RHYTHM: Reasoning with Hierarchical Temporal Tokenization for Human Mobility

Haoyu He[†] **Haozheng Luo**[‡] **Yan Chen**[‡] **Qi R. Wang**[†]
[†] Northeastern University [‡] Northwestern University
{he.haoyu1, q.wang}@northeastern.edu
hluo@u.northwestern.edu, ychen@northwestern.edu

Abstract

Predicting human mobility is inherently challenging due to complex long-range dependencies and multi-scale periodic behaviors. To address this, we introduce **RHYTHM** (Reasoning with Hierarchical Temporal Tokenization for Human Mobility), a unified framework that leverages large language models (LLMs) as general-purpose spatio-temporal predictors and trajectory reasoners. Methodologically, RHYTHM employs temporal tokenization to partition each trajectory into daily segments and encode them as discrete tokens with hierarchical attention that captures both daily and weekly dependencies, thereby quadratically reducing the sequence length while preserving cyclical information. Additionally, we enrich token representations by adding pre-computed prompt embeddings for trajectory segments and prediction targets via a frozen LLM, and feeding these combined embeddings back into the LLM backbone to capture complex interdependencies. Computationally, RHYTHM keeps the pretrained LLM backbone frozen, yielding faster training and lower memory usage. We evaluate our model against state-of-the-art methods using three real-world datasets. Notably, RHYTHM achieves a **2.4%** improvement in overall accuracy, a **5.0%** increase on weekends, and a **24.6%** reduction in training time. Code is publicly available at https://github.com/he-h/rhythm.

1 Introduction

Human mobility shapes transportation systems [6, 62], informs epidemic control strategies [63, 8], and guides sustainable city planning [4], making accurate movement prediction essential for optimizing infrastructure, managing disease spread, and building resilient communities [42]. Yet human trajectories exhibit long-range dependencies, spatial heterogeneity [78, 19], and dynamic influences such as weather anomalies or special events [7, 36], producing non-stationary, multi-scale spatio-temporal patterns.

To address this challenge, we introduce **RHYTHM** (Reasoning with Hierarchical Temporal Tokenization for Human Mobility), a human mobility foundation model that rethinks mobility modeling via structured temporal abstraction. We posit that human mobility, like language, follows compositional structures: daily routines form tokens, and weekly rhythms form higher-order syntax. RHYTHM operationalizes this analogy by tokenizing time into segments and reasoning over them with a frozen large language model (LLM). This framework unites multi-scale temporal tokenization with the reasoning capabilities of pretrained LLMs, delivering a scalable and general approach for mobility prediction with reduced computational cost.

The design of RHYTHM is inspired by inherent patterns of human mobility. *People's movements are not random; they follow an underlying order marked by recurring daily and weekly rhythms* [12, 24, 18]. Notably, Song et al. [58] quantify this regularity by showing that 93% of daily trajectories are

39th Conference on Neural Information Processing Systems (NeurIPS 2025).

predictable, underscoring the critical role of cyclical temporal context in mobility modeling. Capturing this cyclical regularity requires models that can jointly represent local behaviors (e.g., morning commutes) and global temporal dependencies (e.g., weekly routines). Yet existing approaches fall short: Markov and RNN-based methods either disregard long-term periodicity or suffer vanishing gradients over long sequences [76, 19, 22], while transformer-based methods treat time as static, failing to disentangle multi-scale temporal patterns [27, 72]. To bridge this gap, we decompose each trajectory into meaningful segments, tokenizing each into discrete representations that capture local patterns through intra-segment attention. These segment tokens are then pooled into higher-level representations, enabling inter-segment attention to model long-range dependencies across days, as illustrated in Figure 1, thereby reducing sequence length and quadratically lowering attention cost. Each segment token is augmented with pre-computed semantic embeddings derived from a frozen LLM, enriching temporal tokens with contextual meaning before being processed by the LLM backbone for reasoning.

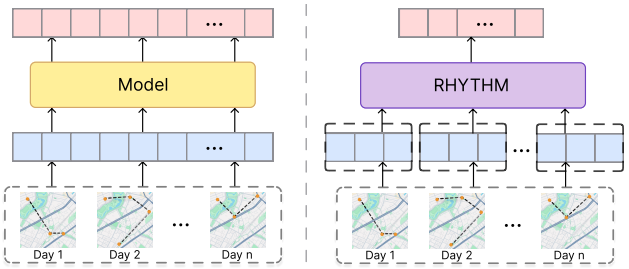

Figure 1: **Motivation for RHYTHM.** Instead of processing entire trajectories as a continuous sequence, RHYTHM segments them into tokens to better capture periodic patterns.

Recent advances demonstrates LLMs' remarkable capabilities not only as sequential representation extractors to capture the spatio-temporal attention patterns but also as reasoning models [13, 9]. Prior works [25, 59, 17, 21] demonstrate their reasoning capabilities through techniques such as few-shot prompting [9], chain-of-thought reasoning [50, 49, 66], and in-context learning [16]. However, mobility-specific models like PMT [71] and ST-MoE-BERT [26] lack the capability to leverage LLMs for modeling complex correlations in human flows, limiting their predictive performance. By integrating an LLM-based reasoning module, RHYTHM more effectively models these complex interdependencies. To maintain scalability, RHYTHM adopts a parameter-efficient adaptation strategy by freezing the pretrained LLM and avoid extensive fine-tuning. This design captures fine-grained spatio-temporal dynamics, deep semantic context, and leverages LLM reasoning—all while minimizing computational and memory overhead—making RHYTHM ideally suited for deployment in resource-constrained, real-world environments.

Contributions. We propose **RHYTHM**, a unified, computationally efficient framework that captures both temporal dynamics and cyclical patterns, as illustrated in Figure 2. Our contributions are as follows:

- We introduce temporal tokenization that encodes daily mobility patterns as discrete tokens, reducing the processed sequence length while capturing cyclical and multi-scale mobility dependencies through hierarchical attention mechanism.
- We design an efficient prompt-guided approach that integrates semantic trajectory information and task description with segment embeddings, enhancing RHYTHM's ability to interpret complex mobility patterns.
- We propose a parameter-efficient adaptation strategy using frozen pretrained LLMs, reducing trainable parameters to **12.37%** of the full model size and achieving a **24.6%** reduction in computational cost compared to other baselines.
- Empirically, we evaluate RHYTHM on three real-world mobility datasets, demonstrating superior performance compared to state-of-the-art models. RHYTHM achieves a **2.4%** improvement in prediction accuracy, with a **5.0%** increase on weekends.

2 Related Work

In this section, we provide a brief overview of related work, with a detailed review in Appendix B.

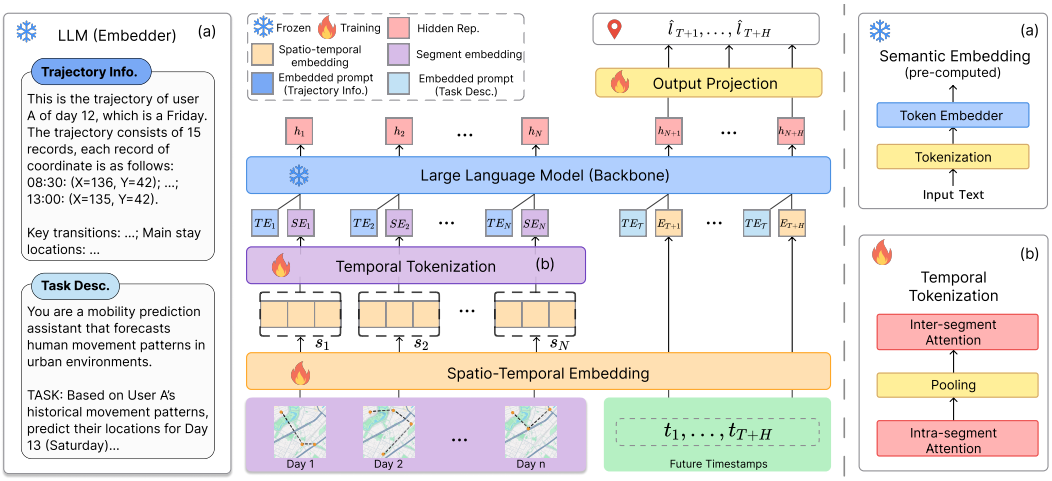

Figure 2: **The proposed architecture of RHYTHM.** Our framework processes historical trajectories through spatio-temporal embedding and temporal tokenization (**b**), capturing local and global dependencies via hierarchical attention. Segment representations are enriched with semantic embeddings from trajectory information, while future timestamps incorporate task description context (**a**). This combined sequence passes through a frozen LLM backbone with output projection to generate location predictions.

Mobility Prediction. Human mobility prediction progresses from probabilistic approaches [75, 22] to deep learning architectures, as demonstrated in recent studies. Sequence models like LSTM [37] and attention mechanisms [19] improve temporal modeling, while graph-based methods [55, 14] integrate spatial relationships. Transformer architectures [65, 77, 45] further enhance long-range dependency modeling but struggle with the hierarchical structure of mobility patterns. Recent work with LLMs [20, 64] shows promise but typically treats mobility as generic sequences, ignoring the inherent periodicity of human movement.

Cross-domain Adaptation of LLMs. LLMs emerge as powerful natural language processing systems and quickly evolve into general-purpose foundation models capable of reasoning and generation tasks [9, 1]. Their remarkable adaptability has enabled successful applications in computer vision [5, 53], speech [69, 46], biomedicine [82, 44, 57], time series forecasting [11, 81], and finance [31, 70]. While many adaptations rely on parameter-efficient fine-tuning methods like LoRA [29], recent approaches maintain frozen LLMs by utilizing them as sequential representation extractors, preserving their semantic capabilities while reducing computational costs [40, 32, 2]. To the best of our knowledge, RHYTHM is the **first** approach that adapts frozen LLMs to mobility prediction without compromising the model's reasoning capabilities or requiring extensive fine-tuning.

3 Method

In this section, we introduce **RHYTHM**, an LLM-based deep architecture tailored for prompt-guided representation learning of spatio-temporal patterns with its periodicity, as shown in Figure 2. In the following, we first define the problem and then introduce the model structure of RHYTHM, including its computational efficiency and theoretical guarantees.

3.1 Problem definition

Let $\mathcal{X} = \{x_1, x_2, \ldots, x_T\}$ denote a user's historical trajectory, where each $x_i = (t_i, l_i)$ consists of a timestamp t_i and a location $l_i \in \mathcal{L}$ from a finite set of locations $\mathcal{L}$. Given a sequence of future timestamps $\mathcal{T} = \{t_{T+1}, t_{T+2}, \ldots, t_{T+H}\}$ with prediction horizon H, the goal is to predict the corresponding future locations $\mathcal{Y} = \{l_{T+1}, l_{T+2}, \ldots, l_{T+H}\}$. Formally, we seek a function

$$f : (\mathcal{X}, \mathcal{T}) \mapsto \mathcal{Y},$$

which maps historical trajectories and future timestamps to the user's future locations.

3.2 Model structure

Spatio-Temporal Feature Encoding. For each observation x_i, we construct temporal embeddings to capture cyclical patterns in human movements:

$$\mathbf{E}_i^{\text{temporal}} = \mathbf{E}^{\text{ToD}}(t_i) \| \mathbf{E}^{\text{DoW}}(t_i),$$

where $\cdot\|\cdot$ indicates concatenation, $\mathbf{E}^{\text{ToD}}$ represents the time-of-day embedding (capturing 24-hour cycles), and $\mathbf{E}^{\text{DoW}}$ represents the day-of-week embedding (capturing weekly patterns). These are learnable embeddings that map discrete temporal indices to continuous representations, with $\mathbf{E}_i^{\text{temporal}} \in \mathbb{R}^D$ with D matching the backbone LLM's input dimension.

Spatial embeddings $\mathbf{E}_i^{\text{spatial}} \in \mathbb{R}^D$ for location l_i is defined as:

$$\mathbf{E}_i^{\text{spatial}} = \mathbf{E}^{\text{Loc}}(l_i) \| (W_{\text{coord}}[\text{lat}_i, \text{lon}_i]^T + b_{\text{coord}}),$$

where $\mathbf{E}^{\text{Loc}}$ denotes the categorical location embedding, and the second term projects the geographic coordinates $(\text{lat}_i, \text{lon}_i)$ into the embedding space. Here, $W_{\text{coord}} \in \mathbb{R}^{d_{\text{coord}} \times 2}$ is the projection matrix and d_{coord} denotes the projected dimension.

The spatio-temporal embedding $\mathbf{E}_i \in \mathbb{R}^D$ is obtained by element-wise addition:

$$\mathbf{E}_i = \mathbf{E}_i^{\text{temporal}} + \mathbf{E}_i^{\text{spatial}}.$$

For future timestamps without known locations or missing historical records, the spatial component is set to zero, allowing the model to operate on temporal information alone while preserving dimensional consistency.

Temporal Tokenization. Human mobility patterns exhibit inherent multi-scale temporal structures that span both short-term routines (e.g., daily activities) and long-term periodicities (e.g., weekly rhythms) [58, 24]. To effectively model these dynamics, RHYTHM employs a temporal tokenization mechanism that effectively disentangles local patterns from global dependencies, inspired by Liu et al. [40]. Formally, we partition the embedded sequence $\mathcal{X}$ into N non-overlapping segments $\{\mathbf{s}_1, \mathbf{s}_2, \ldots, \mathbf{s}_N\}$, each capturing meaningful temporal intervals (e.g., daily patterns):

$$\mathbf{s}_i = \{E_{(i-1)L+1}, E_{(i-1)L+2}, \ldots, E_{iL}\} \quad \text{for } i = 1, 2, \ldots, N,$$

where each segment $\mathbf{s_i}$ has length L (number of time steps). Within each segment, we employ intra-segment attention to model local temporal dependencies:

$$\widetilde{\mathbf{E}}^{(i)} = \text{Attention}(\mathbf{s}_i).$$

Our attention mechanism follows a pre-norm transformer architecture with a gated feed-forward network as introduced by Dubey et al. [17], which enhances gradient flow during training and increases model expressivity. The implementation details are provided in Appendix C. To enable efficient modeling of cross-segment dependencies, we apply a learnable pooling operation that condenses each segment into a discrete token representation:

$$\mathbf{SE}_i = \text{Pool}\big(\widetilde{\mathbf{E}}^{(i)}\big).$$

The resulting segment tokens $\{\widetilde{\mathbf{SE}}_1, \widetilde{\mathbf{SE}}_2, \ldots, \widetilde{\mathbf{SE}}_N\}$ undergo inter-segment attention to capture broader temporal context and long-range dependencies:

$$\widetilde{\mathbf{SE}}_{1:N} = \text{Attention}(\mathbf{SE}_{1:N}),$$

yielding refined segment-level embedding $\widetilde{\mathbf{SE}}_\mathbf{i} \in \mathbb{R}^D$ that integrates contextual information across multiple temporal scales. By reducing the effective sequence length from T to N while preserving both fine-grained temporal dynamics and long-term dependencies, our approach addresses the computational challenges of modeling extended mobility trajectories.

Semantic Context Integration. Prior work such as FPT [81] employs LLMs as general-purpose sequential representation extractors. However, models like those in Wu et al. [67], Nie et al. [48] typically discard semantic embeddings and other critical information. For instance, timestamp attributes (e.g., day of the week, hour of the day) are essential for capturing chronological patterns in

human mobility, while spatial details—such as coordinates—provide additional context for accurate prediction.

While recent works [40, 68, 80] begin to incorporate semantic information in sequential data, they typically employ traditional embedding approaches that fail to holistically capture the rich contextual information inherent in mobility patterns. Our work addresses this limitation by developing a mobility-based semantic embedding method that leverages detailed trajectory information. A key challenge in utilizing LLMs for mobility prediction is balancing information richness with computational efficiency. Unlike approaches such as LLM-Mob [64] that rely on extensive prompting—which can lead to excessive context lengths and computational overhead, our approach breaks trajectory information into smaller, segment-specific pieces that retain essential mobility patterns while ensuring shorter prompts for each component. This decomposition significantly improves computational efficiency without sacrificing semantic richness. For each segment token, we provide informative trajectory descriptions, and for each future timestamp, we provide task descriptions and timestamp information, clarifying the expected output as shown in Appendix D. These prompts are then processed by pretrained LLMs to generate semantic embeddings. We adopt a strategy that uses the special end-of-sequence (<EOS>) token for positional embedding, thereby integrating semantic information without extending the overall context length.

Technically, we define the semantic embedding $\mathbf{TE}_i \in \mathbb{R}^D$ for each token as follows:

$$\mathbf{TE}_i = \mathrm{SelectLast}(\mathrm{LLM}(\mathrm{Prompt}(x_{(i-1)L+1:iL}))).$$

Similarly semantic embedding of task description for future timestamp is defined as $\mathbf{TE}_{\mathcal{T}} = \mathrm{SelectLast}(\mathrm{LLM}(\mathrm{Prompt}(\mathcal{T})))$. Notably, $\mathbf{TE}$ is pre-computed using the LLM, so a runtime forward pass through the language model is not required during training.

Semantic–Temporal Alignment for Mobility Prediction. Since the latent space of the LLM encompasses both temporal tokens and semantic tokens, the semantic embedding can be aligned with the corresponding time span without extending the context length. Consequently, the combined embedding $\mathbf{CE}_i$ for segment i is obtained by elementwise adding the segment embedding $\widetilde{\mathbf{SE}}_i$ and semantic embedding $\mathbf{SE}_i$:

$$\mathbf{CE}_i = \widetilde{\mathbf{SE}}_i + \mathbf{TE}_i.$$

Here, $\mathbf{TE}_i$ serves a role similar to positional embeddings [61], while avoiding the sequence length overhead incurred by prompt concatenation [32]. Similarly, the combined embedding for future timestep $T + j$ is computed as $\mathbf{CE}_{N+j} = \widetilde{\mathbf{E}}_{N+j} + \mathbf{TE}_{\mathcal{T}}$, following the combined embedding for N segments.

After obtaining the enriched embeddings $\mathbf{CE}_i$, we feed them into the backbone of RHYTHM–a pretrained LLM. The LLM processes these embeddings through its deep layers, performing in-context reasoning over the aligned temporal and semantic information, and yields contextualized hidden representations h_i from its last hidden layer.

$$h_i = \mathrm{LLM}(\mathbf{CE}_i).$$

Then we apply an output projection layer to map the LLM's final representations to a set of logits corresponding to candidate locations.

$$P(l_{T+j}|\mathcal{X}, \mathcal{T}) = \mathrm{softmax}(W_o \mathbf{h}_{N+j} + \mathbf{b}_o),$$

where $W_o \in \mathbb{R}^{|\mathcal{L}| \times D}$. These logits are then used to determine the most likely location predictions, thereby generating human mobility forecasting as defined in our problem statement. See Appendix E for implementation details.

3.3 Computational Efficiency

RHYTHM achieves computational and parameter efficiency through several complementary design choices. Semantic embeddings are computed once—offline—using the frozen LLM prior to model training, thereby eliminating any need for language-model inference at runtime. Simultaneously, our temporal tokenization mechanism significantly reduces sequence length from $T + H$ to $N + H$, thereby decreasing the quadratic attention complexity from $\mathcal{O}((T + H)^2)$ to $\mathcal{O}((N + H)^2)$, which is particularly valuable when processing extended mobility histories. Furthermore, we keep the LLM

backbone parameters frozen during training, resulting in faster convergence and reduced memory requirements. The combination of these approaches enables RHYTHM to efficiently process long mobility trajectories while maintaining strong predictive performance (as shown in Figure 3), making it suitable for large-scale mobility prediction tasks where computational resources may be constrained.

3.4 Theoretical Guarantee

We emphasize that our design choices provide strong theoretical guarantees. Employing an LLM as a universal sequential representation extractor provides two key advantages: (1) it ensures the convergence of output values, as demonstrated in Zhou et al. [81, Theorem E.2], and (2) it guarantees a uniform distribution of the feature space in the last hidden layer of the LLM, as outlined in Zhou et al. [81, Theorem E.3]. Together, these properties enable LLMs to enhance the learning capability of the final multi-layer perceptron layer. Additionally, since our model is transformer-based, Ramsauer et al. [54] demonstrates that the transformer architecture is a special case of modern Hopfield networks. Our approach has a guaranteed upper bound on memory retrieval error in LLMs [30, Lemma 3.2]. These theoretical benefits reinforce our method, and our results provide validation.

4 Experiment

In this section, we conduct experiments to demonstrate the performance and efficiency of RHYTHM. We evaluate the performance of RHYTHM on three real-world mobility datasets and compare it with several state-of-the-art baselines. We also conduct a series of ablation studies to investigate the effectiveness of the proposed strategies.

Models. To evaluate the model's performance on mobility prediction, we use multiple pretrained LLMs as the backbone of RHYTHM. These models are obtained from Hugging Face along with their pretrained weights. The specific LLM variants used are detailed in Appendix G.4.

Evaluation Metrics. For mobility prediction, we employ Accuracy@k, where candidate locations are ranked based on model-predicted probabilities, and a prediction is considered correct if the true location is among the top k—and Mean Reciprocal Rank (MRR) to evaluate ranking performance. These metrics have been shown to correlate well with human mobility prediction tasks [23, 19]. We also utilize Dynamic Time Warping (DTW) [47] and BLEU [51] as real-world metrics to evaluate the performance. DTW quantifies their spatial alignment, while BLEU measures the n-gram similarity between predicted and ground-truth trajectories. Detailed descriptions about the evaluation metrics can be found in Appendix G.1.

Datasets. We evaluate our approach on three real-world datasets collected from the cities of Kumamoto, Sapporo, and Hiroshima sourced from YJMob100K [74]. Each day is divided into 48 time slots (each representing 30 minutes), though not every slot contains an observation. Each dataset is divided into training, validation, and test sets based on days, with 70%, 20%, and 10% of the data allocated to each set, respectively. More details about the dataset can be found in Appendix F.

Settings. The temporal resolution is 30 minutes. In our experiment, we use a 7-day lookback window with 336 time slots and set the prediction horizon to 48 time slots (1 day). Also, we set the segment length as 48 time slots for our experiments. The model is trained using cross-entropy loss to maximize prediction accuracy across the target locations.

4.1 Overall Performance

To assess the efficiency of RHYTHM on human mobility prediction, we compare RHYTHM with several state-of-the-art baselines. In this experiment, we evaluate the models on the test datasets using FP16 precision. We conduct each evaluation three times with different random seeds and present the average for each metric.

Baselines. To evaluate the performance of RHYTHM, we compare to LSTM-based models, transformer-based models, and LLM-based models. For the transformer-based models, we conduct experiments with PatchTST [48], PMT [71], ST-MoE-BERT [26], CMHSA [27], iTransformer [39]

and COLA [65]. Among these models, ST-MoE-BERT, PMT and COLA are the state-of-the-art models for human mobility prediction. PatchTST and iTransformer are two powerful transformer-based models for time series forecasting. We add the spatiotemporal embedding to the input of those transformer-based time-series models for fair comparison. For the LLM-based models, we conduct experiments with Time-LLM [32] and Mobility-LLM [23]. Time-LLM is a state-of-the-art model for time series forecasting using LLMs. We also add the spatiotemporal embedding to the input of Time-LLM for fair comparison. Additionally, in order to make a fair comparison, we use the Llama-3.2 1B[1] as the pretrained LLM model for fine-tuning Time-LLM. Mobility-LLM is a versatile LLM-based framework designed for multiple mobility tasks. For the LSTM-based models, we conduct experiments with LSTM [34] and DeepMove [19].

Results. Table 1 show that RHYTHM outperforms the baselines across three datasets in most metrics. On the Sapporo and Hiroshima dataset, RHYTHM achieves the best performance in all evaluation metric. These findings underscore the effectiveness of RHYTHM in mobility prediction tasks. CMHSA and PMT may perform better in Accuracy@3 on Kumamoto due to their specialized attention mechanisms that effectively capture mid-range candidate locations in this region. Despite sharing an LLM-based architecture, Mobility-LLM underperforms compared to RHYTHM, likely because it was primarily designed for visiting intention tasks requiring rich semantic context. In contrast, RHYTHM leverages temporal tokenization and LLM to model multi-scale spatio-temporal dependencies, prioritizing precise location likelihood maximization. This design focus enables RHYTHM to excel in top-rank precision metrics. Overall, RHYTHM achieves a 2.4% improvement in Accuracy@1 and a 1.0% Accuracy@5 respectively compared to the best baseline model.

Table 1: **Performance of RHYTHM and baselines on the Kumamoto, Sapporo, and Hiroshima datasets.** The evaluation metrics include Accuracy@k for different values of k. The reported results are averaged over three runs; variance values are omitted as all are $\leq 2\%$. The best results are highlighted in **bold**, and the second-best results are underlined. RHYTHM demonstrates superior performance compared to baselines across most configurations.

	Kumamoto			Sapporo			Hiroshima		
Model	Acc@1	Acc@3	Acc@5	Acc@1	Acc@3	Acc@5	Acc@1	Acc@3	Acc@5
LSTM	0.2652	0.4799	0.5472	0.2310	0.3940	0.4526	0.2129	0.3775	0.4415
DeepMove	0.2779	0.4986	0.5683	0.2825	0.4672	0.5264	0.2804	0.4810	0.5477
PatchTST	0.2751	0.5018	0.5716	0.2703	0.4582	0.5168	0.2752	0.4839	0.5522
iTransformer	0.2609	0.4724	0.5412	0.2696	0.4500	0.5070	0.2804	0.4857	0.5523
Time-LLM	0.2712	0.4848	0.5535	0.2792	0.4746	0.5352	0.2698	0.4753	0.5426
CMHSA	0.2862	0.5182	0.5887	0.2890	**0.4901**	0.5525	0.2874	0.5001	0.5684
PMT	0.2697	0.4475	0.5187	0.2878	0.4896	0.5522	0.2850	0.4982	0.5668
COLA	0.2864	0.5186	0.5896	0.2847	0.4865	0.5497	0.2874	0.5013	0.5708
ST-MoE-BERT	0.2862	0.5155	0.5871	0.2869	0.4856	0.5480	0.2839	0.4925	0.5601
Mobility-LLM	0.2666	0.4793	0.5448	0.2838	0.4703	0.5288	0.2826	0.4856	0.5525
RHYTHM-Llama-1B	0.2929	0.5200	0.5835	0.2931	0.4876	0.5502	0.2913	0.5027	0.5753
RHYTHM-Gemma-2B	0.2923	0.5191	0.5932	**0.2943**	0.4896	**0.5545**	**0.2953**	**0.5074**	**0.5798**
RHYTHM-Llama-3B	**0.2941**	**0.5205**	**0.5947**	0.2938	0.4875	0.5523	0.2929	0.5032	0.5756

Geographical Evaluation. We evaluate RHYTHM against baseline models using BLEU and DTW, which respectively measure n-gram similarity and spatial alignment error between predicted and ground-truth trajectories. As shown in Table 2, RHYTHM scores the best DTW performance on Sapporo, demonstrating superior spatial alignment. While COLA leads in BLEU scores for all cities, RHYTHM ranks second in Kumamoto. This highlights a key trade-off between exact sequence matching and minimizing spatial deviations. One potential explanation is that COLA employs a post-hoc adjustment technique that recalibrates predictions to better align with the long-tail frequency distribution of locations, which may enhance mid-tier accuracy by mitigating overconfidence in dominant locations. Notably, RHYTHM significantly outperforms LSTM-based methods and transformer baselines by leveraging temporal tokenization and prompt-guided reasoning to enhance sequential coherence and spatial precision. This results in an optimal balance for real-world mobility tasks. For MRR, RHYTHM consistently outperforms all baselines, achieving a 1.44% improvement over

[1] https://huggingface.co/meta-llama/Llama-3.2-1B

the best baseline and demonstrates its superior ranking capability across diverse mobility patterns. Additional experimental results and extended evaluations are reported in Appendix H.

Table 2: **Performance comparison of RHYTHM with baselines using geographical metrics.** The evaluation metrics include DTW (↓), BLEU (↑), and MRR (↑). The best results are highlighted in **bold**, and the second-best results are underlined.

	Kumamoto			Sapporo			Hiroshima		
Model	DTW	BLEU	MRR	DTW	BLEU	MRR	DTW	BLEU	MRR
LSTM	5014	0.1564	0.3860	4507	0.1716	0.3270	5908	0.1544	0.3113
DeepMove	4630	0.1746	0.4021	3818	0.1959	0.3887	4981	0.1933	0.3959
PatchTST	5251	0.1315	0.4021	4099	0.1784	0.3773	5021	0.1884	0.3945
iTransformer	6178	0.1275	0.3796	4074	0.1780	0.3730	5094	0.1789	0.3977
Time-LLM	5984	0.1285	0.3912	3915	0.2145	0.3902	5126	0.1988	0.3872
CMHSA	4490	0.1810	0.4158	3786	0.2299	0.4034	4841	0.2289	0.4086
PMT	4536	0.1524	0.3720	3799	0.2017	0.4026	4851	0.2009	0.4065
COLA	4446	**0.2064**	0.4164	3793	**0.2496**	0.3996	**4840**	**0.2445**	0.4095
ST-MoE-BERT	4691	0.1557	0.4151	3796	0.2102	0.4001	4889	0.2117	0.4031
Mobility-LLM	5603	0.1649	0.3858	3911	0.1917	0.3902	4985	0.2056	0.3990
RHYTHM-Llama-1B	4478	0.1793	0.4216	**3745**	**0.2496**	0.4045	5059	0.2083	0.4069
RHYTHM-Gemma-2B	**4416**	0.1928	0.4205	3995	0.2019	**0.4065**	4857	0.2109	**0.4173**
RHYTHM-Llama-3B	4470	0.1814	**0.4220**	4035	0.1917	0.4048	4935	0.2093	0.4140

Transferability. To demonstrate that RHYTHM transfers well across pretrained LLMs, we vary the size of the pretrained backbone and train it on the mobility prediction datasets (see Table 1 for detailed results). In our experiments, we change the size of the pretrained model in RHYTHM and test them on the mobility prediction datasets. We use the Llama-3.2-1B, Llama-3.2-3B, and Gemma-2-2B model as the pretrained backbone of RHYTHM. The results indicate that the performance of RHYTHM improves as the model size increases. Notably, Llama-3.2-3B and Gemma-2-2B model outperforms the Llama-3.2-1B model in most metrics. This result demonstrates the performance of RHYTHM scales with LLM size and suggests that larger models may achieve even greater performance improvements on larger datasets. Note that our models are pretrained with 30 epochs. It's plausible that Llama-3.2-3B model requires more epochs to fully converge and realize its full performance potential compared with Llama-3.2-1B model. However, Llama-3.2-3B model still achieves competitive performance compared to Llama-3.2-1B model. Overall, Llama-3.2-3B model demonstrated a 0.40% improvement in Acc@1 compared to Llama-3.2-1B model, highlighting the scalability of RHYTHM.

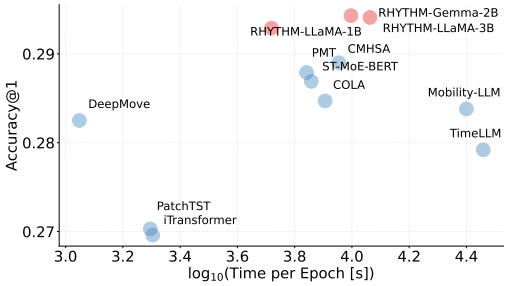

Figure 3: **Training Speed vs. performance of RHYTHM and baseline models on the Sapporo Dataset.**

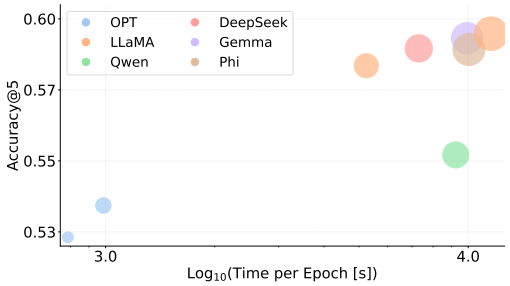

Figure 4: **Efficiency comparison of alternative LLMs, evaluated by the same configuration of Table 4.**

Training Speed. To evaluate the training speed of RHYTHM, we conduct experiments on Sapporo using the same training configuration. We run these experiments on a single NVIDIA A100 GPU with 40GB of memory. The results are shown in Figure 3. RHYTHM reduces the number of trainable parameters to only 12.37% of the full model size, reflecting its parameter-efficient design. In terms of runtime, it achieves a 24.6% reduction in training time compared with the best-performing baseline, while remaining faster than most other models, being 80.6% faster than LLM-based methods

on average. Although RHYTHM is slower than lightweight models such as LSTM, DeepMove, PatchTST, and iTransformer, it substantially outperforms them in accuracy. Moreover, RHYTHM maintains computational efficiency comparable to PMT, COLA and ST-MoE-BERT, despite having significantly higher parameter counts, demonstrating its parameter-efficient design and scalable architecture. Furthermore, RHYTHM's training speed scales predictably with parameter count: Llama-3B is 2.2 times slower than Llama-1B model, while Gemma-2-2B shows a 1.9 times slowdown. A detailed breakdown of preprocessing time, storage cost, and training time across datasets is provided in Appendix I.

Daily and Weekly Trend Analysis. We analyze the periodic accuracy trends of RHYTHM and baselines on Sapporo, measuring performance fluctuations across daily and weekly intervals in Figure 5. RHYTHM demonstrates distinct performance characteristics: achieving 5.0% and 3.4% improvements during weekends and evening peak hours respectively, while showing comparable performance during highly regular periods like nighttime and standard weekday working hours. This pattern reveals a fundamental insight—RHYTHM excels precisely when mobility prediction becomes a complex decision-making task rather than simple pattern matching. During regular hours, mobility is largely deterministic with fixed routines where traditional models' pattern memorization suffices; however, weekends and transitional periods involve nuanced choices influenced by multiple contextual factors, aligning with findings from Barbosa et al. [3] on weekend variability. In these complex scenarios, RHYTHM's hierarchical attention captures both local daily context and global weekly patterns, while the LLM backbone provides reasoning capabilities to model non-routine decision points. This makes RHYTHM particularly valuable for real-world applications where handling irregular, unpredictable periods is crucial for system reliability, even if simpler models suffice for deterministic segments.

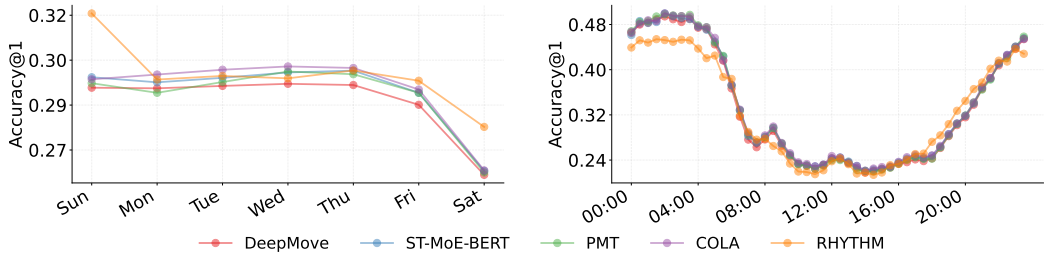

Figure 5: **Weekly (left) and daily (right) accuracy trends of RHYTHM and baselines on Sapporo.** These results illustrate that prediction performance fluctuates over both daily and weekly intervals.

4.2 Method Analysis

In this section, we perform ablation studies to assess the effectiveness of the proposed strategies and test the scaling behavior of RHYTHM.

Ablation study. All experiments utilize Llama-3.2-1B as the backbone model. To evaluate our key components, we conduct ablation studies across three datasets, as shown in Table 3. Removing temporal tokenization significantly degrades performance by 5.39%, while eliminating hierarchical attention (HA) results in a 0.90% decrease, demonstrating that structured temporal encoding is the most critical element of our framework. Regarding semantic enhancement, our findings indicate that both trajectory information and task description prompts contribute substantially to RHYTHM's effectiveness, with their removal causing a combined performance drop of 1.82%. Task descriptions yield marginally higher impact than trajectory information, with their removal causing an additional 0.10% performance decrease compared to omitting trajectory information. Further ablation studies examining the contribution of different design choices are included in Appendix J.

Scaling Behavior. Scalability is a critical factor in the success of large-scale models. We assess RHYTHM's scalability by analyzing its performance in different model sizes. We conduct experiments using pretrained LLMs of varying sizes, including OPT, Llama-3.2, DeepSeek-R1, Gemma-2, Phi-2, and Qwen 2.5 detailed in Table 6. As shown in Table 4, the predictor's prediction performance generally improves as the number of LLM parameters increases. This observation is consistent with

Table 3: **Ablation study on each module in RHYTHM.** We evaluate each module's contribution to overall performance. The best results are highlighted in **bold**. Each module significantly influences RHYTHM's performance across all datasets.

	Kumamoto			Sapporo			Hiroshima		
Model	Acc@1	Acc@3	Acc@5	Acc@1	Acc@3	Acc@5	Acc@1	Acc@3	Acc@5
RHYTHM	**0.2929**	**0.5200**	**0.5835**	**0.2938**	**0.4866**	**0.5502**	**0.2913**	**0.5027**	**0.5753**
w/o HA	0.2917	0.5163	0.5881	0.2901	0.4856	0.5481	0.2895	0.4946	0.5657
w/o token	0.2801	0.5049	0.5764	0.2768	0.4775	0.5409	0.2749	0.4812	0.5535
w/o Traj info.	0.2914	0.5176	0.5891	0.2879	0.4842	0.5472	0.2858	0.4916	0.5633
w/o Task desc.	0.2895	0.5166	0.5889	0.2883	0.4839	0.5463	0.2882	0.4934	0.5648

scaling law dynamics in large models [33]. This scaling behavior highlights the trade-off between predictive performance and adaptation cost. To capture this balance, we assess RHYTHM's scalability across three dimensions: model performance, parameter size, and training and inference speed (time per epoch), as shown in Figure 4. Our results indicate that the largest model, Llama-3.2-3B, achieves the best performance for human mobility prediction, while Llama-3.2-1B remains the most suitable choice in RHYTHM, providing an optimal balance between performance and computational cost.

Table 4: **Scalability Performance on RHYTHM.** We conduct experiments to evaluate the scalability of RHYTHM on Sapporo using pretrained models of varying parameter sizes. The evaluation metrics include Accuracy@k, MRR, training time per epoch (in seconds), and inference time per epoch (in seconds). The best results are highlighted in bold, while the second-best results are underlined. In most configurations, the performance of RHYTHM improves as the model size increases.

Backbone	Training Time (s)	Inference Time (s)	Acc@1	Acc@3	Acc@5	MRR
OPT-125M	787	107	0.2798	0.4726	0.5231	0.3819
OPT-350M	986	224	0.2837	0.4789	0.5343	0.3923
Llama-3.2-1B	5235	359	<u>0.2929</u>	<u>0.5200</u>	0.5835	<u>0.4216</u>
Qwen-2.5-1.5B	9241	336	0.2897	0.4873	0.5521	0.4049
DeepSeek-R1-1.5B	7308	335	0.2921	0.5164	0.5896	0.4188
Gemma-2-2B	9928	559	0.2923	0.5191	<u>0.5932</u>	0.4205
Phi-2	10047	693	0.2915	0.5166	0.5892	0.4183
Llama-3.2-3B	11566	762	**0.2941**	**0.5205**	**0.5948**	**0.4220**

5 Conclusion

This paper proposes RHYTHM, an efficient and scalable framework for mobility prediction. RHYTHM leverages temporal tokenization with hierarchical attention mechanisms to model spatio-temporal dependencies while incorporating semantic embeddings to capture cyclical patterns. The integration of frozen pretrained LLMs as reasoning engines enables RHYTHM to interpret nuanced decision-making processes that influence mobility choices, particularly in scenarios with irregular or non-routine movement patterns, at reduced computational costs. Empirical results demonstrate that RHYTHM significantly outperforms state-of-the-art methods in accuracy. Moreover, its high scalability allows for the seamless integration of different pretrained LLMs in a plug-and-play manner, offering a flexible and efficient prediction framework.

Limitations. It is worth noting that RHYTHM has certain limitations. Its performance depends heavily on the quality of pretrained LLMs, which were designed for language tasks rather than mobility prediction. If these models are resource-constrained, they may fail to capture user mobility patterns accurately. RHYTHM does not adopt an autoregressive prediction strategy; although widely studied in time-series modeling [40], we instead emphasize holistic sequence prediction to capture broader contextual dependencies. Future extensions may integrate autoregressive decoding to more closely mimic step-by-step human mobility decisions. In addition, while freezing pretrained LLMs improves efficiency, RHYTHM's training time remains high, limiting its practicality in some applications. Despite these challenges, RHYTHM provides a novel framework for mobility prediction, advancing efficiency and accuracy. Future work will focus on refining fine-tuning and quantization methods [43, 15, 73] to improve scalability and reduce resource demands.

Acknowledgments and Disclosure of Funding

The authors would like to thank Mingzhen for insightful discussions and the anonymous reviewers for their constructive comments. H.H. and Q.R.W.'s work is supported by the National Science Foundation (NSF) under Grant Nos. 2125326, 2114197, 2228533, and 2402438, as well as by the Northeastern University iSUPER Impact Engine. H.L. is partially supported by the OpenAI Researcher Access Program. This research was supported in part through the computational resources and staff contributions provided by the Quest High Performance Computing facility at Northwestern University, which is jointly supported by the Office of the Provost, the Office for Research, and Northwestern University Information Technology. Any opinions, findings, conclusions, or recommendations expressed in the paper are those of the authors and do not necessarily reflect the views of the funding agencies.

References

[1] Ibrahim M Alabdulmohsin, Behnam Neyshabur, and Xiaohua Zhai. Revisiting neural scaling laws in language and vision. *Advances in Neural Information Processing Systems*, 35:22300–22312, 2022.

[2] Jean-Baptiste Alayrac, Jeff Donahue, Pauline Luc, Antoine Miech, Iain Barr, Yana Hasson, Karel Lenc, Arthur Mensch, Katherine Millican, Malcolm Reynolds, et al. Flamingo: a visual language model for few-shot learning. *Advances in neural information processing systems*, 35: 23716–23736, 2022.

[3] Hugo Barbosa, Marc Barthelemy, Gourab Ghoshal, Charlotte R James, Maxime Lenormand, Thomas Louail, Ronaldo Menezes, José J Ramasco, Filippo Simini, and Marcello Tomasini. Human mobility: Models and applications. *Physics Reports*, 734:1–74, 2018.

[4] Michael Batty. *The new science of cities*. MIT press, 2013.

[5] William Berrios, Gautam Mittal, Tristan Thrush, Douwe Kiela, and Amanpreet Singh. Towards language models that can see: Computer vision through the lens of natural language. *arXiv preprint arXiv:2306.16410*, 2023.

[6] Luís MA Bettencourt, José Lobo, Dirk Helbing, Christian Kühnert, and Geoffrey B West. Growth, innovation, scaling, and the pace of life in cities. *Proceedings of the national academy of sciences*, 104(17):7301–7306, 2007.

[7] Sebastiano Bontorin, Simone Centellegher, Riccardo Gallotti, Luca Pappalardo, Bruno Lepri, and Massimiliano Luca. Mixing individual and collective behaviors to predict out-of-routine mobility. *Proceedings of the National Academy of Sciences*, 122(17):e2414848122, 2025.

[8] Dirk Brockmann, Lars Hufnagel, and Theo Geisel. The scaling laws of human travel. *Nature*, 439(7075):462–465, 2006.

[9] 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.

[10] Oriol Cabanas-Tirapu, Lluís Danús, Esteban Moro, Marta Sales-Pardo, and Roger Guimerà. Human mobility is well described by closed-form gravity-like models learned automatically from data. *Nature Communications*, 16(1):1336, 2025.

[11] Ching Chang, Wei-Yao Wang, Wen-Chih Peng, and Tien-Fu Chen. Llm4ts: Aligning pre-trained llms as data-efficient time-series forecasters. *ACM Transactions on Intelligent Systems and Technology*, 16(3):1–20, 2025.

[12] Eunjoon Cho, Seth A Myers, and Jure Leskovec. Friendship and mobility: user movement in location-based social networks. In *Proceedings of the 17th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 1082–1090, 2011.

[13] Aakanksha Chowdhery, Sharan Narang, Jacob Devlin, Maarten Bosma, Gaurav Mishra, Adam Roberts, Paul Barham, Hyung Won Chung, Charles Sutton, Sebastian Gehrmann, et al. Palm: Scaling language modeling with pathways. *Journal of Machine Learning Research*, 24(240): 1–113, 2023.

[14] Weizhen Dang, Haibo Wang, Shirui Pan, Pei Zhang, Chuan Zhou, Xin Chen, and Jilong Wang. Predicting human mobility via graph convolutional dual-attentive networks. In *Proceedings of the Fifteenth ACM International Conference on Web Search and Data Mining*, pages 192–200, 2022.

[15] Tim Dettmers, Artidoro Pagnoni, Ari Holtzman, and Luke Zettlemoyer. Qlora: Efficient finetuning of quantized llms. *Advances in Neural Information Processing Systems*, 36, 2024.

[16] Qingxiu Dong, Lei Li, Damai Dai, Ce Zheng, Jingyuan Ma, Rui Li, Heming Xia, Jingjing Xu, Zhiyong Wu, Baobao Chang, Xu Sun, Lei Li, and Zhifang Sui. A survey on in-context learning. In Yaser Al-Onaizan, Mohit Bansal, and Yun-Nung Chen, editors, *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing*, pages 1107–1128, Miami, Florida, USA, November 2024. Association for Computational Linguistics.

[17] Abhimanyu Dubey, Abhinav Jauhri, Abhinav Pandey, Abhishek Kadian, Ahmad Al-Dahle, Aiesha Letman, Akhil Mathur, Alan Schelten, Amy Yang, Angela Fan, et al. The llama 3 herd of models. *arXiv preprint arXiv:2407.21783*, 2024.

[18] Nathan Eagle and Alex Pentland. Reality mining: sensing complex social systems. *Personal and ubiquitous computing*, 10:255–268, 2006.

[19] Jie Feng, Yong Li, Chao Zhang, Funing Sun, Fanchao Meng, Ang Guo, and Depeng Jin. Deepmove: Predicting human mobility with attentional recurrent networks. In *Proceedings of the 2018 world wide web conference*, pages 1459–1468, 2018.

[20] Jie Feng, Yuwei Du, Jie Zhao, and Yong Li. AgentMove: A large language model based agentic framework for zero-shot next location prediction. In Luis Chiruzzo, Alan Ritter, and Lu Wang, editors, *Proceedings of the 2025 Conference of the Nations of the Americas Chapter of the Association for Computational Linguistics: Human Language Technologies (Volume 1: Long Papers)*, pages 1322–1338, Albuquerque, New Mexico, April 2025. Association for Computational Linguistics. ISBN 979-8-89176-189-6.

[21] Luciano Floridi and Massimo Chiriatti. Gpt-3: Its nature, scope, limits, and consequences. *Minds and Machines*, 30:681–694, 2020.

[22] Sébastien Gambs, Marc-Olivier Killijian, and Miguel Núñez del Prado Cortez. Next place prediction using mobility markov chains. In *Proceedings of the first workshop on measurement, privacy, and mobility*, pages 1–6, 2012.

[23] Letian Gong, Yan Lin, Xinyue Zhang, Yiwen Lu, Xuedi Han, Yichen Liu, Shengnan Guo, Youfang Lin, and Huaiyu Wan. Mobility-LLM: Learning visiting intentions and travel preference from human mobility data with large language models. In *The Thirty-eighth Annual Conference on Neural Information Processing Systems*, 2024.

[24] Marta C Gonzalez, Cesar A Hidalgo, and Albert-Laszlo Barabasi. Understanding individual human mobility patterns. *nature*, 453(7196):779–782, 2008.

[25] Daya Guo, Dejian Yang, Haowei Zhang, Junxiao Song, Ruoyu Zhang, Runxin Xu, Qihao Zhu, Shirong Ma, Peiyi Wang, Xiao Bi, et al. Deepseek-r1: Incentivizing reasoning capability in llms via reinforcement learning. *arXiv preprint arXiv:2501.12948*, 2025.

[26] Haoyu He, Haozheng Luo, and Qi R Wang. St-moe-bert: A spatial-temporal mixture-of-experts framework for long-term cross-city mobility prediction. In *Proceedings of the 2nd ACM SIGSPATIAL International Workshop on Human Mobility Prediction Challenge*, pages 10–15, 2024.

[27] Ye Hong, Yatao Zhang, Konrad Schindler, and Martin Raubal. Context-aware multi-head self-attentional neural network model for next location prediction. *Transportation Research Part C: Emerging Technologies*, 156:104315, 2023.

[28] Shang-Ling Hsu, Emmanuel Tung, John Krumm, Cyrus Shahabi, and Khurram Shafique. Trajgpt: Controlled synthetic trajectory generation using a multitask transformer-based spatiotemporal model. In *Proceedings of the 32nd ACM International Conference on Advances in Geographic Information Systems*, pages 362–371, 2024.

[29] Edward J. Hu, Yelong Shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, and Weizhu Chen. Lora: Low-rank adaptation of large language models, 2021.

[30] Jerry Yao-Chieh Hu, Maojiang Su, En jui kuo, Zhao Song, and Han Liu. Computational limits of low-rank adaptation (loRA) fine-tuning for transformer models. In *The Thirteenth International Conference on Learning Representations*, 2025.

[31] Allen H Huang, Hui Wang, and Yi Yang. Finbert: A large language model for extracting information from financial text. *Contemporary Accounting Research*, 40(2):806–841, 2023.

[32] Ming Jin, Shiyu Wang, Lintao Ma, Zhixuan Chu, James Y. Zhang, Xiaoming Shi, Pin-Yu Chen, Yuxuan Liang, Yuan-Fang Li, Shirui Pan, and Qingsong Wen. Time-LLM: Time series forecasting by reprogramming large language models. In *The Twelfth International Conference on Learning Representations*, 2024.

[33] Jared Kaplan, Sam McCandlish, Tom Henighan, Tom B Brown, Benjamin Chess, Rewon Child, Scott Gray, Alec Radford, Jeffrey Wu, and Dario Amodei. Scaling laws for neural language models. *arXiv preprint arXiv:2001.08361*, 2020.

[34] Dejiang Kong and Fei Wu. Hst-lstm: A hierarchical spatial-temporal long-short term memory network for location prediction. In *Ijcai*, volume 18, pages 2341–2347, 2018.

[35] Jinhyuk Lee, Wonjin Yoon, Sungdong Kim, Donghyeon Kim, Sunkyu Kim, Chan Ho So, and Jaewoo Kang. Biobert: a pre-trained biomedical language representation model for biomedical text mining. *Bioinformatics*, 36(4):1234–1240, 2020.

[36] Weiyu Li, Qi Wang, Yuanyuan Liu, Mario L Small, and Jianxi Gao. A spatiotemporal decay model of human mobility when facing large-scale crises. *Proceedings of the National Academy of Sciences*, 119(33):e2203042119, 2022.

[37] Qiang Liu, Shu Wu, Liang Wang, and Tieniu Tan. Predicting the next location: A recurrent model with spatial and temporal contexts. In *Proceedings of the AAAI conference on artificial intelligence*, volume 30, 2016.

[38] Yifan Liu, Xishun Liao, Haoxuan Ma, Brian Yueshuai He, Chris Stanford, and Jiaqi Ma. Human mobility modeling with limited information via large language models. *arXiv preprint arXiv:2409.17495*, 2024.

[39] Yong Liu, Tengge Hu, Haoran Zhang, Haixu Wu, Shiyu Wang, Lintao Ma, and Mingsheng Long. itransformer: Inverted transformers are effective for time series forecasting. In *The Twelfth International Conference on Learning Representations*, 2024.

[40] Yong Liu, Guo Qin, Xiangdong Huang, Jianmin Wang, and Mingsheng Long. Autotimes: Autoregressive time series forecasters via large language models. In *The Thirty-eighth Annual Conference on Neural Information Processing Systems*, 2024.

[41] Ilya Loshchilov and Frank Hutter. Decoupled weight decay regularization. In *International Conference on Learning Representations*, 2019.

[42] Massimiliano Luca, Gianni Barlacchi, Bruno Lepri, and Luca Pappalardo. A survey on deep learning for human mobility. *ACM Computing Surveys (CSUR)*, 55(1):1–44, 2021.

[43] Haozheng Luo, Chenghao Qiu, Maojiang Su, Zhihan Zhou, Zoe Mehta, Guo Ye, Jerry Yao-Chieh Hu, and Han Liu. Fast and low-cost genomic foundation models via outlier removal. In *Forty-second International Conference on Machine Learning*, 2025.

[44] Renqian Luo, Liai Sun, Yingce Xia, Tao Qin, Sheng Zhang, Hoifung Poon, and Tie-Yan Liu. Biogpt: generative pre-trained transformer for biomedical text generation and mining. *Briefings in bioinformatics*, 23(6):bbac409, 2022.

[45] Yingtao Luo, Qiang Liu, and Zhaocheng Liu. Stan: Spatio-temporal attention network for next location recommendation. In *Proceedings of the web conference 2021*, pages 2177–2185, 2021.

[46] Soumi Maiti, Yifan Peng, Shukjae Choi, Jee-weon Jung, Xuankai Chang, and Shinji Watanabe. Voxtlm: Unified decoder-only models for consolidating speech recognition, synthesis and speech, text continuation tasks. In *ICASSP 2024-2024 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 13326–13330. IEEE, 2024.

[47] Meinard Müller. Dynamic time warping. *Information retrieval for music and motion*, pages 69–84, 2007.

[48] Yuqi Nie, Nam H Nguyen, Phanwadee Sinthong, and Jayant Kalagnanam. A time series is worth 64 words: Long-term forecasting with transformers. In *The Eleventh International Conference on Learning Representations*, 2023.

[49] Zhenyu Pan, Haozheng Luo, Manling Li, and Han Liu. Conv-coa: Improving open-domain question answering in large language models via conversational chain-of-action. *arXiv preprint arXiv:2405.17822*, 2024.

[50] Zhenyu Pan, Haozheng Luo, Manling Li, and Han Liu. Chain-of-action: Faithful and multimodal question answering through large language models. In *The Thirteenth International Conference on Learning Representations*, 2025.

[51] Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu. Bleu: a method for automatic evaluation of machine translation. In *Proceedings of the 40th annual meeting of the Association for Computational Linguistics*, pages 311–318, 2002.

[52] Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, et al. Pytorch: An imperative style, high-performance deep learning library. *Advances in neural information processing systems*, 32, 2019.

[53] Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, et al. Learning transferable visual models from natural language supervision. In *International conference on machine learning*, pages 8748–8763. PMLR, 2021.

[54] Hubert Ramsauer, Bernhard Schäfl, Johannes Lehner, Philipp Seidl, Michael Widrich, Lukas Gruber, Markus Holzleitner, Thomas Adler, David Kreil, Michael K Kopp, Günter Klambauer, Johannes Brandstetter, and Sepp Hochreiter. Hopfield networks is all you need. In *International Conference on Learning Representations*, 2021.

[55] Xuan Rao, Lisi Chen, Yong Liu, Shuo Shang, Bin Yao, and Peng Han. Graph-flashback network for next location recommendation. In *Proceedings of the 28th ACM SIGKDD conference on knowledge discovery and data mining*, pages 1463–1471, 2022.

[56] Filippo Simini, Marta C González, Amos Maritan, and Albert-László Barabási. A universal model for mobility and migration patterns. *Nature*, 484(7392):96–100, 2012.

[57] Karan Singhal, Shekoofeh Azizi, Tao Tu, S Sara Mahdavi, Jason Wei, Hyung Won Chung, Nathan Scales, Ajay Tanwani, Heather Cole-Lewis, Stephen Pfohl, et al. Large language models encode clinical knowledge. *Nature*, 620(7972):172–180, 2023.

[58] Chaoming Song, Zehui Qu, Nicholas Blumm, and Albert-László Barabási. Limits of predictability in human mobility. *Science*, 327(5968):1018–1021, 2010.

[59] Qwen Team. Qwq-32b: Embracing the power of reinforcement learning, March 2025.

[60] Maria Tsimpoukelli, Jacob L Menick, Serkan Cabi, SM Eslami, Oriol Vinyals, and Felix Hill. Multimodal few-shot learning with frozen language models. *Advances in Neural Information Processing Systems*, 34:200–212, 2021.

[61] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. *Advances in neural information processing systems*, 30, 2017.

[62] Geoff Vigar. *The politics of mobility: transport, the environment, and public policy*. Taylor & Francis, 2002.

[63] Qi Wang, Nolan Edward Phillips, Mario L Small, and Robert J Sampson. Urban mobility and neighborhood isolation in america's 50 largest cities. *Proceedings of the National Academy of Sciences*, 115(30):7735–7740, 2018.

[64] Xinglei Wang, Meng Fang, Zichao Zeng, and Tao Cheng. Where would i go next? large language models as human mobility predictors. *arXiv preprint arXiv:2308.15197*, 2023.

[65] Yu Wang, Tongya Zheng, Yuxuan Liang, Shunyu Liu, and Mingli Song. Cola: Cross-city mobility transformer for human trajectory simulation. In *Proceedings of the ACM on Web Conference 2024*, pages 3509–3520, 2024.

[66] Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, Fei Xia, Ed Chi, Quoc V Le, Denny Zhou, et al. Chain-of-thought prompting elicits reasoning in large language models. *Advances in neural information processing systems*, 35:24824–24837, 2022.

[67] Dennis Wu, Jerry Yao-Chieh Hu, Weijian Li, Bo-Yu Chen, and Han Liu. STanhop: Sparse tandem hopfield model for memory-enhanced time series prediction. In *The Twelfth International Conference on Learning Representations*, 2024.

[68] Haixu Wu, Jiehui Xu, Jianmin Wang, and Mingsheng Long. Autoformer: Decomposition transformers with auto-correlation for long-term series forecasting, 2022.

[69] Shang Wu, Yen-Ju Lu, Haozheng Luo, Maojiang Su, Jerry Yao-Chieh Hu, Jiayi Wang, Jing Liu, Najim Dehak, Jesus Villalba, and Han Liu. SPARQ: Outlier-free speechLM with fast adaptation and robust quantization, 2025.

[70] Shijie Wu, Ozan Irsoy, Steven Lu, Vadim Dabravolski, Mark Dredze, Sebastian Gehrmann, Prabhanjan Kambadur, David Rosenberg, and Gideon Mann. Bloomberggpt: A large language model for finance. *arXiv preprint arXiv:2303.17564*, 2023.

[71] Xinhua Wu, Haoyu He, Yanchao Wang, and Qi Wang. Pretrained mobility transformer: A foundation model for human mobility. *arXiv preprint arXiv:2406.02578*, 2024.

[72] Yongji Wu, Defu Lian, Shuowei Jin, and Enhong Chen. Graph convolutional networks on user mobility heterogeneous graphs for social relationship inference. In *IJCAI*, pages 3898–3904, 2019.

[73] Guangxuan Xiao, Ji Lin, Mickael Seznec, Hao Wu, Julien Demouth, and Song Han. Smoothquant: Accurate and efficient post-training quantization for large language models. In *International Conference on Machine Learning*, pages 38087–38099. PMLR, 2023.

[74] Takahiro Yabe, Kota Tsubouchi, Toru Shimizu, Yoshihide Sekimoto, Kaoru Sezaki, Esteban Moro, and Alex Pentland. Yjmob100k: City-scale and longitudinal dataset of anonymized human mobility trajectories. *Scientific Data*, 11(1):397, 2024.

[75] Dingqi Yang, Daqing Zhang, Vincent W Zheng, and Zhiyong Yu. Modeling user activity preference by leveraging user spatial temporal characteristics in lbsns. *IEEE Transactions on Systems, Man, and Cybernetics: Systems*, 45(1):129–142, 2014.

[76] Dingqi Yang, Benjamin Fankhauser, Paolo Rosso, and Philippe Cudre-Mauroux. Location prediction over sparse user mobility traces using rnns. In *Proceedings of the twenty-ninth international joint conference on artificial intelligence*, pages 2184–2190, 2020.

[77] Song Yang, Jiamou Liu, and Kaiqi Zhao. Getnext: trajectory flow map enhanced transformer for next poi recommendation. In *Proceedings of the 45th International ACM SIGIR Conference on research and development in information retrieval*, pages 1144–1153, 2022.

[78] Wei Zhai, Xueyin Bai, Yu Shi, Yu Han, Zhong-Ren Peng, and Chaolin Gu. Beyond word2vec: An approach for urban functional region extraction and identification by combining place2vec and pois. *Computers, environment and urban systems*, 74:1–12, 2019.

[79] Yunhao Zhang and Junchi Yan. Crossformer: Transformer utilizing cross-dimension dependency for multivariate time series forecasting. In *The eleventh international conference on learning representations*, 2023.

[80] Haoyi Zhou, Shanghang Zhang, Jieqi Peng, Shuai Zhang, Jianxin Li, Hui Xiong, and Wancai Zhang. Informer: Beyond efficient transformer for long sequence time-series forecasting, 2021.

[81] Tian Zhou, Peisong Niu, Xue Wang, Liang Sun, and Rong Jin. One fits all: Power general time series analysis by pretrained LM. In *Thirty-seventh Conference on Neural Information Processing Systems*, 2023.

[82] Zhihan Zhou, Weimin Wu, Jieke Wu, Lizhen Shi, Zhong Wang, and Han Liu. Genomeocean: Efficient foundation model for genome generation, 2025.

NeurIPS Paper Checklist

1. **Claims**

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

   Answer: [Yes]

   Justification: The abstract and introduction clearly state the main claims made in the paper.

   Guidelines:

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

2. **Limitations**

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

   Answer: [Yes]

   Justification: We discuss the limitations in Section 5.

   Guidelines:

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

3. **Theory assumptions and proofs**

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

   Answer: [Yes]

Justification: We provide theory guarantees of RHYTHM in Section 3.4.

Guidelines:

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

4. **Experimental result reproducibility**

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

   Answer: [Yes]

   Justification: We provide all the details necessary to reproduce the main experimental results in the paper. The details of the experiments are included in both the main paper and the supplemental material.

   Guidelines:

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

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

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

Answer: [Yes]

Justification: The code and data, including detailed instructions to reproduce the main experimental results, are publicly available at https://github.com/he-h/rhythm.

Guidelines:

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

6. **Experimental setting/details**

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

Answer: [Yes]

Justification: We introduce the experimental setting in main paper and provide additional details in the supplemental material.

Guidelines:

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

7. **Experiment statistical significance**

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

Answer: [Yes]

Justification: We report the values in all tables and observed stable results with less than 2% variance.

Guidelines:

- The answer NA means that the paper does not include experiments.
- The authors should answer "Yes" if the results are accompanied by error bars, confidence intervals, or statistical significance tests, at least for the experiments that support the main claims of the paper.

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

8. **Experiments compute resources**

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

   Answer: [Yes]

   Justification: We introduce the compute resources in Appendix G.2.

   Guidelines:

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

9. **Code of ethics**

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

   Answer: [Yes]

   Justification: We have carefully reviewed and followed the NeurIPS Code of Ethics throughout our research process.

   Guidelines:

   - The answer NA means that the authors have not reviewed the NeurIPS Code of Ethics.
   - If the authors answer No, they should explain the special circumstances that require a deviation from the Code of Ethics.
   - The authors should make sure to preserve anonymity (e.g., if there is a special consideration due to laws or regulations in their jurisdiction).

10. **Broader impacts**

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

    Answer: [Yes]

    Justification: We discuss the broader impacts in Appendix A.

    Guidelines:

    - The answer NA means that there is no societal impact of the work performed.

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

11. **Safeguards**

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

    Answer: [NA]

    Justification: Our model does not have such a risk of misuse.

    Guidelines:

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

12. **Licenses for existing assets**

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

    Answer: [Yes]

    Justification: We have properly credited all assets used in our research and respected their licenses and terms of use.

    Guidelines:

    - The answer NA means that the paper does not use existing assets.
    - The authors should cite the original paper that produced the code package or dataset.
    - The authors should state which version of the asset is used and, if possible, include a URL.
    - The name of the license (e.g., CC-BY 4.0) should be included for each asset.
    - For scraped data from a particular source (e.g., website), the copyright and terms of service of that source should be provided.

- If assets are released, the license, copyright information, and terms of use in the package should be provided. For popular datasets, `paperswithcode.com/datasets` has curated licenses for some datasets. Their licensing guide can help determine the license of a dataset.
- For existing datasets that are re-packaged, both the original license and the license of the derived asset (if it has changed) should be provided.
- If this information is not available online, the authors are encouraged to reach out to the asset's creators.

13. **New assets**

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

    Answer: [NA]

    Justification: We do not release a new assets in our work.

    Guidelines:

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

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

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

    Answer: [NA]

    Justification: We do not include crowdsourcing experiments or research involving human subjects in this paper.

    Guidelines:

    - The answer NA means that the paper does not involve crowdsourcing nor research with human subjects.
    - Including this information in the supplemental material is fine, but if the main contribution of the paper involves human subjects, then as much detail as possible should be included in the main paper.
    - According to the NeurIPS Code of Ethics, workers involved in data collection, curation, or other labor should be paid at least the minimum wage in the country of the data collector.

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

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

    Answer: [NA]

    Justification: We do not include crowdsourcing experiments or research involving human subjects in this paper.

    Guidelines:

    - The answer NA means that the paper does not involve crowdsourcing nor research with human subjects.

- Depending on the country in which research is conducted, IRB approval (or equivalent) may be required for any human subjects research. If you obtained IRB approval, you should clearly state this in the paper.
- We recognize that the procedures for this may vary significantly between institutions and locations, and we expect authors to adhere to the NeurIPS Code of Ethics and the guidelines for their institution.
- For initial submissions, do not include any information that would break anonymity (if applicable), such as the institution conducting the review.

16. **Declaration of LLM usage**

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

    Answer: [NA]

    Justification: We only use LLMs to assist with writing and formatting the paper.

    Guidelines:

    - The answer NA means that the core method development in this research does not involve LLMs as any important, original, or non-standard components.
    - Please refer to our LLM policy (https://neurips.cc/Conferences/2025/LLM) for what should or should not be described.

Supplementary Material

A Broader Impact **24**

B Extended Related Work **24**

C Attention Implementation Details **26**

D Prompt Design Examples **26**

E Implementation Details **27**

F Dataset **28**

G Experiment Settings **28**
G.1 Evaluation Metrics 28
G.2 Computational Resource 28
G.3 Hyperparameters 28
G.4 LLM variants 29

H Additional Experimental Results **29**
H.1 Autoregressive vs Non-autoregressive Strategy 29
H.2 Deployment efficiency compared to LLM-based baselines 29

I Resource Requirements and Computational Cost **30**
I.1 Dataset Preprocessing 30
I.2 Training Resource Usage 30

J Additional Ablation Studies **30**
J.1 Segment Length Sensitivity 30
J.2 Impact of Pretrained LLMs 31
J.3 Computational Cost of RHYTHM Components 31

A Broader Impact

This paper introduces a new foundation model for human mobility, aiming to improve the reliability and generalization of foundation model applications in spatio-temporal domains. While the work does not have immediate societal implications, it lays the groundwork for future applications in urban planning, public health, disaster response, and transportation. However, the model may inadvertently encode or amplify biases present in the training data, potentially leading to inequitable outcomes in mobility predictions.

B Extended Related Work

Mobility Prediction. Human mobility prediction has evolved from foundational statistical models to advanced deep learning frameworks. Physics-inspired models such as the gravity model [10] and the radiation model [56] predict aggregate population flows using distance and opportunity metrics but lack individual-level detail. To address this, probabilistic approaches like Markov chains [22] and tensor factorization [75] emerge, modeling location transitions at the user level. While these methods improve personalization, they struggle with sparse trajectories and higher-order dependencies inherent in real-world mobility data.

Deep learning introduces sequence-aware architectures like LSTM [37], which capture local temporal contexts, and attention-enhanced variants [19] that address vanishing gradients. However, these models often overlook cyclical patterns. Hybrid approaches like Graph-Flashback [55] and GC-DAN [14] integrate graph structures to model spatial relationships, but their reliance on fixed-length

sequences limits scalability for long-term forecasting. Transformers [61] revolutionize the field with self-attention mechanisms for long-term dependency modeling. Innovations like STAN [45] combine spatial-temporal attention for next-POI recommendation, while COLA [65] extends this to cross-city dynamics. GETNext [77] further refines predictions by disentangling individual preferences from population flows. Despite these advances, Transformer-based methods remain timestamp-centric, incurring quadratic complexity for multi-day sequences and failing to explicitly model hierarchical periodicities (e.g., daily vs. weekly rhythms).

While Transformer-based approaches improve long-term dependency modeling, they still rely on timestamp-centric encoding and struggle with hierarchical periodicities. Recent work explores large language models (LLMs) as an alternative, leveraging their strong generalization capabilities for mobility tasks [23, 38]. Studies like LLM-Mob [64] and AgentMove [20] leverage prompt engineering for next-location prediction and trajectory user linking, while TrajGPT [28] generates synthetic visits via autoregressive decoding. However, these approaches treat mobility sequences as generic token streams, neglecting structured periodic patterns and the modality gap between natural language and spatio-temporal data. Unlike existing LLM-based methods that treat mobility sequences as generic tokens, our approach uses temporal tokenization to explicitly model structured periodicity (daily/weekly cycles), thereby mitigating modality mismatches and capturing multi-scale dependencies for improved long-term mobility prediction.

Time Series Foundation Models. Existing time series foundation models can be divided into two categories: transformer-based models and language-based models. For transformer-based time series models [67, 39, 48], prior studies focus on transformer architecture and self-attention mechanisms to capture temporal dependency in time series data. For instance, PatchTST [48] introduces a patch-based self-attention mechanism to capture long-range dependencies in time series data. STanHop [67] and Crossformer [79] employ hierarchical self-attention to capture temporal dependencies and hierarchical structures in time series data. For language-based time series models [40, 32], prior studies adapt the LLMs to time series data and achieve state-of-the-art performance in time series forecasting tasks. For instance, AutoTime [40] introduces a novel autoregressive structure to capture the temporal dependency in time series data. Time-LLM [32] employs a large language model to capture the complex transitions of time series data. However, these models struggle to capture the inherent complexity of human mobility—with its abrupt location shifts and temporal dynamics-whereas RHYTHM leverages a novel spatio-temporal embedding paired with an autoregressive framework to effectively model these intricate transitions.

Cross-domain Adaptation of LLMs. LLMs have evolved from specialized natural language processing systems into versatile foundation models capable of sophisticated reasoning across diverse tasks [1, 9]. Their transformer-based architecture and extensive pretraining have enabled remarkable transfer capabilities to domains beyond text. In computer vision, models like CLIP [53] align visual and textual representations for zero-shot recognition, while in time series analysis, approaches such as One-Fits-All [81] and LLM4TS [11] demonstrate competitive forecasting through tokenized numerical sequences. In biomedicine, BioBERT [35] and BioGPT [44] demonstrate significant gains on clinical NLP benchmarks, while instruction-tuned models like Med-PaLM approach expert-level medical QA performance [57]. In finance, domain-specific LLMs such as FinBERT [31] and BloombergGPT [70] substantially outperform general-purpose models on sentiment analysis and information extraction.

Rather than computationally expensive full fine-tuning, parameter-efficient adaptation strategies have gained prominence. Low-Rank Adaptation (LoRA) [29] introduces trainable low-rank matrices into attention layers, while more recent approaches keep LLMs entirely frozen by using lightweight adapters that project non-linguistic inputs into the model's embedding space. In vision, prefix-tuning methods [2, 60] train small encoders to produce "prompts" for frozen LLMs, while time series approaches [40, 32] employ projection layers to convert numeric sequences into token embeddings.

Applications of LLMs to human mobility modeling remain limited, with existing approaches relying primarily on parameter-intensive adaptation. Mobility-LLM [23] employs partial fine-tuning, while LLM-Mob [64] leverages in-context learning but lacks structured temporal modeling. In contrast, RHYTHM maintains a fully frozen LLM backbone, preserving the model's pre-trained knowledge while introducing a specialized spatio-temporal framework that efficiently adapts to mobility data characteristics.

C Attention Implementation Details

Our attention mechanism implements a pre-norm transformer block to enhance training stability, with a gated feed-forward network for improved expressivity. The mathematical formulation of our attention block is as follows:

$$Z = \text{LayerNorm}(X) + \text{Multi-Head Attention}(\text{LayerNorm}(X)),$$
$$\widetilde{Z} = Z + \text{GatedFFN}(\text{LayerNorm}(Z)),$$

where X is the input sequence. The multi-head attention operation computes:

$$\text{Multi-Head}(X) = [\text{head}_1 \| \text{head}_2 \| \dots \| \text{head}_h] W_{\text{out}},$$
$$\text{head}_i = \text{Softmax}\left(\frac{X W_{q,i} (X W_{k,i})^\top}{\sqrt{d_k}} \right) X W_{v,i},$$

with h attention heads, where $W_{q,i}, W_{k,i}, W_{v,i} \in \mathbb{R}^{d \times d_k}$ are the query, key, and value projection matrices for the i-th head, and $W_{\text{out}} \in \mathbb{R}^{d \times d}$ is the output projection matrix. The gated feed-forward network incorporates an adaptive gating mechanism:

$$\text{GatedFFN}(Z) = \text{FFN}(Z) \odot \sigma(W_{\text{gate}} Z),$$
$$\text{FFN}(Z) = W_2 \, \text{GELU}(W_1 Z),$$

where σ denotes the sigmoid function, $\odot$ represents element-wise multiplication, and $W_{\text{gate}} \in \mathbb{R}^{d \times d}$ is the learnable gating matrix. The feed-forward network expands the hidden dimension by a factor of 4, with $W_1 \in \mathbb{R}^{4d \times d}$ and $W_2 \in \mathbb{R}^{d \times 4d}$. Dropout is applied after both the attention and feed-forward operations to prevent overfitting.

D Prompt Design Examples

Trajectory Information

```
This is the trajectory of user <User_ID> of day <Day_ID> which is a
<Day_of_Week>.  The trajectory consists of <N> records, each record of
coordinate is as follows:
08:30:  (X=136, Y=42);
09:00:  (X=136, Y=42);
09:30:  (X=137, Y=41);
10:00:  (X=146, Y=37);
10:30:  (X=145, Y=38);
11:00:  (X=144, Y=38);
11:30:  (X=135, Y=41);
12:00:  (X=135, Y=42);
12:30:  (X=135, Y=42);
13:00:  (X=135, Y=42).

Key transitions:  At 10:00:  (X=137, Y=41)→ (X=146, Y=37); At 11:30:  (X=144,
Y=38) → (X=135, Y=41).

Main stay locations:  (X=136, Y=42) from 08:30 to 09:30 (0.5 hours); (X=145,
Y=38) from 10:00 to 11:00 (0.5 hours); (X=135, Y=42) from 11:30 to 13:00 (1.5
hours).
```

Task Description

```
You are a mobility prediction assistant that forecasts human movement
patterns in urban environments.  The city is represented as a 200 x 200
grid of cells, where each cell is identified by coordinates (X,Y). The X
```

```
coordinate increases from left (0) to right (199), and the Y coordinate
increases from top (0) to bottom (199).

TASK: Based on User <User_ID>'s historical movement patterns, predict their
locations for Day <Day_ID> (<Day_of_Week>).  The predictions should capture
expected locations at 30-minute intervals throughout the day (48 time slots).
The model should analyze patterns like frequent locations, typical daily
routines, and time-dependent behaviors to generate accurate predictions of
where this user is likely to be throughout the next day.

The previous days' trajectory data contains information about the user's
typical movement patterns, regular visited locations, transition times,
and duration of stays.  Key patterns to consider include:  home and work
locations, morning and evening routines, lunch-time behaviors, weekend vs.
weekday differences, and recurring visit patterns.
```

E Implementation Details

Algorithm 1 RHYTHM – Overall Pipeline

Require: Trajectory $X = \{(t_i, l_i)\}_{i=1}^{T}$ (timestamps and location IDs); prediction horizon $\{t_{T+1}, \ldots, t_{T+H}\}$; segment length L; frozen LLM

function PRECOMPUTESEMANTICS($X, L, \{t_{T+1}, \ldots, t_{T+H}\}$, LLM)
  $N \leftarrow T/L$
  for $i = 1$ to N **do** ▷ trajectory information
    $TE_i \leftarrow \text{SelectLast}(\text{LLM}(\text{Prompt}_{\text{seg}}(\{X_1, \ldots, X_T\})))$
  end for
  $TE^T \leftarrow \text{SelectLast}(\text{LLM}(\text{Prompt}_{\text{task}}(\{t_{T+1}, \ldots, t_{T+H}\})))$ ▷ task description
  return $\{TE_i\}_{i=1}^{N}$, TE^T
end function

function PREDICT($X, \{t_{T+1}, \ldots, t_{T+H}\}, \{TE_i\}_{i=1}^{N}, TE^T, L$, LLM)
  for $i = 1$ to T **do** ▷ embed time + location into token space
    $E_i^{\text{temporal}} \leftarrow \text{TemporalEmbed}(t_i)$
    $E_i^{\text{spatial}} \leftarrow \text{SpatialEmbed}(l_i)$
    $E_i \leftarrow E_i^{\text{temporal}} + E_i^{\text{spatial}}$
  end for
  Partition $\{E_i\}_{i=1}^{T}$ into $N = T/L$ segments $s_i = \{E_{(i-1)L+1:\, iL}\}$
  for $i = 1$ to N **do** ▷ intra-segment attention, then pool to one token
    $\widetilde{E}^{(i)} \leftarrow \text{IntraAttention}(s_i)$
    $SE_i \leftarrow \text{Pool}(\widetilde{E}^{(i)})$
  end for
  $\{\widetilde{SE}_i\}_{i=1}^{N} \leftarrow \text{InterAttention}(\{SE_i\}_{i=1}^{N})$ ▷ model long-range dependency
  for $i = 1$ to N **do** ▷ additive alignment of semantics at segment level
    $CE_i \leftarrow \widetilde{SE}_i + TE_i$
  end for
  for $j = 1$ to H **do** ▷ future-time tokens + task semantics
    $E_{N+j} \leftarrow \text{TemporalEmbed}(t_{T+j})$
    $CE_{N+j} \leftarrow E_{N+j} + TE^T$
  end for
  $h_{1:N+H} \leftarrow \text{LLM}(\{CE_{1:N+H}\})$ ▷ frozen backbone forward
  $p_{1:H} \leftarrow \text{Softmax}(\text{ProjToClasses}(h_{N+1:N+H}))$
  return $\{\arg\max p_j\}_{j=1}^{H}$
end function

F Dataset

We provide the detail of datasets used in this paper as shown in Table 5.

Table 5: **Dataset Statistics**

City	Users	Duration	Spatial Resolution	Places
Kumamoto	3k	75 days	500m × 500m	40k
Sapporo	17k	75 days	500m × 500m	40k
Hiroshima	22k	75 days	500m × 500m	40k

G Experiment Settings

G.1 Evaluation Metrics

Accuracy@k measures proportion of correct predictions within top-k ranked locations:

$$\text{Accuracy@}k = \frac{1}{H}\sum_{i=1}^{H} \mathbb{1}(l_{T+i} \in \text{top-}k(\hat{p}_{T+i})),$$

where $\mathbb{1}(\cdot)$ is the indicator function and $\hat{p}_{T+i}$ is the predicted probability distribution.

Mean Reciprocal Rank (MRR) evaluates quality of ranked predictions:

$$\text{MRR} = \frac{1}{H}\sum_{i=1}^{H} \frac{1}{\text{rank}(l_{T+i})},$$

where $\text{rank}(l_{T+i})$ is the rank position of the true location.

Dynamic Time Warping (DTW) measures spatial similarity between trajectories:

$$\text{DTW}(\mathcal{Y}, \hat{\mathcal{Y}}) = \min_{\pi} \sum_{(i,j)\in\pi} d(l_{T+i}, \hat{l}_{T+j}),$$

where π is a valid warping path and $d(\cdot,\cdot)$ is the Euclidean distance.

BLEU quantifies n-gram overlap between predicted and ground-truth sequences:

$$\text{BLEU} = BP \cdot \exp\left(\sum_{n=1}^{N} w_n \log p_n\right),$$

where p_n is n-gram precision, w_n is the weight for each n-gram level, and BP is a brevity penalty.

G.2 Computational Resource

We perform all experiments using a single NVIDIA A100 GPU with 40GB of memory and a 24-core Intel(R) Xeon(R) Gold 6338 CPU operating at 2.00GHz. Our code is developed in PyTorch [52] and utilizes the Hugging Face Transformer Library[2] for experimental execution.

G.3 Hyperparameters

We present the hyperparameters used in the training stage for each model. Embeddings for time-of-day and day-of-week, the categorical location embedding, and the coordinate projection all use hidden dimensions of 128, 128, 256, and 128, respectively. We use **AdamW** [41] as the optimizer. For model training, we conduct a systematic hyperparameter search, exploring learning rates from the set $\{1 \times 10^{-4}, 3 \times 10^{-4}, 5 \times 10^{-4}\}$ and weight decay values from $\{0, 0.001, 0.01\}$. Through extensive validation experiments, we determine the optimal configuration for each dataset. All models are trained with a consistent batch size of 64 across all datasets for fair comparison. The final hyperparameter settings are selected based on performance on the validation set.

[2] https://huggingface.co/docs/transformers

G.4 LLM variants

Our experiments deployed multiple foundation language models as text embedders and frozen backbones within RHYTHM to evaluate cross-scale performance. Table 6 presents the pre-trained models accessed through the Hugging Face Transformers library, ranging from 125M to 3B parameters.

Table 6: **Pre-trained language models employed as backbones in RHYTHM.**

Model	Parameters	HuggingFace Repository
OPT-125M	125M	facebook/opt-125m
OPT-350M	350M	facebook/opt-350m
Llama-3.2-1B	1.24B	meta-llama/Llama-3.2-1B
Qwen-2.5-1.5B	1.54B	Qwen/Qwen2.5-1.5B
DeepSeek-R1-1.5B	1.78B	deepseek-ai/DeepSeek-R1-Distill-Qwen-1.5B
Gemma-2-2B	2.61B	google/gemma-2-2b-it
Phi-2	2.78B	microsoft/phi-2
Llama-3.2-3B	3.21B	meta-llama/Llama-3.2-3B

H Additional Experimental Results

H.1 Autoregressive vs Non-autoregressive Strategy

Table 7 compares RHYTHM under autoregressive and non-autoregressive strategies. The non-autoregressive approach delivers comparable accuracy while being over two orders of magnitude faster.

Table 7: **Comparison of RHYTHM with autoregressive and non-autoregressive prediction strategies.** Results are reported using Acc@1, Acc@3, Acc@5, and computational time per iteration (s/iter). The best results are highlighted in **bold**.

Model	Time (s/iter)	Acc@1	Acc@3	Acc@5
Non-autoregressive	**0.96**	**0.2929**	0.5200	**0.5835**
Autoregressive	39.80	0.2884	**0.5247**	0.5801

H.2 Deployment efficiency compared to LLM-based baselines

Table 8 reports deployment efficiency for RHYTHM compared with LLM-based baselines, evaluated under both GPU and CPU inference. On GPU, RHYTHM requires substantially less memory and achieves lower latency than TimeLLM, while remaining competitive with Mobility-LLM. On CPU, RHYTHM also demonstrates favorable latency and moderate RAM usage, underscoring its practicality for deployment in resource-constrained environments.

Table 8: **Deployment efficiency of RHYTHM, TimeLLM, and Mobility-LLM. Deployment efficiency of RHYTHM, TimeLLM, and Mobility-LLM.** We report model size, GPU memory footprint (GRAM), inference latency on GPU and CPU, and RAM usage.

		GPU		CPU	
Model	Size (MB)	GRAM (MB)	Latency (ms)	RAM (MB)	Latency (ms)
RHYTHM	5841.9	5741.4	261.6 ± 0.8	12497.9	39.5 ± 0.9
TimeLLM	3762.7	11213.1	392.7 ± 9.5	18677.8	64.2 ± 2.5
Mobility-LLM	4880.7	9872.8	192.1 ± 5.2	10552.2	32.3 ± 1.9

I Resource Requirements and Computational Cost

In this section, we detail RHYTHM's resource requirements to provide a practical perspective on the computational cost of deploying RHYTHM.

I.1 Dataset Preprocessing

Time and storage costs of semantic embedding generation are detailed in Table 9.

Table 9: **Preprocessing cost of RHYTHM across datasets.**

Dataset	Preprocessing Time (h)	Storage Size (GB)
Kumamoto	3.1	1.6
Sapporo	15.9	9.1
Hiroshima	20.1	11.8

I.2 Training Resource Usage

Training requirements with varying sequence lengths are summarized in Table 10, including GPU memory (GRAM) and runtime per epoch. The results show that memory consumption and training time do not increase linearly with sequence length, demonstrating that RHYTHM can handle long sequences at moderate additional cost due to its temporal tokenization design.

Table 10: **Training cost with varying sequence lengths.**

Sequence Length (T)	GRAM (MB)	Time/Epoch (min)
48	7126.3	23.1
168	7469.5	24.6
336	8202.0	26.5
672	9826.4	30.9

J Additional Ablation Studies

J.1 Segment Length Sensitivity

We analyze the impact of varying temporal segment length L on prediction accuracy and efficiency in Table 11. Smaller segments (e.g., $L = 1$, 30 minutes) capture fine-grained details but result in excessive fragmentation and substantially higher computation time. In contrast, very large segments (e.g., $L = 96$, 2 days) reduce runtime but blur meaningful daily mobility boundaries, leading to degraded accuracy. Daily segmentation ($L = 48$) achieves the best balance, yielding the highest predictive performance while keeping iteration time under 1 second. These findings validate our choice of $L = 48$ as a balanced temporal unit for mobility modeling.

Table 11: **Effect of varying segment length L on prediction accuracy and runtime.** The best results are highlighted in **bold**.

Segment Length (L)	Segments (N)	Acc@1	Acc@3	Acc@5	Time (s/iter)
1 (30 min)	336	0.2801	0.5049	0.5764	6.59
24 (12 hr)	14	0.2883	0.5124	0.5792	1.10
48 (1 day)	7	**0.2929**	**0.5200**	**0.5835**	0.96
96 (2 days)	3	0.2851	0.5087	0.5743	0.93

J.2 Impact of Pretrained LLMs

RHYTHM benefits substantially from pretraining: the pretrained LLM variant achieves the strongest accuracy, while randomly initialized and LLM-free variants lag behind as shown in Table 12.

Table 12: **Comparison of RHYTHM with pretrained, randomly initialized, and no-LLM variants on Kumamoto.** Pretraining provides the best accuracy across all metrics.

RHYTHM Variant	Acc@1	Acc@3	Acc@5
w/ pretrained LLM (ours)	**0.2929**	**0.5200**	**0.5835**
w/ randomly initialized LLM (frozen)	0.2556	0.4842	0.5313
w/o LLM (only attention)	0.2749	0.4921	0.5623

J.3 Computational Cost of RHYTHM Components

Table 13 shows how different architectural choices affect model size and training time. Temporal tokenization and hierarchical attention both contribute to reducing runtime without significantly increasing parameter count, highlighting their role in RHYTHM's efficient design.

Table 13: **Trainable parameters (absolute and relative) and training time per epoch for different architectural configurations of RHYTHM on Kumamoto.**

Configuration	Trainable Params (MB)	Share of Total (%)	Time/Epoch (min)
Full RHYTHM (frozen LLM)	152	12.37	31
Unfrozen LLM w/ LoRA	200	16.27	102
w/o Temporal Tokenization	148	12.00	53
Only Attention Module (no LLM)	152	12.37	16
w/o HA	39	3.25	20