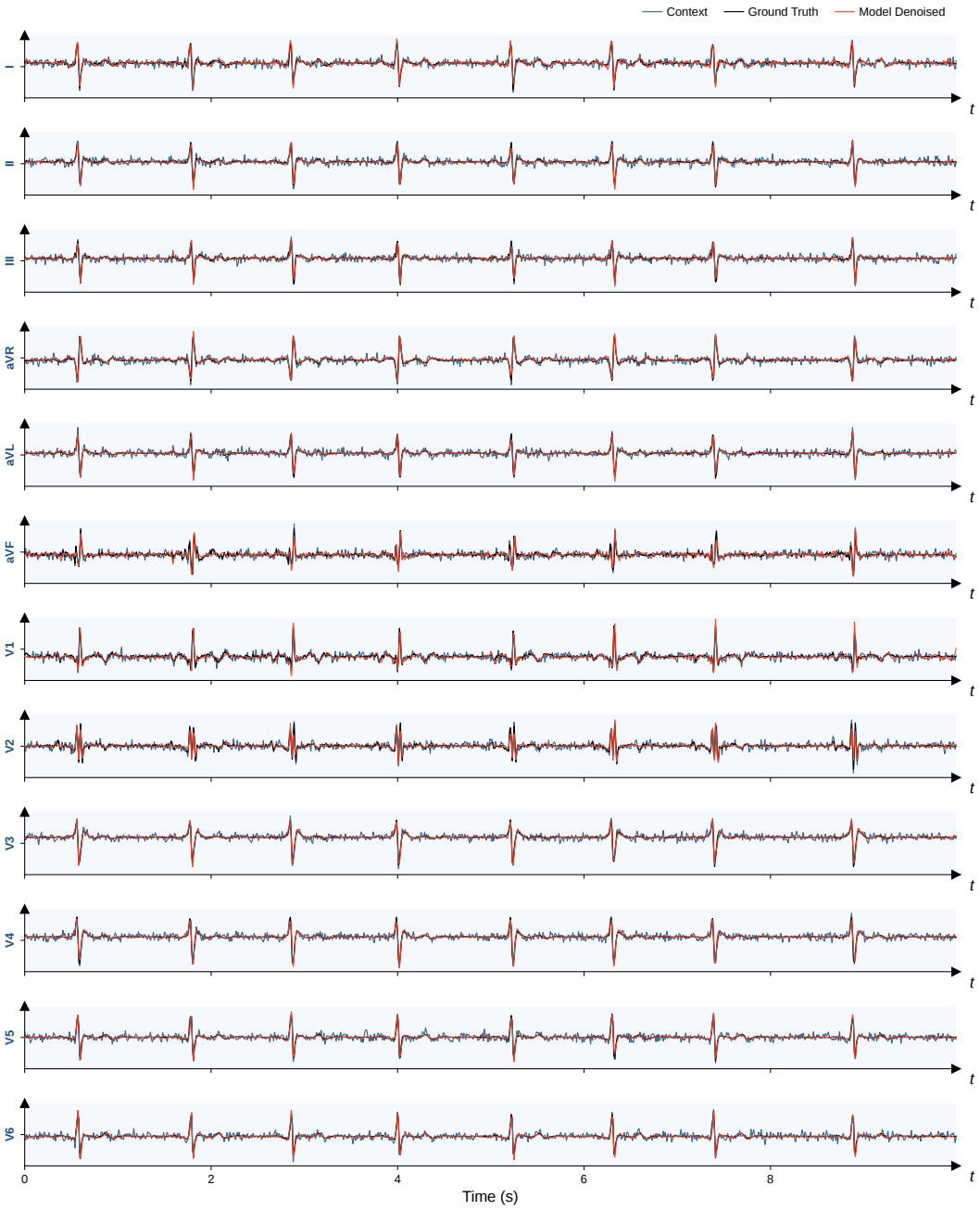

Figure 13: **Visualization of ECG denoising task.**

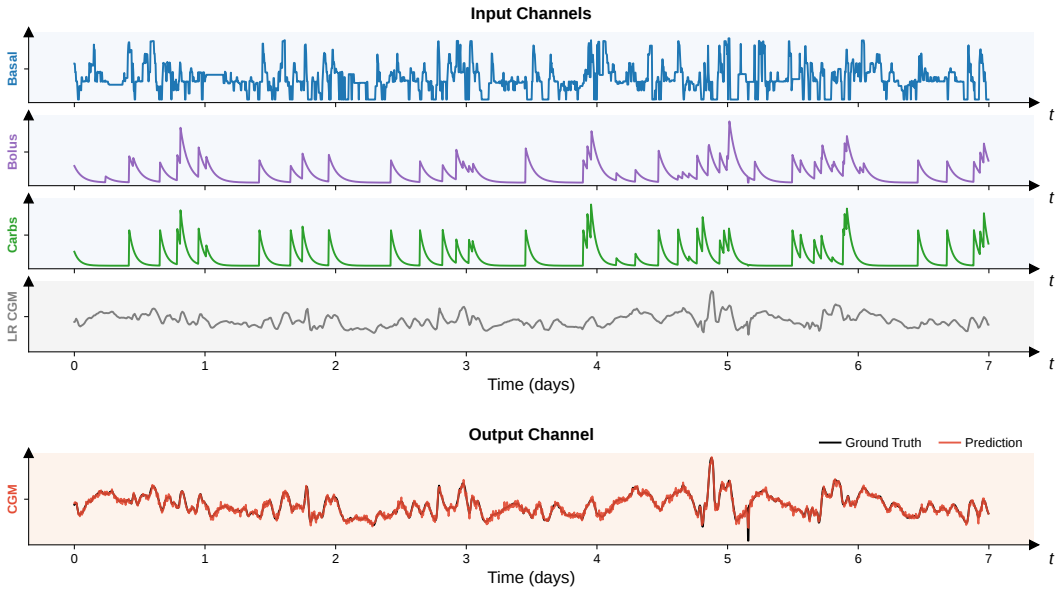

Figure 14: **Visualization of glucose signal super-resolution task.**

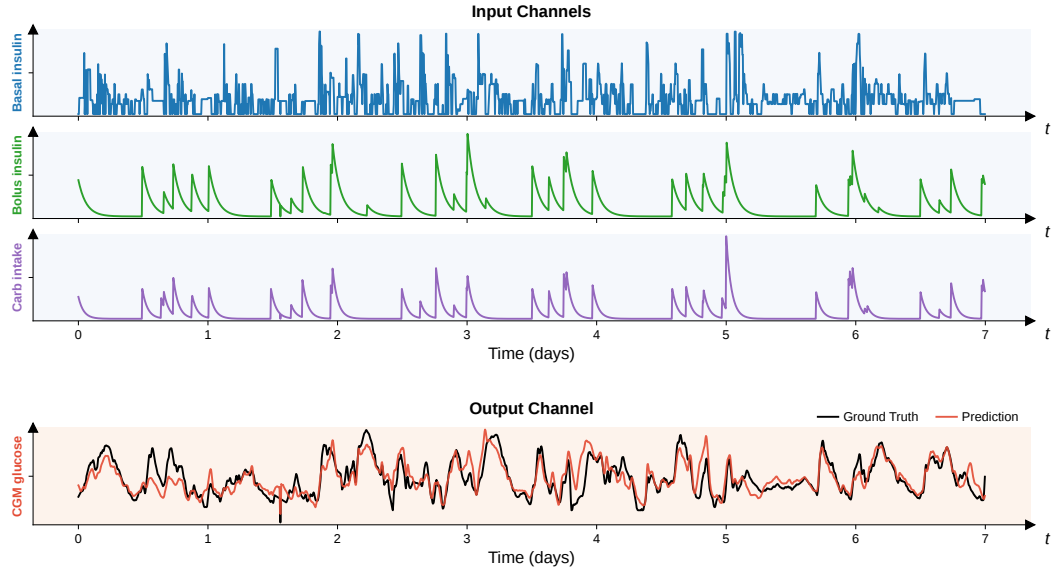

Figure 15: **Visualization of glucose signal translation task.**

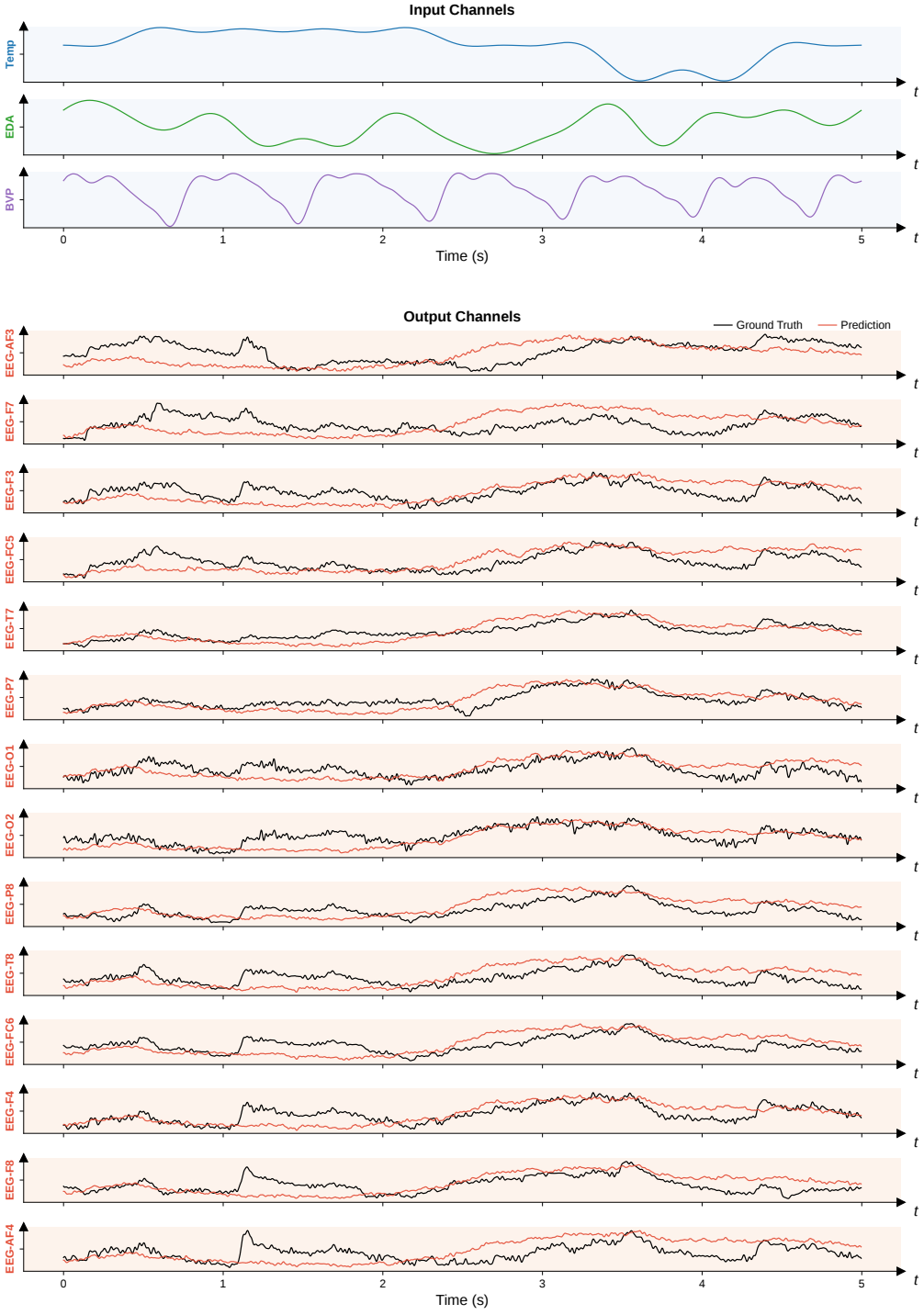

Figure 16: **Visualization of peripheral to Brain signal translation task.**

E Implementation Details

E.1 Training Settings

Shared training protocol. The main challenge in `SensorGen` is that different generative families have different native architectures, losses, samplers, and conditioning mechanisms. Appendix E therefore defines the shared experimental protocol used to make comparisons attributable to model behavior. For the main experiments, since we aim to understand how each modeling paradigm behaves in different sensor generation tasks, we didn't apply complex training and optimization techniques for each method, and avoided hyper-parameter searching if training curves are healthy and the generated samples are in reasonable morphologies. Therefore, we tried our best to align the training and evaluation settings of each method, unless otherwise specified, models are trained for up to 50K steps or until convergence with a per-rank batch size of 64, corresponding to a global batch size of 256 on $4\times$ GH200 GPUs. Since the official implementations differ in architecture design and default configurations, we tried our best to do parameter matching across different architectural families. Table 28 summarizes the model-specific parameter-scale, learning-rate, EMA, and warmup configurations used by each model family. Table 29 summarizes the training-side structure for the main benchmark tasks.

Detailed settings in analysis studies. For the analysis studies, we isolate one design factor at a time while keeping the remaining setup fixed whenever possible. For demographic encoding, we compare no demographic conditioning, raw demographic covariates, and z-score normalized demographic covariates on the CGM translation task. For long-sequence generation, we construct paired short- and long-sequence variants that differ only in generated window length. We evaluate SHHS sleep imputation with $1{,}920$ vs. $19{,}200$ time steps, CAPTURE-24 activity-to-IMU generation with 900 vs. $9{,}000$ time steps, and PPG-to-ECG translation with $1{,}000$ vs. $6{,}000$ time steps. To ensure comparable metrics, long generated signals are sliced into segments matching the corresponding short-sequence duration, and metrics are averaged across slices. For time-frequency modeling, we compare raw-waveform training with STFT-magnitude spectrogram conditioning under the same emotion-to-EEG setting. For scaling analysis, we vary training steps and model size while keeping the data, conditioning, sampling, and evaluation protocol fixed. For signal normalization, we compare z-score normalization with range-based min–max normalization to $[-1, 1]$ on the same invasive blood pressure translation task setup. For synthetic-data utility, we evaluate two uses of generated ECGs: real-plus-synthetic ECG–text pretraining for representation transfer, and real-plus-synthetic supervised classifier training, each compared against its real-only counterpart under the same downstream protocol.

E.2 Evaluation Details

In this section, we provide the evaluation protocol used in our study. Since `SensorGen` spans heterogeneous generation settings, we select metrics according to the structure of each task rather than enforcing a single metric family across all tasks. Semantic-to-signal tasks are evaluated with distribution-level feature-space metrics, paired generation tasks are evaluated with sample-level reconstruction metrics, and synthetic-data studies are evaluated through downstream utility.

Sampling protocol. Unless specified, for each model checkpoint in each task setting, we generate $N_g = 1024$ samples conditioned on held-out test examples. Unless otherwise specified, the main experiment uses either the checkpoint at the maximum training budget or the converged checkpoint within training budget. Sampling follows the default sampler provided by the official implementation of each model family.

Distribution-level metrics. For semantic-to-signal generation, multiple plausible signals may correspond to the same semantic condition. We therefore compare generated samples against the held-out real distribution using FID, Precision, and Recall in a learned signal-feature space. We use MIRA Li et al. (2025a) to extract embedding and support the calculation of these metrics. We pick these metrics because they measure distributional alignment, sample fidelity, and distributional coverage Lai et al. (2025).

Paired reconstruction metrics. For tasks with a unique paired target, we report MSE, MAE, SMSE Wang et al. (2004). MSE and MAE quantify point-wise reconstruction error, while SMSE captures spectral mismatch in log-power-spectral-density space, which is particularly informative for signals with substantial high-frequency content. Specifically, SMSE is computed as:

$$\mathrm{SMSE} = \frac{1}{NCF} \sum_{n=1}^{N} \sum_{c=1}^{C} \sum_{f=1}^{F} \left(\log(1 + \mathrm{PSD}(\widehat{x}_n)[c, f]) - \log(1 + \mathrm{PSD}(x_n)[c, f])\right)^2, \quad (1)$$

where $\widehat{x}_n$ and x_n denote the generated and ground-truth signals for the n-th sample, C is the number of channels, and F is the number of one-sided frequency bins.

Paired metrics are computed in the normalized signal space used during training, typically with per-window min–max normalization to $[-1, 1]$. Please check Appendix A for per-task normalization details.

Evaluation settings for analysis studies. For controlled analyses, we reuse the main sampling and evaluation protocol whenever possible, changing only the training variant, representation, or evaluation view required by the analysis.

Table 28: **Model capacity, training configuration, and throughput in SensorGen.** We instantiate each representative model with a comparable parameter scale whenever possible, while preserving its native architectural design and optimization convention.

Model	Total Params	Hier.	Model width	EMA	LR	Warmup	Throughput (steps/s)
DiT	≈272M	×	768	0.9998	10^{-4}	–	4.11
SiT	≈272M	×	768	0.9998	10^{-4}	–	4.38
TarFlow	≈143M	×	512	–	10^{-4}	10% of total steps	2.37
MAR	≈338M	×	768	0.9999	10^{-4}	10% of total epochs	5.71
FractalGen	≈441M	✓	1024	–	5×10^{-5}	10% of total epochs	2.51
Imagen	≈547M	✓	768	–	10^{-4}	10% of total steps	3.92

F Model Implementation Details

We adapt representative families of generative models to time-series physiological signals using a unified implementation interface. Here we describe how each family is instantiated for multi-channel sensor time series. Across all models, the input signal is represented as $x \in \mathbb{R}^{B \times C \times T}$, where B, C, and T denote the batch size, number of channels, and number of time steps, respectively.

Patchification. For transformer-based models, we replace 2-D image patchification with non-overlapping 1-D signal patchification Narayanswamy et al. (2024); Yang et al. (2023). Given a patch length P, DiT, SiT, Imagen, MAR, and FractalGen convert the signal into a token sequence $z \in \mathbb{R}^{B \times N \times D}$, where $N = T/P$ and D is the hidden dimension. For DiT, SiT, and Imagen, this is implemented with a 1-D convolutional patch embedder. The generated tokens are then projected back to the waveform domain through an unpatchification head.

Conditioning approach. We have used sample-level semantic conditions, such as text reports, labels, and metadata, denoted as c_1. And also dense signal-side conditions, such as historical signals, source channels, low-resolution references, temporal boundaries, or noisy observations, which are noted as c_2. Specifically, DiT, SiT, and Imagen use sample-level modulation for c_1 and cross-attention for c_2; MAR uses c_1 buffer tokens and decoder cross-attention for c_2; FractalGen injects conditions across hierarchical levels; and TarFlow uses token-additive sparse conditioning for c_1 with cross-attention for c_2. The codebase will be fully open after acceptance, and all implementation details can be found there.

Table 29: **Configurations of generation target and input conditions.**

Dataset	Task	Target x	Semantic condition c_1	Signal-side condition c_2	Window T	Span
MIMIC-IV ECG	Text-to-ECG	12-lead ECG	Free-text ECG report	–	1,000	10 s
PhyMER	Emotion-to-EEG	14-channel EEG	Emotion label	–	1,280	5 s
CAPTURE-24	Activity-to-IMU	3-axis accelerometer	Activity label	–	900	30 s
PPG-DaLiA	Forecasting	Future HR, ECG, and BVP	–	Historical HR, ECG, and BVP	6,000	60 s
SHHS	Sleep imputation	Masked middle segment	–	Left/right context from EEG, EOG, ECG, and EMG	1,920	30 s
VitalDB	Medication-aware forecasting	Future physiological signals	Medication trajectories	Historical physiological signals	3,000	30 s
PPG-DaLiA	PPG-to-ECG	ECG	–	PPG signal	6,000	60 s
PhyMER	Peripheral-to-EEG	14-channel EEG	–	Temperature, EDA, and BVP	1,280	5 s
VitalDB	BP reconstruction	Arterial blood pressure	Non-invasive BP summary when used	PPG signal	1,500	30 s
Metabonet	CGM translation	Continuous glucose monitoring signal	Demographic covariates	Insulin and carbohydrate histories	2,016	7 d
Metabonet	CGM super-resolution	High-resolution CGM trajectory	Demographic covariates	Low-resolution CGM, insulin, and carbohydrate histories	2,016	7 d
MIMIC-IV ECG	ECG denoising	Clean ECG	denoise	Noisy ECG	1000	10 s
MIMIC-IV ECG	ECG editing	edited ECG	editing guidance	source ECG	1000	10 s
CAPTURE-24	IMU editing	Edited IMU signal	Target activity or editing guidance	Source IMU signal	900	30 s

F.1 Diffusion Models

DiT. DiT Peebles & Xie (2023) is a representative transformer-based diffusion architecture that replaces the conventional U-Net denoiser with a scalable transformer backbone. Based on official implementation, we adapt DiT to 1-D physiological signals by replacing the 2-D image patch embedder with a 1-D patch embedder. The model is trained with the standard diffusion noise-prediction objective:

$$\mathcal{L}_{\mathrm{DiT}} = \mathbb{E}_{x_0,t,\epsilon}\left[\left\|\epsilon - \epsilon_\theta\left(x_t, t, c_1, c_2\right)\right\|_2^2\right], \tag{2}$$

where x_0 denotes the data sample and x_t denotes the noisy signal at diffusion timestep t. We follow the official implementation to inject the timestep embedding and sparse condition c_1 through adaLN-zero modulation. We inject the dense condition c_2 through cross-attention. The output tokens are projected back to waveform patches and unpatchified into the generated signal.

F.2 Flow Matching

SiT. SiT Ma et al. (2024) is a representative transformer-based flow-matching model that shares the scalable transformer design of DiT but replaces the Markovian diffusion objective with velocity prediction. We adapt SiT to 1-D physiological signals using the same patchification, positional embedding, backbone, and conditioning interface as DiT. Given a data sample x_0 and a Gaussian endpoint $x_1 \sim \mathcal{N}(0, I)$, we use a linear interpolation path:

$$x_t = (1-t)x_0 + tx_1, \qquad t \sim \mathrm{Uniform}(0,1). \tag{3}$$

The target velocity under this path is $x_1 - x_0$, and the model is trained with the flow-matching objective:

$$\mathcal{L}_{\mathrm{SiT}} = \mathbb{E}_{x_0,x_1,t}\left[\left\|v_\theta\left(x_t, t, c_1, c_2\right) - (x_1 - x_0)\right\|_2^2\right]. \tag{4}$$

F.3 AUTOREGRESSIVE MODELS

MAR. MAR Li et al. (2024) is a masked autoregressive generative model for continuous-valued data, avoiding the need for vector quantization. We adapt its image-generation implementation to 1-D physiological signals by applying masked autoregressive modeling over 1-D signal tokens. During training, the signal is patchified into a token sequence, a subset of tokens is masked, and the decoder predicts contextual representations for the masked positions. Each masked token is generated by a lightweight continuous diffusion head conditioned on its decoder representation. We inject sparse condition c_1 through prepended tokens and the global conditioning path, and inject dense condition c_2 through decoder cross-attention. At inference time, MAR generates signals by iterative unmasking, repeatedly predicting and revealing subsets of masked tokens according to the sampling schedule.

F.4 NORMALIZING FLOWS

TarFlow. TarFlow Zhai et al. (2024) is a transformer-based normalizing flow that enables exact likelihood training through invertible transformations. We adapt its official implementation to 1-D physiological signals by reshaping the waveform into non-overlapping patches. Let $z^{(0)} \in \mathbb{R}^{B\times N\times CP}$ denote the reshaped signal patches. TarFlow applies a stack of invertible transformer blocks that transform the data representation into a latent variable z. With affine coupling, each block predicts scale and shift parameters (a_ℓ, b_ℓ) and applies

$$z^{(\ell+1)} = \left(z^{(\ell)} - b_\ell\right) \odot \exp\left(-a_\ell\right), \tag{5}$$

up to the block-specific permutation.

Since the transformation is invertible with a tractable Jacobian determinant, TarFlow is trained by exact negative log-likelihood:

$$\mathcal{L}_{\text{TarFlow}} = \frac{1}{2}\mathbb{E}\left[\|z\|_2^2\right] - \mathbb{E}\left[\log\left|\det J\right|\right], \tag{6}$$

where z is the final latent variable and J is the Jacobian of the data-to-latent transformation.

We inject sparse condition c_1 as an additive sample-level token, and inject dense condition c_2 through cross-attention inside the invertible blocks. For tasks such as super-resolution, low-resolution signals are used as fixed conditioning inputs rather than concatenated as additional channels.

F.5 HIERARCHICAL MODELING METHODS

FractalGen. FractalGen Li et al. (2025b) is a hierarchical coarse-to-fine generative model. We adapt its official implementation to 1-D physiological signals by representing each signal through multiple temporal levels and recursively refining coarse tokens into finer temporal structure. Let $x^{(0)}, x^{(1)}, \ldots, x^{(K)}$ denote representations from coarse to fine levels. The hierarchical generation process can be factorized as

$$p_\theta(x) = \prod_{k=0}^{K-1} p_{\theta_k}\left(x^{(k+1)} \mid x^{(k)}, c_1, c_2\right), \tag{7}$$

where parent-level representations provide context for finer-level generation.

The upper levels of FractalGen use MAR-style iterative masked token prediction over continuous token representations, while the innermost level uses an autoregressive discretized loss over per-channel amplitude bins. Sparse conditions are injected into the top-level generator and then influence finer levels through parent-level representations, while dense conditions are passed through level-specific projection pathways when enabled. This design enables long-horizon signal generation by modeling coarse temporal structure and fine-grained waveform details in a coarse-to-fine hierarchy.

Imagen. Imagen Saharia et al. (2022) is a cascaded diffusion model for coarse-to-fine hierarchical generation. We adapt its official implementation to 1-D sensor signals by replacing the image-domain modules with the same 1-D transformer backbone and convolutional patchifier used in our DiT and

SiT implementations. In our implementation, the first stage generates a low-resolution signal, and the second stage refines it to the target resolution:

$$x_{\mathrm{LR}} \sim p_{\theta_1}(\cdot \mid c_1, c_2), \qquad x_{\mathrm{HR}} \sim p_{\theta_2}(\cdot \mid c_1, c_2, \mathrm{Up}(x_{\mathrm{LR}})). \tag{8}$$

Both stages are trained with diffusion-style noise-prediction objectives. For super-resolution, we train only the second-stage model, since the low-resolution signal is directly provided by the task. The model follows the DiT/SiT-style conditioning pathway for c_1 and c_2, with the low-resolution signal additionally provided as second-stage conditioning. In the main experiments, we use deterministic low-resolution conditioning.

F.6 SPECTROGRAM-DOMAIN VARIANTS.

For high-frequency EEG settings, we utilize raw-domain architecture augmented with STFT-magnitude spectrogram conditioning. The STFT magnitude is encoded as a dense signal-side condition, while the main generator directly predicts raw EEG waveforms. This variant evaluates whether spectrogram conditioning improves high-frequency physiological signal generation.

F.7 NAIVE BASELINES

We also include deterministic naive baselines for forecasting tasks with observed signal conditions c_2. These baselines do not train any parameters. Instead, they generate predictions by applying mean pooling to the provided historical context. For each target channel, c_2 contains the observed historical values. We pool these values over time and repeat the resulting scalar across the prediction horizon, producing a constant future trajectory for that channel. We evaluate the deterministic prediction $\widehat{\mathbf{x}}$ against the paired target signal $\mathbf{x}$ using the same sample-level metrics as the learned generators.