Do Large Foundation Models Improve Time Series Segmentation? An Industrial Case Study in Oil and Gas Drilling

Imane Khaouja [*1] **Amine El Khair** [*1] **Abdallah Benzine** [1] **Sebastiaan Buiting** [1] **Soumyadipta Sengupta** [1] **Youssef Tamaazousti** [1]

Abstract

Segmenting time series into meaningful events is critical in domains like drilling, where accurate activity recognition enables operational optimization and real-time decision-making. Yet, segmentation remains challenging due to noise and multivariate complexity. Recently, Foundation Models for Time Series (FM4TS) have emerged as general-purpose solutions, but their effectiveness for segmentation is unclear.

In this study, we benchmark popular FM4TS (both pretrained and trained from scratch) against a fully convolutional network (FCNN) baseline on two tasks: a simple univariate and a complex multivariate segmentation problem. We also assess how performance scales with data size.

Results show CNNs are strong baselines, often outperforming or matching FM4TS. Pretraining offers limited or even negative impact on FM4TS performance, highlighting challenges in transferring segment-level features. Interestingly FM4TS seems to scale better with more data, suggesting potential advantages in data-rich settings.

1. Introduction

Oil and gas drilling generates vast amounts of multivariate sensor data that measure critical parameters such as pressure, hookload, torque, and flow rate. Automatic segmentation of these data into meaningful operational phases or anomalies is crucial for safety, operational efficiency, and real-time decision making. However, accurate segmentation remains challenging due to noisy signals, irregular patterns, and complex interactions among multiple sensor channels.

Traditional methods rely heavily on statistical rules or handcrafted features (Arnaout et al., 2012; Serapião et al., 2006), often failing under realistic drilling conditions due to their limited adaptability. Deep learning methods improved accuracy by learning temporal representations directly from the data (Perslev et al., 2019; Lea et al., 2017; Ismail-Fawaz et al., 2019). Yet, these approaches typically require extensive labeled datasets and custom architectures, limiting their adoption in industrial environments (Benzine et al., 2024).

Recently, Foundation Models for Time Series (FM4TS) — including Time-MoE (Shi et al., 2024), Chronos-Bolt (Ansari et al., 2024), Timer (Liu et al., 2024), Moment (Goswami et al., 2024), Moirai (Woo et al., 2024), and GPT4TS (Zhou et al., 2023) — have emerged as general-purpose models, showing excellent results on forecasting and classification tasks. However, their performance for segmentation tasks has not yet been systematically evaluated, leaving uncertainty regarding their suitability for noisy, complex real-world industrial time series (Buiting et al., 2024).

In this work, we investigate whether pre-trained FM4TS can be directly adapted—without architecture modifications to the segmentation of oil and gas drilling data. Specifically, we evaluate several FM4TS models by attaching lightweight classification heads and fine-tuning them end-to-end. We benchmark their performance against a robust CNN baseline on two distinct segmentation tasks: a straightforward periodic task (downlinking detection) and a complex, irregular multivariate anomaly detection task (hookload anomalies). The two tasks and their difficulty is illustrasted in Figure 1.

Our findings indicate that pretrained weights offer limited or negative utility for FM4TS in segmentation, suggesting current pretraining objectives may lack segment-specificity. Furthermore, confirming results from (Buiting et al., 2024), simpler CNNs often serve as strong baselines, diminishing the perceived universal advantage of FM4TS. While these observations call for task-specific FM4TS and potentially new pretraining strategies, it is noteworthy that FM4TS demonstrate better scalability with larger datasets.

2. Methodology

The segmentation task involves assigning a categorical label to each time step within a given time series sequence.

[*]Equal contribution [1]AIQ, Abu Dhabi. Correspondence to: Imane Khaouja <imane.khaouja@aiqintelligence.ae>, Abdallah Benzine <abdallah.benzine@aiqintellingence.ae>.

Proceedings of the 42nd International Conference on Machine Learning, Vancouver, Canada. PMLR 267, 2025. Copyright 2025 by the author(s).

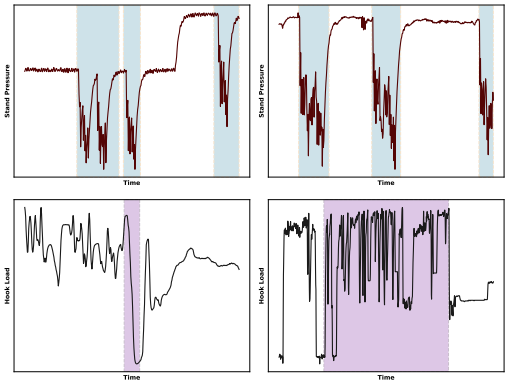

Figure 1: Example of segmentations for Downlinking (top row) and Hookload anomaly (bottom row) tasks.

This labeling identifies distinct operational phases or anomalies within drilling data sequences. Specifically, we frame the problem as a sequence-to-sequence classification task, where each input sequence consists of sensor readings and the output is a corresponding sequence of categorical labels indicating operational status or anomaly presence.

For each window, the model must accurately detect the onset and offset of operational phases or anomalous events, effectively delineating segments in a continuous stream of sensor data. The segmentation task thus requires the model not only to classify individual points but also to capture temporal dependencies and transitions between segments.

Two tasks are considered:

Downlinking Segmentation This task focuses on detecting command transmission intervals from the standpipe pressure signal. The goal is to segment the univariate time series to precisely identify intervals where downlinking commands are transmitted, characterized by distinct and periodic pressure variations. The first row of Figure 1 illustrates downlinking segments highlighted in yellow.

Hookload Anomaly Segmentation This more complex task involves identifying anomalous patterns in hookload behavior, crucial for operational safety and performance monitoring. The second row of Figure 1 shows different hookload anomaly segments highlighted in yellow. It is evaluated in two scenarios:

- **Univariate Hookload Anomaly**: Segmentation based solely on the hookload signal, focusing on detecting anomalies from hookload behavior alone.
- **Multivariate Hookload Anomaly**: Segmentation based on multiple drilling sensor inputs, including Block Position, Bit Depth, Hole Depth, Flow Rate, Rotary RPM, Torque, and Standpipe Pressure. This setup aims to capture anomalies that are contextually dependent on interactions between multiple sensor readings.

We compare two approaches:

Fully Convolutional Neural Network As a baseline, we use a fully convolutional neural network (FCNN) with an encoder-decoder structure, previously proven to be a strong benchmark for drilling-related time series analysis (Buiting et al., 2024). The encoder applies 1D convolutions and temporal pooling to capture multi-resolution patterns, while the decoder upsamples features with skip connections.

Pretrained Foundation Models We evaluate pretrained FM4TS by adding a simple classification head and then fine-tuning the entire model for segmentation tasks. For comparison, we also evaluate these models when trained entirely from scratch on the same tasks. If the model has an encoder-decoder architecture (like Chronos), only the encoder is used. The setup depends on input dimensionality:

- **Single-channel input:** The FM4TS processes the input sequence directly. The output embeddings go through a dense classification layer to predict per-time-step labels.
- **Multi-channel input:** We consider two cases:
  1. If the FM4TS supports multivariate input (like Moirai), all channels are processed jointly.
  2. If the FM4TS is originally univariate (like Chronos-Bolt), each channel is passed independently. We aggregate the resulting embeddings and feed them into a classification head.

We test Time-Moe (Shi et al., 2024), Chronos (Ansari et al., 2024), Timer (Liu et al., 2024), Moment (Goswami et al., 2024), GPT4TS (Zhou et al., 2023), and Moirai (Woo et al., 2024).

Segmentation Task Let $\mathbf{X} \in \mathbb{R}^{T \times C}$ denote a multivariate time series of length T with C sensor channels, where $\mathbf{X}_t \in \mathbb{R}^C$ is the observation at time step t. The goal of segmentation is to learn a function $f_\theta : \mathbb{R}^{T \times C} \to \{1, \ldots, K\}^T$ that maps the input sequence to a sequence of discrete labels $\mathbf{y} = (y_1, \ldots, y_T)$, where each $y_t \in \{1, \ldots, K\}$ denotes the class at time step t, and K is the number of segment classes.

The model f_θ is trained to minimize a segmentation loss over a dataset of N labeled examples $\{(\mathbf{X}^{(i)}, \mathbf{y}^{(i)})\}_{i=1}^N$.

3. Experiments

3.1. Tasks and Data

Each task uses a 60/20/20 split for training, validation, and testing. We applied minimal preprocessing beyond clipping out-of-range values and linearly interpolating occasional missing points. All signals are scaled, and sequences are fed as overlapping windows (length = 512, stride = 128). The dataset consists of over 370,000 windows extracted from real drilling operations across multiple rigs. Each window is labeled with segment boundaries by domain experts. The labeling process is manual, with experts reviewing multivariate sensor patterns to assign accurate segment classes.

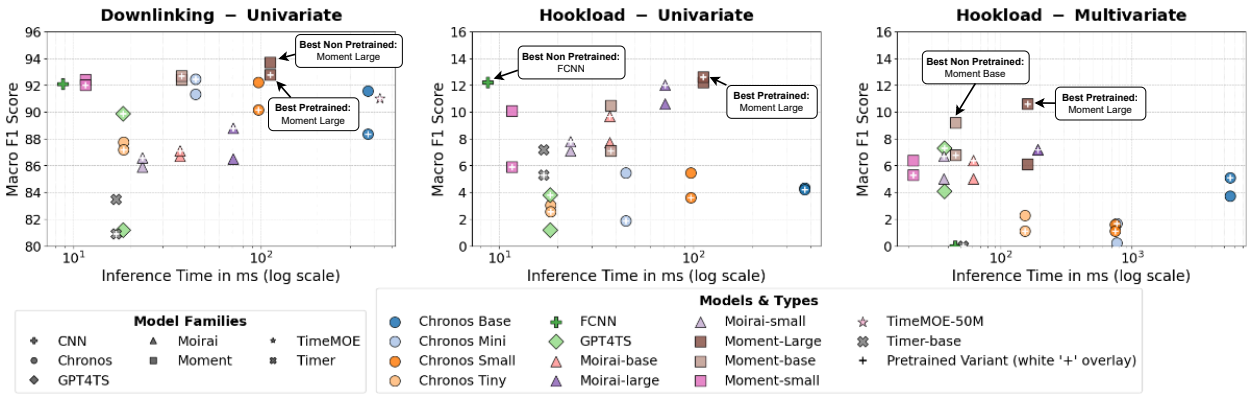

Figure 2: Comparison of the performance of large models for time series on different Drilling related tasks

3.2. Training Setup

All models are trained end-to-end with cross-entropy loss. For FM4TS, we fine-tune the entire model along with a lightweight classification head. We use the Adam optimizer with a learning rate of 2.5e-5 and a batch size of 32. Training runs for up to 25 epochs with early stopping based on F1-score on the validation set (patience = 8).

Furthermore, we conducted experiments with Frozen Weights, where pretrained model weights were kept static during fine-tuning (cf. Appendice A.1).

3.3. Evaluation Metric

We evaluate the performance using an IoU-based F1 score (Sejak et al., 2025; Redina et al., 2025). Unlike standard point-wise metrics, this approach accounts for segment alignment and overlap.

For each predicted segment, we compute its Intersection over Union (IoU) with all ground truth segments of the same class. A prediction is considered a true positive (TP) if its IoU with any ground truth segment exceeds 0.5. Unmatched predictions are counted as false positives (FP), and unmatched ground truth segments as false negatives (FN).

4. Results

4.1. Performance with the Full Training Set

Figure 2 compares macro–F_1 scores and inference times across all models and tasks with 100% training data.

Downlinking For the Downlinking task, top-performing models achieve high F1 scores. The FCNN model reaches a 92.1% F1 score with lower inference time, performing just below Moment-Large (92.8% F1), which incurs a higher inference cost. Increasing model size beyond these generally offers no significant performance improvement. This suggests that for such pressure-based tasks with abundant data, a relatively simple convolutional architecture is sufficient.

Hookload, Univariate. A notable decrease in F_1 scores is observed for all models on this task. Despite this general trend, the FCNN demonstrates strong performance, achieving a 12.2% F_1 and surpassing nearly all larger transformer models, coupled with a lower inference cost. While pretraining provides a modest uplift for certain models like Moment-Large (e.g., to 12.6% F_1) and Moirai Large (to 12.0% F_1) compared to their from-scratch counterparts (e.g., 12.2% and 10.6% respectively), it does not bridge the performance gap to the FCNN considering the inference time.

Hookload, Multivariate. On the Hookload Multivariate task, the benefit of additional features is not consistently substantial. Moment Large pretrained performs best (10.6%), slightly ahead from-scratch smaller counterpart (Moment Base 9.2%). FCNN fails to capture the noisy cross-sensor patterns in this task. Larger models learn to leverage the added context, but the gains remain limited.

Model family observations. FCNN is strong on Downlinking and Hookload Univariate despite its simplicity. Moment is the most consistent across tasks, with small benefits from pretraining in the univariate case. Chronos-Bolt excels on DL but degrades on both hookload setups. Moirai, GPT4TS, Time-Moe and Timer fall behind across all tasks. Also, Encoder based models (Moirai, Moment and Chronos-Bolt) seem to perform better than decoder-only models (Time-MoE, Timer and GPT4TS) for segmentation.

On Pretraining. Its benefit is inconsistent. It adds 2–3 points on Hookload Univariate and Multivariate for some models, has no measurable impact on Downlinking.

Key Takeaways. Larger models do not reliably benefit once enough training data is available.

Task complexity limits performance more than model size. Pretraining offers limited help, and the FCNN baseline outperforms or matches large transformers on two out of three tasks, while being smaller, faster, and easier to deploy.

4.2. Limited Supervision: Does Pretraining Pay Off?

As shown previously, generic pretraining has a marginal impact given a full training set. We now use Figure 3 and Table 1 to test if a smaller label budget alters this finding. Note

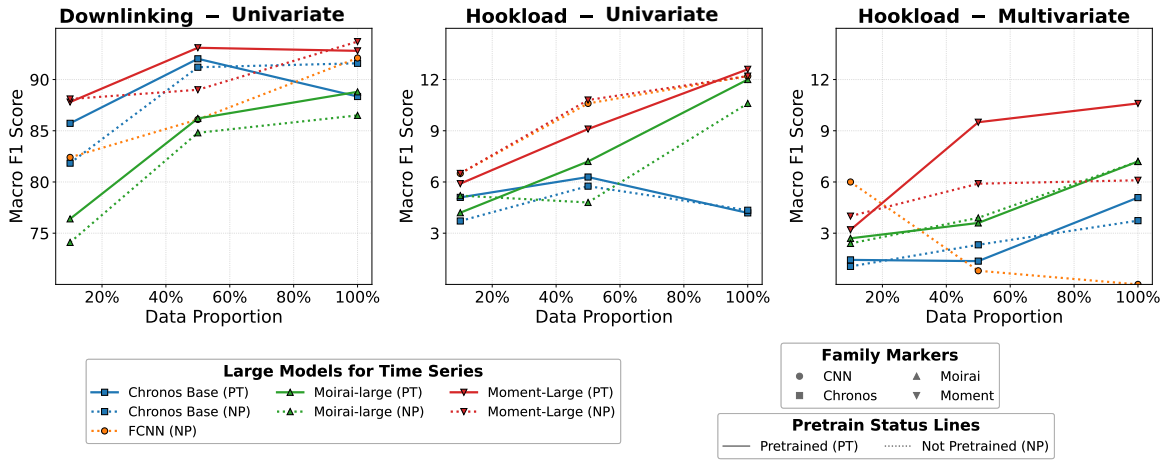

Figure 3: Comparison of segmentation performance across models, data sizes, and tasks.

Table 1: Mean Δ F1 by Family, Data Proportion, and Task. **Values are averaged across all model sizes within each family.** (Performance gains are color-coded: green for positive, red for negative, and orange for minimal gain)

Family	Downlinking - Univariate			Hookload - Univariate			Hookload - Multivariate		
	10%	50%	100%	10%	50%	100%	10%	50%	100%
Chronos	-7.01	-1.99	-1.19	-0.04	-0.84	-1.51	0.14	-0.86	0.30
GPT	55.0	86.3	8.7	1.0	1.6	2.6	1.9	0.9	3.2
Moirai	-1.03	0.57	1.13	0.20	1.87	1.37	-0.07	-0.27	1.03
Moment	0.17	1.67	-0.33	-0.73	-0.17	-2.40	-0.70	0.67	0.33
Timer	0.50	-2.40	-2.60	1.10	-3.30	-1.90	-2.40	-5.80	0.00

that Figure 3 displays only top-performing models to maintain plot readability. Table 1 presents the mean difference in F_1 scores (Δ F1) between pretrained and non-pretrained models within each FM4TS family, across different data proportions and tasks. A clear pattern emerges for self-supervised pretraining on time series data, which generally shows underperformance or marginal benefit, distinguishing it from models like GPT that leverage weights from pretraining on other modalities like natural language.

Downlinking. At 10% data the gains are marginal for pretrained vs non-pretrained: Moment-Small climbs by 0.6 points; Moirai-Small actually drops by six. The clearest benefit of pretraining for Downlinking, as also suggested by the positive average Δ F1 for Moment and Moirai in Table 1 at 50% data: Moment-Large adds 4.1 points (93.1 vs 89.0). At 100% top performing model is a non pretrained model.

Hookload, Univariate. The pattern flips. With 10% data, pretrained Moment-Large falls below its scratch twin (5.9 vs 6.5), and the same reversal appears for Chronos-Base. At 50% data, the picture stays mixed; Moirai is the only family showing a consistent positive average F1 in Table 1, with Moirai-Large gaining 2.4 points from pretraining, while Moment-Large loses 1.7. The FCNN, with no pretraining, achieves a strong 12.2% F_1 score, closely rivaling the top-performing Moment-Large (pretrained).

Hookload, Multivariate. Using all channels does not stabilize the performance. Moment models pretraining is a net negative at 10% (-0.7) split. The gain with pretraining remains very limited. While pretraining offers a significant performance uplift for Moment Large, its advantages are otherwise inconsistent or negative for other Moment variants and data proportions (Table 2). Meanwhile, the FCNN model struggles with multi-channel data; while its F_1 score reaches 6% at the 10% data split, it fails to exceed 1% at higher data proportions (50% and 100%).

Key Takeaways. While pretraining offers a moderate boost (up to four) on the easiest task with mid-scale data, its effect on noisier hookload problems remains inconsistent. Even at 100% data, the benefits of pretraining are often negligible. In fact, in 9 of the 15 settings in Table 1, pretraining either degrades performance or yields a marginal gain of no more than 0.4%, reinforcing that sufficient labeled data and careful inductive bias can match or outperform large pretrained checkpoints. However, FM4TS exhibit a key distinction in data scaling, appearing to leverage larger datasets more effectively than CNNs, which suggests their potential advantage in data-rich scenarios.

5. Discussion and Conclusion

We benchmarked pretrained Foundation Models for Time Series (FM4TS) on drilling segmentation tasks and compared them to a simple CNN. Pretraining showed limited or inconsistent benefits. In many cases, the CNN matched or outperformed FM4TS despite being smaller and faster.

These findings have practical implications. For real-time deployment—where latency, compute, and reliability matter—lightweight CNNs are a better fit. FM4TS, while promising at scale, may be too costly for edge environments unless task-specific pretraining is introduced.

Overall, segmentation still benefits more from inductive biases and labeled data than from generic pretraining. Future work should explore efficient pretraining objectives, and hybrid strategies that balance scalability with field constraints.

References

Ansari, A. F., Stella, L., Turkmen, C., Zhang, X., Mercado, P., Shen, H., Shchur, O., Rangapuram, S. S., Arango, S. P., Kapoor, S., et al. Chronos: Learning the language of time series. *arXiv preprint arXiv:2403.07815*, 2024.

Arnaout, A., Fruhwirth, R., Esmael, B., and Thonhauser, G. Intelligent real-time drilling operations classification using trend analysis of drilling rig sensors data. In *SPE Kuwait Int. Petroleum Conf. and Exhibition (KIPCE)*, 2012. doi: 10.2118/163302-MS.

Benzine, A., Tamaazousti, Y., Vadakkekalam, S., Balakrishnan, S., Chraibi, I., Ezzeddine, D., Arnaout, A., Khambete, S. P., Bimastianto, P., Muhammad, S. D., et al. Ai-automated codification-qc model for daily drilling reports. In *Abu Dhabi International Petroleum Exhibition and Conference*, pp. D041S154R001. SPE, 2024.

Buiting, J., Sengupta, S., Gupta, B., Tamaazousti, Y., et al. When larger isn't better: Lightweight cnns outperform large time-series models in classification of oil and gas drilling data. In *NeurIPS Workshop on Time Series in the Age of Large Models*, 2024.

Goswami, M., Szafer, K., Choudhry, A., Cai, Y., Li, S., and Dubrawski, A. Moment: A family of open time-series foundation models. *arXiv preprint arXiv:2402.03885*, 2024.

Ismail-Fawaz, H., Forestier, G., Weber, J., Idoumghar, L., and Muller, P. Deep learning for time series classification: A review. *Data Mining and Knowledge Discovery*, 33(4): 917–963, 2019. doi: 10.1007/s10618-019-00619-1.

Lea, C., Flynn, M. D., Vidal, R., Reiter, A., and Hager, G. D. Temporal convolutional networks for action segmentation and detection. In *Proc. IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*, pp. 1003–1012, 2017. doi: 10.1109/CVPR.2017.113.

Liu, Y., Zhang, H., Li, C., Huang, X., Wang, J., and Long, M. Timer: generative pre-trained transformers are large time series models. In *Proceedings of the 41st International Conference on Machine Learning*, pp. 32369–32399, 2024.

Perslev, M., Jensen, M. H., Darkner, S., Jennum, P. J., and Igel, C. U-time: A fully convolutional network for time series segmentation applied to sleep staging. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2019.

Redina, R., Hejc, J., Filipenska, M., and Starek, Z. Analyzing the performance of biomedical time-series segmentation with electrophysiology data. *Scientific Reports*, 15 (1):11776, 2025.

Sejak, M., Mivalt, F., Sladky, V., Vsiansky, V., Carvalho, D. Z., St. Louis, E., Worrell, G., and Kremen, V. Open-spindlenet: An open-source deep learning network for reliable sleep spindle detection. *medRxiv*, pp. 2025–04, 2025.

Serapião, A. B. S., Tavares, R. M., Mendes, J. R. P., and Guilherme, I. R. Classification of petroleum well drilling operations using support vector machine (svm). In *Proc. Int. Conf. on Computational Intelligence for Modeling, Control and Automation (CIMCA)*, 2006. doi: 10.1109/CIMCA.2006.66.

Shi, X., Wang, S., Nie, Y., Li, D., Ye, Z., Wen, Q., and Jin, M. Time-moe: Billion-scale time series foundation models with mixture of experts. *arXiv preprint arXiv:2409.16040*, 2024.

Woo, G., Liu, C., Kumar, A., Xiong, C., Savarese, S., and Sahoo, D. Unified training of universal time series forecasting transformers. arxiv 2024. *arXiv preprint arXiv:2402.02592*, 2024.

Zhou, T., Niu, P., Sun, L., Jin, R., et al. One fits all: Power general time series analysis by pretrained lm. *Advances in neural information processing systems*, 36:43322–43355, 2023.

A. Appendix

A.1. Segmentation Performance Analysis with Frozen Pretrained Weights

This section details the experimental results when fine-tuning the models with their pretrained backbone weights frozen, only training a new classification head. The corresponding F1 scores are presented in Table 2.

Freezing the pretrained weights and only training a classification head leads to a drastic reduction in performance across almost all models, tasks, and data portions compared to end-to-end fine-tuning or even training from scratch in some cases.

Specifically for the Downlinking (DL) task: Most models, including all Chronos-Bolt variants (Tiny, Mini, Small, and Base), exhibit extremely poor performance, often resulting in F1 scores of 0.0, especially with 10% and 50% data under the frozen weights setting. This indicates that the generic features learned during pretraining, without further adaptation of the backbone, are insufficient for this task. Moment models (Small, Base, Large), Timer, and GPT4TS show some capability when 100% of the training data is available, achieving F1 scores such as 40.7% (Moment Small), 55.9% (Moment Base), 85.3% (Moment Large), 52.9% (Timer), and 80.9% (GPT4TS). However, these scores are still generally lower than their counterparts fine-tuned end-to-end. For instance, Moment Large (DL, 100%) drops from 92.8% (Pretrained Weights) to 85.3% (Frozen Weights). GPT4TS shows some resilience with 50% data (72.5% F_1), but its performance at 10% is 0.0. The TimeMOE model also scores 0.0 at 10% and 50% data, reaching only 0.9% with 100% data under frozen weights for Downlinking. Moirai models (Small, Base, Large) also struggle significantly with frozen weights, with F1 scores remaining very low (e.g., Moirai Large at 5.8% with 100% data).

For the Hookload Multivariate (HL ALL) and Hookload Univariate (HL ONLY) tasks with frozen weights: The performance degradation is even more pronounced. Nearly all models, including the entire Moment family (Small, Base, and Large), yield F1 scores at or near 0.0 across all data rates (10%, 50%, 100%) for the Hookload Multivariate (HL ALL) task when weights are frozen. For instance, even with 100% data, Moment Small scores only 3.2%, Moment Base 1.9%, and Moment Large 0.0% for HL ALL with frozen weights. This is a stark contrast to scenarios with full fine-tuning; for example, Moment Large (HL ALL, 100%) drops from **10.6%** (Pretrained Weights) or 6.1% (From Scratch) to 0.0% (Frozen Weights). Chronos-Bolt variants, FCNN, Moirai, and Timer models consistently score at or near 0.0 for both Hookload tasks with frozen weights. GPT4TS also fails on Hookload tasks with frozen weights, scoring 0.0 on Hookload Univariate and low single digits (e.g., 4.5% at 100% data) on Hookload Multivariate. TimeMOE similarly performs poorly on Hookload tasks with frozen weights, with scores near zero.

These results suggest that for the evaluated downstream tasks (Downlinking, Hookload Multivariate, and Hookload Univariate), the features learned by the pretrained foundation models are not directly transferable. Effective adaptation requires fine-tuning the entire model, allowing the pretrained representations to be adjusted for the specific nuances of the target task and dataset. The frozen weight strategy proves largely ineffective, highlighting the importance of end-to-end fine-tuning for achieving optimal performance in segmentation.

Table 2: Comparison of F1 scores for various models across different training strategies (Pretrained, From Scratch, Frozen) and data availability rates. Tasks include Downlinking (DL), Hookload Multivariate (HL ALL), and Hookload Univariate (HL ONLY). Bold values indicate the best performance in each column.

		PRETRAINED WEIGHTS			FROM SCRATCH			FROZEN WEIGHTS		
Model	Data	DL	HL ALL	HL ONLY	DL	HL ALL	HL ONLY	DL	HL ALL	HL ONLY
FCNN (Baseline)	10%	-	-	-	82.4	**6.0**	6.5	0.0	0.0	0.0
	50%	-	-	-	86.1	0.8	10.6	0.0	0.0	0.0
	100%	-	-	-	92.1	0.0	12.2	0.0	0.0	0.0
Chronos-Bolt Tiny	10%	56.6	1.6	2.5	65.9	1.6	2.8	0.0	0.0	0.0
	50%	74.6	0.2	4.2	81.3	1.0	3.8	0.0	0.0	0.0
	100%	87.7	1.1	2.5	87.7	2.3	3.0	0.0	0.0	0.0
Chronos-Bolt Mini	10%	65.4	1.7	3.9	83.8	2.0	2.3	0.0	0.0	0.0
	50%	85.6	0.7	2.1	88.2	1.9	3.9	0.0	0.0	0.0
	100%	92.4	1.1	1.8	91.3	0.2	5.4	0.0	0.0	0.0
Chronos-Bolt Small	10%	79.5	2.2	2.7	83.7	1.0	3.8	0.0	0.0	0.0
	50%	90.7	0.8	5.6	90.2	2.7	6.4	0.0	0.0	0.0
	100%	90.1	1.1	3.6	92.2	1.6	5.4	0.0	0.0	0.0
Chronos-Bolt Base	10%	85.7	**5.1**	6.6	81.4	1.1	3.7	0.0	0.0	0.0
	50%	92.0	1.4	6.2	91.2	2.3	5.7	0.0	0.0	0.0
	100%	88.3	1.4	4.1	91.5	3.7	4.3	0.0	0.0	0.0
Moment Small	10%	86.3	2.9	4.7	85.7	3.2	4.7	0.0	0.0	3.4
	50%	92.0	5.6	7.0	90.8	5.6	6.8	0.0	0.0	3.5
	100%	92.0	5.3	5.9	92.4	6.4	10.1	40.7	3.2	4.2
Moment Base	10%	84.9	2.5	4.1	84.7	3.5	5.7	0.0	0.0	2.8
	50%	90.0	5.3	7.9	90.3	**6.9**	6.9	0.0	0.0	3.2
	100%	92.7	6.8	7.1	92.4	**9.2**	10.5	55.9	1.9	**5.1**
Moment Large	10%	**87.8**	3.2	**5.9**	**88.1**	4.0	**6.5**	0.0	0.0	**4.8**
	50%	**93.1**	**9.2**	**9.1**	**91.2**	5.9	**10.8**	0.0	0.0	**3.8**
	100%	**92.8**	**10.6**	**12.6**	**93.7**	6.1	**12.2**	**85.3**	0.0	2.4
Moirai Small	10%	64.4	1.1	4.7	70.4	2.1	3.1	1.9	0.0	0.4
	50%	82.0	2.6	6.7	81.7	2.3	6.3	0.0	0.0	0.0
	100%	86.6	6.7	7.8	85.9	5.0	7.1	1.5	0.0	0.0
Moirai Base	10%	74.2	2.2	3.8	73.6	1.7	3.8	1.6	0.0	0.4
	50%	84.0	2.2	9.2	84.0	3.0	6.4	0.0	0.0	0.0
	100%	87.1	6.4	9.7	86.7	5.0	7.7	0.0	0.0	0.0
Moirai Large	10%	76.4	2.7	4.2	74.1	2.4	5.2	**5.8**	1.6	2.5
	50%	86.2	3.6	7.2	84.8	3.9	4.8	0.3	0.0	0.09
	100%	88.8	7.2	12.0	86.5	7.2	10.6	0.2	0.0	0.0
Timer	10%	75.2	0.0	4.5	74.7	2.4	3.4	0.0	0.0	3.4
	50%	79.6	0.0	4.4	72.2	5.8	7.7	0.0	0.0	3.4
	100%	80.9	0.0	5.3	83.5	0.0	7.2	52.9	0.0	2.7
GPT4TS	10%	55.0	4.7	2.5	0.0	2.8	1.5	0.0	3.3	0.0
	50%	86.3	4.8	3.2	0.0	3.9	1.6	**72.5**	4.2	0.0
	100%	89.9	7.3	3.8	81.2	4.1	1.2	80.9	4.5	0.0
TimeMOE	10%	80.4	1.2	0.9	85.9	1.1	3.7	0.0	0.5	0.4
	50%	82.7	2.5	3.6	88.8	2.2	3.5	0.0	0.0	0.4
	100%	91.0	4.2	3.6	92.5	2.6	5.6	0.9	0.0	0.2