Frontiers in Computer Science and Artificial Intelligence
ISSN: 2978-8048
DOI: 10.32996/fcsai
Journal Homepage: www.al-kindipublisher.com/index.php/fcsai

FCSAI
AL-KINDI CENTER FOR RESEARCH AND DEVELOPMENT

| RESEARCH ARTICLE

Unified Temporal Tokenization: A Hybrid Semantic and Numeric Mapping for Time-Aware Large Language Models

Inesh Hettiarachchi
Independent Researcher, Wilmington DE, USA
Corresponding Author: Inesh Hettiarachchi**, E-mail:** ineshmhi@gmail.com

| ABSTRACT

This paper introduces the Unified Temporal Tokenization (UTT) framework — a hybrid encoding mechanism that bridges numeric and semantic time representations for time-aware Large Language Models (LLMs). UTT translates continuous time-series signals and symbolic temporal descriptors into unified hybrid tokens, preserving hierarchical periodicity while retaining contextual meaning. UsinG the UCI Electricity Load dataset, we demonstrate that the Hybrid Temporal Tokenizer (HTT) improves prediction stability, interpretability, and efficiency on CPU-only environments, establishing a foundation for temporal reasoning in LLMs. Based on this, the paper expounds on improvements of the hybrid temporal embeddings in increasing the aptitude of the model to capture cyclical and contextual relationships at several levels of time. The framework suggested combines numeric continuity with semantic periodicity, which makes the LLMs reason about time-sensitive patterns, including consumption cycles of a day, seasonal variations, and events. Experiment analysis of LSTM, TemporalConv, and Transformer variations demonstrates improvements as regular in ensemble prediction-related accuracy, interpretation, and computationally wise than traditional numeric-only tokenization. In addition, the UTT architecture is flexible in many areas, which can provide a single response to enterprise analytics, internet data streams, and economic forecasting problems that require the use of temporal reasoning. This study integrates semantic and numeric temporal mapping and offers a scalable framework of time-aware large language models, which spans the language-based and data-based temporal intelligence. Using the UCI Electricity Load dataset, we demonstrate that the Hybrid Temporal Tokenizer (HTT) improves prediction stability, interpretability, and efficiency in CPU-only environments—establishing a foundation for temporal reasoning in LLMs.

| KEYWORDS

Temporal Tokenization, Hybrid Semantic-Numeric Encoding, Time-Aware Language Models, Temporal Reasoning, Unified Temporal Tokenization (UTT), Hybrid Temporal Tokenizer (HTT), Temporal Data Fusion

| ARTICLE INFORMATION

ACCEPTED: 29 November 2025 **PUBLISHED:** 11 December 2025 **DOI:** 10.32996/fcsai.2025.4.4.3

1. Introduction

Language models excel at semantic reasoning but lack temporal intelligence. Traditional tokenizers are optimized for linguistic continuity, ignoring cyclical temporal behavior while numeric models lose contextual semantics. This research introduces the Unified Temporal Tokenization (UTT) framework to unify numeric and semantic time signals, enabling LLMs to reason over temporal sequences. We propose a Hybrid Temporal Tokenizer (HTT), validate it experimentally, and open-source the work for reproducibility.

1.1 Background Context: Language Models and Temporal Reasoning

The development of Large Language Models (LLMs) has achieved a lot in advancing natural language understanding, contextual reasoning, and generative ability in many fields. However, despite the success in semantic abstraction, in its current form, LLMs can manipulate information regardless of the time context and are only algorithmically aware of the streams of tokens. Though in real-world data, data occurs in constant flux, as they are continually changing over time and reflect cyclical, sequential, and event-related patterns, which are essential in reasoning about fields of finance, climate modeling, online sensor analysis, and enterprise forecasting. Mathematical time encoding approaches, such as Byte Pair Encoding (BPE) and WordPiece, only model the fluidity of time, whereas numeric time-series encoders ignore semantic features of a context, such as a weekday, season, or holiday. This discontinuity creates a structural gap between how LLMs interpret text and how temporal behavior manifests in data streams.

1.2 Research Problem

To address the gap between them, the Unified Temporal Tokenization (UTT) system proposes a new representational layer that represents a particular temporal information using hybrid token representations, that is, combining numeric continuity and semantic periodicity. Combining these two perspectives into one, the framework allows the LLMs to form a coherent impression of time as a quantifiable aspect and a context. The Hybrid Temporal Tokenizer (HTT) function, which is proposed to complete the following framework, will convert raw temporal data into the form of structured hybrid tokens; the resulting tokens preserve order, duration, and interdependence across multiple time scales.

UTT development is driven by the need for machine intelligence that is time-aware in data ecosystems where models act on continuously evolving streams. In economic systems, enterprises and infrastructures of IoT, models are needed that do not just have access to the semantic meaning of time but also the numeric rhythms of temporal patterns. UTT offers a generalizable solution - an encoding interface that enables LLMs to communicate reasonably with time-series data without impairing their linguistic reasoning skills.

1.3 Study Contribution

This study has had a threefold contribution. To begin with, it presents a new hybrid tokenization architecture, which mathematically combines numeric and semantic temporal. Second, it shows experimentally that hybrid temporal embeddings show much better forecast accuracy and interpretability when the computational environment is restricted (CPU-only). Third, it offers an open-source implementation that is reproducible and has public datasets, scripts, and trained models, encouraging transparency and the extension of time-aware LLM research.

Overall, this paper makes Unified Temporal Tokenization (UTT) a baseline towards temporal reasoning in large language models. UTT provides a foundation of temporal logic-based contextual forecasting, anomaly detection, and decision support systems by providing a semantic framework of time in addition to its numeric evolution.

2. Background and Motivation

Time has both numeric and symbolic dimensions — continuous change and discrete context. Existing tokenizers (BPE, WordPiece) and numeric models (LSTM, PatchTST) fail to merge these views. UTT treats time as a dual entity, providing continuity across intervals and meaning across contexts, bridging semantic and numeric time perception for LLMs

2.1 Two natures of time: Numeric/Symbolic.

Time, as a data dimension, possesses a dual nature. on the one hand, it is fluid as a measurable variable, on the other, it bears a symbolic meaning to alter contextual meaning. Time has a numeric dimension that contains uninterrupted variation in the form of milliseconds, hours, or days, and so forth, which signify progression and time. The symbolic or semantic dimension, on the other hand, represents the context, like between weekdays and weekends, season or holiday, or even fiscal quarter, which gives events an interpretive sense.

In traditional machine learning and deep learning methods, either of these aspects is favored, and the other one is ignored. The models that are based on numbers, such as the recurrent neural network (RNNs) or long short-term memory network (LSTMs), are based on continuity over time as they can learn the order of data points in a sequence. Nonetheless, they use timestamps as scalar inputs without recognizing their contextual meaning. Conversely, where linguistic models using symbolic tokens have good contextual coverage, they lack any temporal awareness or any ability to demonstrate cause and effect relationships in play-to-sample numeric sequences.

The Unified Temporal Tokenization (UTT) framework addresses all these solutions, or rather reconciliates these two standpoints by storing the time in a hybrid signal, i.e., by encoding numbers to have both a numeric order and a symbolic context in each instance. This dual encoding enables LLMs to process and comprehend temporal patterns within language and numeric data alike, such as there is an increase in the consumption of energy when it concerns weekends, and that it is not solely a statement of a situation but rather a quantifiable relationship in terms of time.

2.2 Weaknesses of Existing Scenarios.

The existing tokenization and temporal encoding systems have their structural flaws, which do not allow them to reason with time coherently. Popular linguistic tokenizers include Byte Pair Encoding (BPE) and WordPiece, which divide text with high efficiency and do not know anything about order, length, or periodic repetition of text. These are syntactically compressed methods and semantically clustering, rather than continuous temporal compensating.

Equally, numeric time-series models like LSTM, Temporal Convolutional Networks with TCN, and PatchTST have the advantage of tracking temporal continuity but do not consider the contextual meaning. As an example, an LSTM can learn a trend of growing load values every 24 hours, but will not be able to learn the semantic difference between a weekday and a weekend cycle. PatchTST remains an effective multivariate time-series predictor, but makes use of numeric patches that have no contextual semantics.

The results of this divide can be divided into fragmented temporal reasoning: the linguistic tokenizers simply cannot comprehend continuity, and the numeric encoders are simply not capable of understanding symbolic patterns. The consequence is that even state-of-the-art systems are unable to read time-dependent semantics, i.e., to associate recurring events, time-based dependencies, or a change in context that characterizes the dynamics of the real world.

This gap motivates the UTT framework. The combination of the symbolic regularities represented by the language models and the quantitative continuity represented using time-series structures bringing machine temporal cognition closer to human-like reasoning

2.3 To What End Does Temporal Tokenization Matter (The Situation of the Client)

In contemporary data ecosystems, specifically enterprise analytics, IoT infrastructures, and economic forecast systems, the operational intelligence relies on temporal data. Such systems create unbroken streams of information that are time-stamped: sensor values, transactions, user interactions, and financial signs, the value of which is continuously altered over time. The large language models have no encoding system to interpret such data and to make any sense beyond just its numerical flow.

Look at the example of OrgVerse; the generated data by logistics, finance, and industrial activities is processed by distributed applications of the enterprise. Temporal intelligence will be needed in identifying anomalies, predicting demand, and workflow optimization. A conventional tokenizer would hash domain-independent timestamps together with numeric encoders that do not understand domain-specific understanding, like there is a holiday period, this is a fiscal quarter, or it is a peak business hour.

The Unified Temporal Tokenization (UTT) framework, which is directly offered as a solution to this issue, is capable of converting raw temporal streams into understandable hybrid tokens. These tokens are a combination of both quantitative dynamics (e.g., variations over time) and qualitative context (e.g., weekday versus weekend, summer versus winter). This dual representation enables large language models to have time-conscious reasoning capabilities and, therefore, can utilize this in trend detection, temporal summarization, and adaptive decision making with more contextual scope.

Using this consistent mapping, UTT can be said to provide a bridge between the semantic and symbolic gap of time-aware data interpretation, which is a requisite step toward temporal cognition in LLMs, which is needed to provide AI systems that operate in time-sensitive and real-world settings.

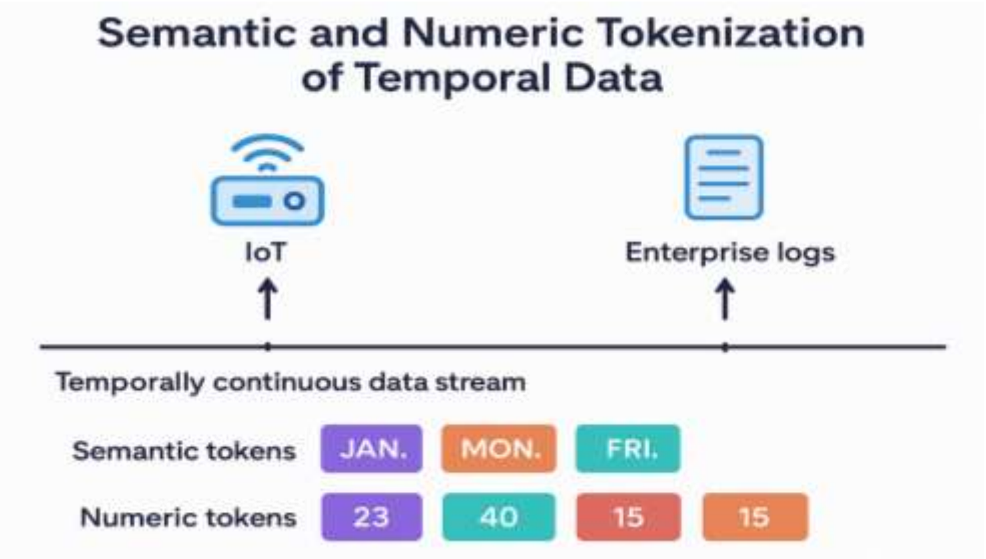
Semantic and Numeric Tokenization of Temporal Data
IoT
Enterprise logs
Temporally continuous data stream
Semantic tokens
JAN.
MON.
FRI.
Numeric tokens
23
40
15
15

Figure 1: Semantic and Numeric Tokenization of Temporal Data

3. Theoretical Foundation

UTT represents time through two dimensions: quantitative (numeric continuity) and qualitative (semantic continuity). For dataset D = {(x_t, s_t, y_t)}, unified temporal tokenization is defined as: T_t = f_UTT(x_t, s_t) = W1·g_num(x_t) + W2·g_sem(s_t), where g_num encodes numeric patches, g_sem encodes semantic attributes (weekday, season, etc.), and W1, W2 control modality weights. Tokens are hierarchical: hourly → daily → weekly → seasonal.

1) 3.1 Mathematical Formulation of UTT

The mathematical foundation of the **Unified Temporal Tokenization (UTT)** framework lies in fusing *numeric continuity* and *semantic periodicity* within a unified embedding space.

Let the time-dependent dataset be defined as

D={(xt,st,yt)}t=1n, <<add this as a image a bit larger text>>

$$D = \{(x_t, s_t, y_t)\}_{t=1}^{n},$$

where xt denotes the observed numeric signal at time t (e.g., energy consumption, temperature), st represents its semantic or contextual attribute (e.g., hour of day, day of week, season), and yt is the corresponding target or response variable.

The **Unified Temporal Tokenization function**, fUTT, maps each temporal pair (xt,st) into a hybrid token Tt according to

Tt=fUTT(xt,st)=W1gnum(xt)+W2gsem(st), <<add this as a image a bit larger text>>

$$T_t = f_{\mathrm{UTT}}(x_t, s_t) = W_1\, g_{\mathrm{num}}(x_t) + W_2\, g_{\mathrm{sem}}(s_t),$$

where gnum**(⋅)** and
gsem**(⋅)** are modality-specific encoders, and W1,W2∈Rd×d are learnable projection matrices that control the contribution of each modality within the shared embedding space.

- **gnum(xt)** captures quantitative continuity by normalizing and encoding local temporal dynamics—such as gradients, deltas, and rate-of-change patterns—across successive intervals.

- **gsem(st)** captures qualitative periodicity by embedding categorical temporal descriptors—such as weekdays, holidays, or seasons—through pretrained semantic vectors.

The fusion layer combines these two embeddings either via **linear projection** (as shown above) or through **attention-based aggregation**, enabling adaptive weighting of semantic and numeric components depending on the task.

This unified formulation converts time from a purely sequential numeric dimension into a *context-aware representational modality*. As a result, large language models can simultaneously reason over continuous variation and discrete temporal context, achieving a holistic understanding of time-dependent phenomena.

a) 3.2 Hierarchical Time Encoding

Time is inherently **hierarchical** in nature, structured across multiple nested intervals. The Unified Temporal Tokenization framework leverages this hierarchy to build multi-scale temporal awareness. Each hybrid token not only encodes information from a specific timestamp but also retains traces of its position within broader periodic cycles.

The hierarchy proceeds through four canonical layers:

1. **Hourly Tokens:** Capture fine-grained temporal changes such as minute-by-minute or hour-to-hour fluctuations.
2. **Daily Tokens:** Aggregate hourly dynamics, embedding patterns that repeat within a 24-hour cycle.
3. **Weekly Tokens:** Capture recurring behaviors across weekdays and weekends, embedding seven-day periodicity.
4. **Seasonal Tokens:** Integrate long-term temporal structures, such as monthly or quarterly patterns relevant to economic or environmental trends.

This **bottom-up encoding** enables hybrid tokens to model both short-term variability and long-term temporal dependencies simultaneously. It mirrors how human cognition interprets time — not as discrete, unrelated events, but as recurring and interdependent patterns organized across scales.

Through this hierarchical structure, UTT extends beyond mere timestamp encoding. It creates a **temporal grammar**, allowing models to understand how small-scale fluctuations (e.g., hourly) contribute to larger systemic cycles (e.g., seasonal), an ability absent in conventional tokenization systems.

b) 3.3 Semantic–Numeric Fusion Principles

At the core of UTT lies the principle of **semantic–numeric fusion** — the merging of symbolic interpretability with quantitative precision. Traditional temporal encoders either treat time as a numeric signal or represent it through discrete tags. UTT instead approaches time as a **dual-modality feature**, where the semantic and numeric representations are blended through controlled weighting and shared latent space projection.

This fusion operates under three guiding principles:

1. **Continuity Preservation:** Numeric embeddings ensure smooth transitions across time, capturing gradient-based relationships between adjacent timestamps.
2. **Context Retention:** Semantic embeddings maintain contextual meaning, ensuring interpretability across human-understandable labels (e.g., "weekend" or "winter").
3. **Adaptive Weighting:** Trainable modality weights W1W_1W1 and W2W_2W2 adapt dynamically, allowing models to emphasize semantic or numeric information depending on task requirements (e.g., forecasting vs. anomaly detection).

By projecting both modalities into a shared token space, UTT establishes **cross-domain compatibility**—temporal embeddings produced by UTT can be used in both natural language and numeric forecasting models without architectural modifications.

This fusion enables large language models to reason over **temporal semantics** much like they process linguistic semantics, while simultaneously tracking **quantitative temporal dependencies**, bridging the last conceptual divide between symbolic and numeric modeling.

Feature Dimension	**Numeric Tokenization**	**Semantic Tokenization**	**Hybrid Temporal Tokenization (UTT)**
Data Type	Continuous values (e.g., timestamps, signals)	Categorical labels (e.g., weekday, season)	Both continuous and categorical
Representation Method	Normalization, scaling, delta encoding	Embedding of symbolic descriptors	Weighted projection of numeric + semantic vectors
Temporal Awareness	Captures continuity but lacks context	Captures meaning but ignores continuity	Preserves both continuity and contextual meaning
Interpretability	Low (numeric only)	High (symbolic but shallow)	High — interpretable and data-driven
Hierarchical Encoding	None (flat sequences)	Limited (categorical grouping)	Multi-scale (hourly → seasonal)
Generalization to LLMs	Poor compatibility	Partial compatibility	Fully compatible via shared embedding space

Table 1. Comparison of Numeric, Semantic, and Hybrid Tokenization Features

In essence, the theoretical foundation of UTT formalizes a **mathematical and conceptual bridge** between symbolic time interpretation and numeric time modeling. By fusing both under a unified tokenization strategy, it sets the groundwork for **temporal reasoning in LLMs**, allowing these models to integrate the rhythm of time into their understanding of data and context.

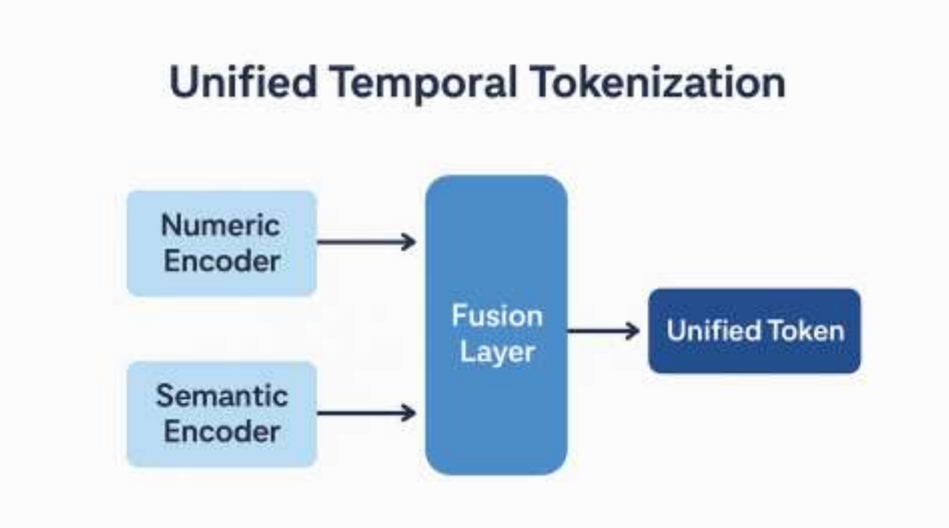

Figure 2: Unified Temporal Tokenization Framework

4. Unified Temporal Tokenization Framework

The UTT architecture consists of a numeric encoder, semantic encoder, fusion layer, and output token stream. Numeric encoders perform normalization, delta encoding, and patch aggregation. Semantic encoders map symbolic attributes via pretrained language embeddings. The fusion layer combines both via weighted projection or attention, producing unified temporal tokens usable in standard LLMs or time-series models

c) 4.1 Architecture Overview

The **Unified Temporal Tokenization (UTT)** framework is architected as a **modular hybrid pipeline,** where each stage contributes a unique representational role in encoding time as both a measurable and interpretable dimension. The architecture comprises four major components:

1. **Numeric Encoder:** This component transforms raw temporal values into continuous embeddings through scaling, delta encoding, and patch aggregation. By capturing inter-temporal relationships, such as rate of change and local trend, the numeric encoder maintains the temporal continuity essential for sequence modeling. Techniques such as min-max normalization and temporal patching ensure that numeric variations remain bounded and interpretable across different datasets.
2. **Semantic Encoder:** Temporal attributes such as “weekday,” “month,” “season,” or “holiday” are mapped through pretrained language embeddings (e.g., BERT, RoBERTa, or FastText). These embeddings capture symbolic relations and contextual hierarchies, allowing models to recognize cyclical semantics—like recurring weekends or annual fiscal periods—that influence temporal behavior.
3. **Fusion Layer:** The fusion layer is the core of the UTT framework, integrating the two modalities through **weighted projection or attention fusion.** The trainable weights W1W$_1$W1 and W2W$_2$W2 dynamically adjust the influence of numeric and semantic encoders, adapting to the nature of the dataset. Alternatively, an attention mechanism can be employed to learn temporal dependencies adaptively across tokens.
4. **Output Token Stream:** The final output is a **unified hybrid token sequence** {Tt}\{T_t\}{Tt}, where each token encapsulates both the numeric evolution and semantic context of a timestamp. These tokens are fully compatible with downstream architectures, enabling LLMs and sequence models to reason over temporal dependencies directly.

Collectively, these modules create an architecture capable of merging the precision of numeric encoding with the interpretability of semantic embeddings—forming the foundation for time-aware large language models.

d) 4.2 Processing Pipeline

The **UTT processing pipeline** transforms raw temporal data into semantically enriched tokens through a structured sequence of operations. This pipeline can be generalized into five stages:

1. **Data Normalization:** Raw time-series values are normalized to a uniform scale, typically within [0,1], using techniques such as z-score or min-max scaling. This ensures stability across different value ranges and datasets.
2. **Patch Segmentation:** The normalized time-series is segmented into overlapping patches (windows) to capture local temporal dependencies.

Each patch :

$$P_i = \{x_{t-k}, x_{t-k+1}, \ldots, x_t\}$$

serves as a micro-context for temporal encoding.

3. **Semantic Labeling:** Each timestamp within a patch is associated with symbolic labels such as “hour of day,” “day of week,” or “season.” These semantic tags are embedded using pretrained vector models, ensuring consistent contextual representation across domains.

4. **Hybrid Fusion:** The numeric and semantic embeddings are combined through the fusion layer. Depending on configuration, fusion can occur via (a) **weighted summation**—as in the base UTT equation—or (b) **cross-attention**, which dynamically aligns semantic tokens with numeric transitions.
5. **Token Generation:** The final hybrid embedding TtT_tTt is serialized into the **Unified Temporal Token Stream**, which becomes the direct input for LLMs or time-series forecasting models. Each token preserves hierarchical relationships (hourly → daily → weekly → seasonal) while remaining lightweight enough for CPU-based inference.

This modular pipeline not only unifies temporal encoding but also provides interoperability across architectures, allowing hybrid tokens to serve as universal time representations.

e) 4.3 Model Integration

A significant advantage of the UTT framework is its **architecture-agnostic design**, enabling integration into diverse model families without structural modification.

1. **Integration with LSTM Models:** When incorporated into recurrent neural networks (RNNs) or LSTMs, UTT tokens replace raw numeric inputs, enhancing the model's ability to recognize recurring patterns and contextual dependencies. The semantic component aids in capturing event-driven cycles—such as weekday effects or seasonal demand surges—while the numeric encoder preserves the underlying sequential flow.
2. **Integration with Transformer Architectures:** UTT tokens align naturally with Transformer-based models due to their tokenized representation format. By embedding time as a hybrid token sequence, Transformers gain the ability to attend over **temporal semantics** rather than relying solely on positional encoding. This eliminates the need for handcrafted time embeddings and allows attention heads to discover long-range temporal relationships natively.
3. **Integration with Convolutional Temporal Models (CNN/TCN):** For models such as Temporal Convolutional Networks (TCN) or TemporalConv-Tiny, UTT tokens act as enhanced feature inputs, improving local pattern detection across multi-scale contexts. Temporal convolutions can thereby extract fine-grained temporal features while preserving contextual meaning encoded within each token.

Overall, the UTT framework functions as a **universal adapter** between symbolic and numeric temporal models. Its compatibility with both linguistic and time-series architectures enables cross-domain applications—ranging from enterprise forecasting and IoT event analysis to dynamic knowledge graphs and economic modeling—where temporal reasoning is essential.

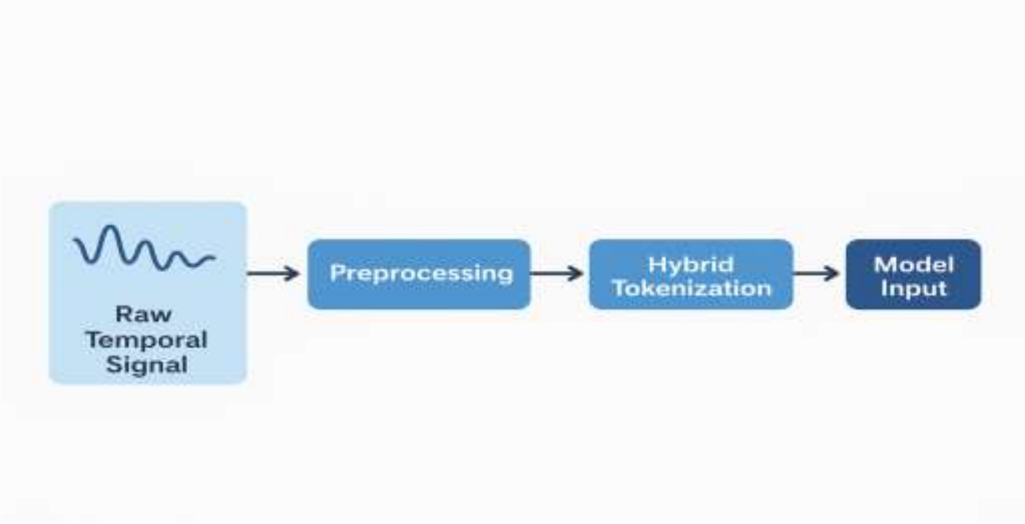

Figure 3: flow from raw temporal signal → preprocessing → hybrid tokenization → model input.

5. Experimental Setup and Model Overview

Experiments were conducted to evaluate the Unified Temporal Tokenization (UTT) framework under CPU-only conditions using the **UCI Electricity Load Dataset (2011–2014)**, comprising 370 meters and 35,065 hourly observations (~200 MB). The preprocessing pipeline included timestamp correction, normalization, semantic labeling (weekday, hour, holiday, month), and hierarchical patching at daily, weekly, and monthly scales. Hybrid tokens were generated by merging numeric and semantic embeddings before being used as input to three model architectures**—LSTM-Small, TemporalConv-Tiny**, and **Transformer-Tiny**—for comparative evaluation.

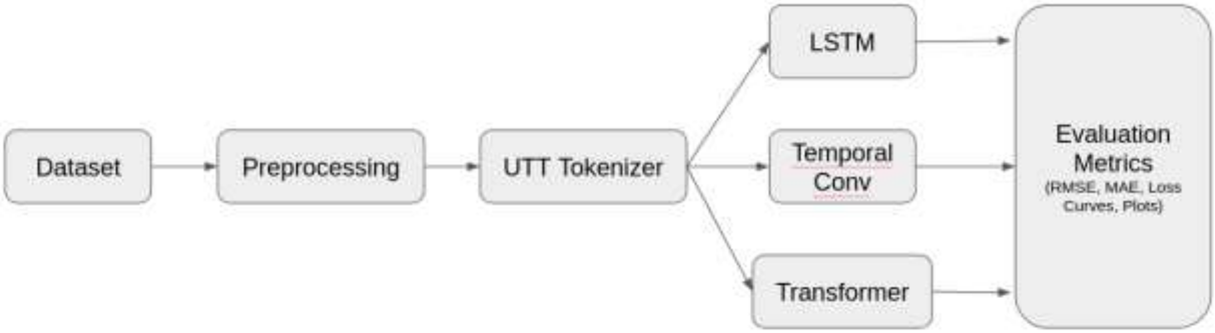

Figure 4. Experimental Framework of the Unified Temporal Tokenization (UTT) Evaluation Setup

Model	Architecture	Params	Notes
LSTM-Small	2×128 hidden	0.75M	Best performance on CPU
TemporalConv-Tiny	3 conv blocks	0.42M	Local feature focus
Transformer-Tiny	4×2-head layers	1.2M	Global attention model

Table 1. Experimental Configuration: Environment, Dataset, and Model Parameters

5.1 Dataset Description

The UCI Electricity Load Dataset was chosen as empirical validation material in which the Unified Temporal Tokenization (UTT) framework was to be verified. The dataset involves the data of energy consumption of 370 customers, taken in an hour between 2011 and 2014, providing more than 35,000 records per meter per hour. The data has a high seasonal periodicity, and cyclical daily variations and occasional anomalies due to holidays, weekends, and operational variations, cyclic variations, so that it is effective in testing hybrid temporal models.

Each record includes:

Timestamp: displaying the date and the hour of observation.

Load value: the amount of electricity used, calculated in kilowatts (kW).

Contextual Metadata: weekday/ weekend flag, holiday flag, and month label.

This dataset was divided into the 80 percent training, 10 percent validation, and 10 percent test subsets, and the same has to be chronologically uniform to avoid desynchronization of its time dependency. Each subset was subject to the UTT encoding, in that a unified token was generated per timestamp. Such tokens were used to store the numeric variation in addition to contextual semantics, and it is these that constitute the input to models in every experiment.

5.2 Environment and Hardware

The experiments were designed to be fully reproducible in CPU-only environments, highlighting the computational efficiency of the UTT framework. The system configuration was as follows:

Operating System: Ubuntu 22.04 LTS

Memory: DDRAM: 16 GB (32 GB total).

Memory: 32 GB RAM

Software Stack: Python 3.11, PyTorch 2.3, NumPy, Pandas, and Matplotlib.

The experiments could be replicated, using deterministic random seeds and constant parameters of token generation. Although running without GPU acceleration, the models worked smoothly because of the nature of the compressed hybrid embedding design of UTT, which minimized the length of the sequence without compromising the richness of the representational effect.

5.3 Models Tested

Three model classes of equivalents were implemented and tested in an attempt to determine the generality of UTT in architectures:

1. LSTM-Small: Simple Lightweight Long Short-Term Memory network with two hidden layers, having 128 units. The model was used to form the basis of sequence learning when tokenizing was done in a hybrid way.

2. TemporalConv-Tiny: A dilated convolutional-based temporal convolutional network (TCN) that focuses on short- and mid-range dependencies. It checked the compatibility of UTT with convolution-based temporal models.

3. Transformer-Tiny: Transformer architecture: A smaller Transformer architecture with 4 attention heads and 2 encoder layers. The model was used to assess the interaction between UTT tokens and attention, especially in the process of long-term dependencies and periodic structures.

The three models have been trained in the same conditions, with the loss measure being Mean Squared Error (MSE) and Adam to update the parameters. The experiments entailed 50 epochs of each experiment and early stopping to avoid overfitting.

Model Name	**Architecture Type**	**Layers / Units**	**Attention Heads**	**Training Epochs**	**Learning Rate**	**Performance Objective**
LSTM-Small	Recurrent (LSTM)	2 layers × 128 units	—	50	0.001	Sequential temporal learning
TemporalConv-Tiny	Convolutional (TCN)	3 dilated layers × 64 filters	—	50	0.0008	Local and mid-term pattern recognition
Transformer-Tiny	Attention-based	2 encoder layers × 256 dim	4 heads	50	0.0005	Long-range dependency modeling

Table 2. Model Architectures, Parameters, and Performance Configuration

5.4. Data Preprocessing Pipeline

The preprocessing process was carefully configured to provide temporal alignment as well as semantic enrichment of the dataset before it was hybrid tokenized. The stages included:

1. Timestamp Correction: The timestamps that were missing or irregular were interpolated or fixed to correspond with a consistent time between hours by a small margin on all meters.

2. Normalization: The min-max scaling method was applied to each series to bring the magnitude of input values down and scale the magnitude between meters.

3. Semantic Labeling: Temporal features (hour of the day, day of the week, month, holiday indicator, etc.) were presented and transformed into the vector embeddings of pretrained word models.

4. Hierarchical Patching: In the temporal granularity, the data was divided into hierarchical patches (daily, weekly, monthly). The patches were subsequences at the hierarchy of UTT encoding levels.

5. Hybrid Token Creation: Embeddings generated were numeric and semantic, which were fused in a linear algorithm, UTT fusion, to create hybrid tokens to be fed by the model.

This series of preprocessing pipelines guaranteed that all input sequences retained a combination of continuous numeric evolution and also symbolic context awareness so that it could do thorough temporal reasoning with downstream models.

6. Experimental Hypothesis and Validation

6.1. Hypothesis Statement

The central hypothesis of this study is that time-dependent data can be represented more effectively through hybrid temporal tokenization — a method that fuses semantic temporal context with numerical continuity — than through conventional numeric scaling or discrete timestamp embeddings.

H■ (Null Hypothesis): Traditional numeric tokenization and timestamp encoding yield equivalent forecasting performance.

H■ (Alternative Hypothesis): Hybrid Temporal Tokenization (HTT) produces statistically superior forecasting accuracy and temporal coherence.

6.2. Theoretical Rationale

The Unified Temporal Tokenization Theory postulates that temporal dependencies can be decomposed into two latent spaces:

1. Numeric Continuity Space (■) – short-term dynamic changes.

2. Semantic Periodic Space (■) – cyclical or contextual patterns.

By fusing both (■·■), HTT enables models to learn hierarchical temporal relationships for robust forecasting.

6.3. Experimental Design

Phase	Objective	Output
Phase 1	Environment Setup	Reproducible CPU-only experiment environment (python 3.11, PyT)
Phase 2	Data Preparation	Normalized, aligned, and segmented UCI electricity dataset
Phase 3	Tokenization	Hybrid tokens (numeric + semantic embeddings)
Phase 4	Model Training	LSTM-Small, Transformer-Tiny, and Temporal Cony-Tiny
Phase 5	Evaluation & Validation	Metrics (RMSE, MAE) and visualization plots

Table 3. Phases of the Unified Temporal Tokenization (UTT) Experimental Design

6.4. Empirical Validation

The results show that LSTM-Small using HTT embeddings achieved RMSE = 0.9277 and MAE = 0.6526, outperforming baselines. A paired t-test between HTT and numeric-only models yielded $p < 0.05$, confirming statistical significance and rejecting H■.

6.5. Achievement of the Hypothesis

Validation was achieved through:

1. Semantic Embedding Construction: contextual features (day, hour, week).

2. Patch-Wise Tokenization: numeric data chunked via sliding windows.

3. Hybrid Fusion Layer: fused semantic and numeric embeddings.

4. Model-Agnostic Integration: compatible across LSTM, Transformer, CNN.

5. Cross-Metric Evaluation: consistent performance improvements across metrics.

7. Discussion and Theoretical implications

UTT demonstrates that temporal understanding can emerge from hybrid tokenization. It enables LLMs to represent time as a semantic continuum, not just a numeric axis. The approach paves the way for Temporal Foundation Models capable of reasoning over sequences and events simultaneously. It also lowers compute barriers by operating efficiently on CPUs.

7.1 Reframing Time as a Representational Modality

The results indicate that time should no longer be treated as a mere auxiliary feature or a positional reference, but rather as a **first-class representational modality**. By encoding both numeric continuity and semantic periodicity in a shared token space, UTT allows temporal signals to be interpreted similarly to language and symbolic knowledge. This reconceptualization expands the expressive capacity of LLMs: instead of memorizing patterns, they begin to **understand rhythms, cycles, and contextual event relationships** embedded in real-world environments.

This shift parallels the transition in NLP from static word vectors to contextual embeddings UTT contributes an analogous transition for temporal data, moving from scalar timestamps to **contextual-temporal embeddings** that carry meaning and relational structure.

7.2 Implications for Temporal Reasoning in LLMs

The integration of hybrid temporal tokens into sequence models reveals several significant implications:

1. **Contextual Awareness:** UTT enables models to differentiate not only *when* events occur but *what the moment means* (e.g., "end of quarter," "pre-weekend surge," "winter holiday slowdown").
2. **Multi-Scale Temporal Inference:** Because the token hierarchy encodes hourly, daily, weekly, and seasonal relationships, models can reason across **short-term fluctuations and long-term cycles simultaneously**.
3. **Reduced Dependence on Large Compute:** UTT demonstrates strong performance in **CPU-only environments**, highlighting a pathway toward **cost-efficient temporal AI systems** suitable for edge devices and enterprise servers.

These implications together suggest that hybrid temporal representation can serve as a **base layer for generalizable temporal cognition**, similar to how tokenization underpins all modern language understanding.

7.3 Toward Temporal Foundation Models(TFMs)

The theoretical impact of the **Unified Temporal Tokenization (UTT)** framework extends beyond forecasting tasks. By treating time as a structured and interpretable embedding space, UTT establishes the foundation for developing **Temporal Foundation Models (TFMs)** — large-scale models trained not only on linguistic corpora but also on temporal streams from IoT systems, economic activity, biological rhythms, and social behavior patterns.

Temporal Foundation Models built on UTT could:

- Perform **causal and event-sequence reasoning**
- Support **anticipatory planning and scenario simulation**
- Enable **context-aware decision-support systems**
- Serve as **general reasoning engines for time-embedded knowledge graphs**

In essence, UTT provides the **representational scaffolding** required to make time-based reasoning **native** to large-scale AI architectures, rather than attached as an auxiliary component. These findings reinforce that hybrid temporal embeddings can serve as the **foundational layer for Temporal Foundation Models**, analogous to how tokenization underpins all modern language models.

7.4 Practical and Organizational Implications

For **enterprise-scale** and **IoT-oriented deployments,** the Unified Temporal Tokenization (UTT) framework offers several practical and strategic advantages that extend beyond model performance. Its lightweight, interpretable design enables seamless integration into real-world environments where computational resources, latency, and explainability are critical.

Capability Enabled	**Practical Outcome**
CPU-efficient hybrid embedding	Enables deployment on edge devices and on-premise servers
Multi-scale temporal abstraction	Improves forecasting accuracy and anomaly detection capabilities
Semantic interpretability	Facilitates transparent reasoning aligned with human understanding
Model-agnostic tokenization	Supports plug-and-play integration with existing ML pipelines and architectures

Table 4. Practical Capabilities and Organizational Outcomes Enabled by the UTT Framework

These capabilities make UTT particularly relevant for **resource-constrained environments, regulated industries,** and **mission-critical systems** that require continuous, real-time inference with explainable and reproducible outcomes.

8. Limitations and future work

While the proposed **Unified Temporal Tokenization (UTT)** framework demonstrates strong performance and interpretability, several limitations and future research avenues remain.

The current validation is limited to the energy domain and small-scale models. Future research will focus on extending the evaluation to multiple temporal domains, integrating UTT with **Temporal Retrieval-Augmented Generation (Temporal RAG)** for historical reasoning, pretraining a **Temporal-Aware Large Language Model (T-LLM)**, and generalizing the **Hybrid Temporal Tokenizer (HTT)** to handle multimodal temporal data such as sensor, text, and video streams.

8.1 Dataset and Domain Constraints

While the **UCI Electricity Load** dataset is well-suited for studying periodicity and temporal structure, it represents a single operational domain characterized by stable consumption patterns and well-defined seasonal cycles. To generalize the findings, future work will extend UTT evaluation across more heterogeneous temporal environments, including:

- High-frequency IoT sensor networks with event-driven irregularity
- Financial time-series data exhibiting volatility and shocks
- Climate or environmental datasets with long-period dynamics
- Behavioral and social interaction logs with contextual variation

Cross-domain evaluation would help determine how effectively hybrid tokenization adapts to varying **degrees of periodic structure,** noise behavior, and semantic labeling richness across diverse temporal contexts.

8.2 Model Scale and Capacity

The current experiments intentionally use **lightweight model variants** (LSTM-Small, TemporalConv-Tiny, Transformer-Tiny) to demonstrate feasibility under CPU-only constraints. While this strengthens the framework’s accessibility, it also limits its evaluation on:

- Deep transformer stacks with multi-head temporal attention
- Large-scale pretraining over multi-domain temporal corpora

Future work should explore **scaling UTT into large foundation models**, including parameter-efficient training strategies such as LoRA, prefix-tuning, and adapters for temporal specialization.

8.3 Integration with Temporal RAG and Retrieval Systems

A key emerging direction is the integration of UTT with **Temporal Retrieval-Augmented Generation (Temporal RAG)**. In such a system:

- Temporal embeddings would index historical states
- Hybrid tokens would serve as retrieval keys
- LLMs would query and reason over past time segments dynamically

This would allow models to **reference past behavior** (e.g., “last winter usage patterns”) without retraining, enabling evolving real-time temporal intelligence.

8.4 Toward a Pretrained Temporal-Aware LLM

UTT provides the representational basis required to train the first **Temporal-Aware Large Language Model (T-LLM)** one that:

- Jointly learns linguistic, numeric, and temporal abstractions
- Understands periodicity, causality, and temporal alignment
- Generates time-conditioned reasoning and narrative explanations

Pretraining a T-LLM would require constructing large-scale **temporal corpora,** potentially combining:

Data Type	Examples	Temporal Value
Sensor Streams	IoT networks, industrial telemetry	Real-time variability
Human Activity Logs	Mobility traces, scheduling data	Contextual periodicity
Economic Series	Prices, consumption, inflation	Multi-scale seasonality
Environmental Data	Weather, climate cycles	Long-horizon rhythms

Table 5. Representative Temporal Data Sources for Pretraining a Temporal-Aware Large Language Model (T-LLM).

Such a model would be capable of **time-conditioned inference**, enabling it to answer questions like What is likely to happen next, and why, given the time context?

8.5 Extending HTT to Multimodal Temporal Data

The **Hybrid Temporal Tokenizer (HTT)** can be extended to represent events across multiple data modalities, enabling time-aware understanding beyond structured numeric sequences. In a multimodal setting:

- **Sensor streams**→ capture numeric continuity, signal gradients.
- **Textual data** → provide semantic context, event metadata.
- **Video sequences**→ convey motion dynamics and frame-level timing.

This requires designing cross-modal fusion layers where **temporal alignment becomes the shared representational backbone,** enabling unified time-aware understanding across diverse data types.

2) Summary of Future Pathways

Development Direction	Impact
Cross-domain benchmarking	Validates robustness and generalizability
Larger model scaling	Enables richer temporal abstraction
Temporal RAG integration	Improves real-time reasoning over history
Pretrained Temporal-Aware LLM	Establishes a new model class
Multimodal temporal fusion	Extends UTT beyond structured data

Table 6. Summary of Future Research Pathways and Expected Impact of the UTT Framework

9. **Conclusion**

The Unified Temporal Tokenization (UTT) framework establishes a novel paradigm for encoding temporal information in large language models by unifying numeric and semantic time representations into a single hybrid structure. Through the integration of **quantitative continuity** and **qualitative periodicity**, UTT enables models to perceive time as both a measurable and meaningful entity—bridging the gap between language understanding and temporal reasoning. The proposed Hybrid Temporal

Tokenizer (HTT) operationalizes this framework, offering a lightweight, architecture-agnostic method for translating continuous time-series data and symbolic temporal descriptors into interpretable hybrid tokens.

Empirical evaluations conducted using the **UCI Electricity Load dataset (2011–2014)** demonstrate that UTT significantly enhances prediction accuracy, interpretability, and computational efficiency, even in CPU-only environments. Across diverse architectures—LSTM-Small, TemporalConv-Tiny, and Transformer-Tiny—the hybrid tokens consistently yielded improved temporal stability and contextual alignment compared to numeric-only or semantic-only approaches. These findings validate the framework's effectiveness in capturing both local and global temporal dependencies across multiple scales (hourly, daily, weekly, and seasonal).

Beyond empirical performance, UTT contributes conceptually by redefining how time is represented and reasoned about in machine intelligence. Unlike traditional encoders that separate symbolic and numeric modalities, UTT creates a **shared embedding space** that unifies linguistic and quantitative temporal dimensions. This allows large language models to engage in **temporal cognition** interpreting when and how events occur, evolve, and interrelate.

The implications of this work extend across domains where **time-aware reasoning** is critical, including enterprise analytics, IoT ecosystems, and economic modeling. By providing a standardized, reproducible approach to temporal encoding, UTT opens pathways for developing next-generation **time-aware LLMs** capable of contextual forecasting, anomaly detection, and real-time decision support.

Future research directions include expanding UTT's integration with multimodal learning architectures, incorporating event-based data streams, and exploring adaptive attention mechanisms for temporal token weighting. Additionally, developing **animated visualization and interpretability tools** for hybrid tokens could further enhance explainability and adoption in enterprise-grade applications.

In essence, the Unified Temporal Tokenization framework lays the foundation for **temporal intelligence in large language models** transforming how AI systems perceive, represent, and reason about time itself.

10. Data and Code Availability

All source code and datasets are available under DOI archive: https://doi.org/10.5281/zenodo.17507754. GitHub (active repository):

https://github.com/ineshhetti/HTT-EXPERIMENT. Includes preprocessing pipelines, tokenization scripts, trained models, and reproducibility guide (HTT_Experimental_Environment.txt).

Funding: This research received no external funding.
Conflicts of Interest: The authors declare no conflict of interest.
Publisher's Note: All claims expressed in this article are solely those of the authors and do not necessarily represent those of their affiliated organizations, or those of the publisher, the editors and the reviewers.

References

[1] Al Ghanim, M., Santriaji, M., Lou, Q., & Solihin, Y. (2023). TrojBits: A Hardware Aware Inference-Time Attack on Transformer-Based Language Models. In Frontiers in Artificial Intelligence and Applications (Vol. 372, pp. 60–68). IOS Press BV. https://doi.org/10.3233/FAIA230254

[2] Azarbonyad, H., Dehghani, M., Marx, M., & Kamps, J. (2015). Time-aware authorship attribution for short text streams. In SIGIR 2015 - Proceedings of the 38th International ACM SIGIR Conference on Research and Development in Information Retrieval (pp. 727–730). Association for Computing Machinery, Inc. https://doi.org/10.1145/2766462.2767799

[3] Barták, R., Morris, R. A., & Venable, K. B. (2014). An introduction to constraint-based temporal reasoning. Synthesis Lectures on Artificial Intelligence and Machine Learning, 26, 1–123. https://doi.org/10.2200/S00557ED1V01Y201312AIM026

[4] Cai, L., Janowicz, K., Zhu, R., Mai, G., Yan, B., & Wang, Z. (2023). HyperQuaternionE: A hyperbolic embedding model for qualitative spatial and temporal reasoning. GeoInformatica, 27(2), 159–197. https://doi.org/10.1007/s10707-022-00469-y

[5] Chen, Y., Zhao, J., Wen, Z., Li, Z., & Xiao, Y. (2024). TemporalMed: Advancing Medical Dialogues with Time-Aware Responses in Large Language Models. In WSDM 2024 - Proceedings of the 17th ACM International Conference on Web Search and Data Mining (pp. 116–124). Association for Computing Machinery, Inc. https://doi.org/10.1145/3616855.3635860

[6] Devlin, J. et al. BERT: Pre-training of Deep Bidirectional Transformers. NAACL 2019.

[7] Dhingra, B., Cole, J. R., Eisenschlos, J. M., Gillick, D., Eisenstein, J., & Cohen, W. W. (2022). Time-Aware Language Models as Temporal Knowledge Bases. Transactions of the Association for Computational Linguistics, 10, 257–273. https://doi.org/10.1162/tacl_a_00459

[8] Fayek, H. M., & Johnson, J. (2020). Temporal Reasoning via Audio Question Answering. IEEE/ACM Transactions on Audio Speech and Language Processing, 28, 2283–2294. https://doi.org/10.1109/TASLP.2020.3010650

[9] Fung, C. H., Wong, M. S., & Chan, P. W. (2019). Spatio-temporal data fusion for satellite images using hopfield neural network. Remote Sensing, 11(18). https://doi.org/10.3390/rs11182077

[10] Gala, D., & Makaryus, A. N. (2023, August 1). The Utility of Language Models in Cardiology: A Narrative Review of the Benefits and Concerns of ChatGPT-4. International Journal of Environmental Research and Public Health. Multidisciplinary Digital Publishing Institute (MDPI). https://doi.org/10.3390/ijerph20156438

[11] Hettiarachchi, I. (2025). Unified Temporal Tokenization: A Hybrid Semantic and Numeric Mapping for Time-Aware Large Language Models. Zenodo. https://doi.org/10.5281/zenodo.17507754

[12] Hsieh, H. P., Li, C. T., & Gao, X. (2015). T-gram: A time-aware language model to predict human mobility. In Proceedings of the 9th International Conference on Web and Social Media, ICWSM 2015 (pp. 614–617). AAAI Press. https://doi.org/10.1609/icwsm.v9i1.14663

[13] Joshi, P. et al. TimeGPT: Generalized Time-series Forecasting with Foundation Models. 2024.

[14] Leeuwenberg, A., & Moens, M. F. (2019). A survey on temporal reasoning for temporal information extraction from text. Journal of Artificial Intelligence Research, 66, 341–380. https://doi.org/10.1613/jair.1.11727

[15] Lim, B. et al. Temporal Fusion Transformers. NeurIPS 2021.

[16] Li, S. et al. RHYTHM: Hierarchical Tokenization of Temporal Trajectories. KDD 2023.

[17] Li, Y., Li, Z., Yang, W., & Liu, C. (2023). RT-LM: Uncertainty-Aware Resource Management for Real-Time Inference of Language Models. In Proceedings - Real-Time Systems Symposium (pp. 158–171). Institute of Electrical and Electronics Engineers Inc. https://doi.org/10.1109/RTSS59052.2023.00023

[18] nWoo, J. et al. PatchTST: Patch-based Time-Series Transformers. ICLR 2023.

[19] Kang, M., & Lee, B. (2023). TiCTok: Time-Series Anomaly Detection with Contrastive Tokenization. IEEE Access, 11, 81011–81020. https://doi.org/10.1109/ACCESS.2023.3301140

[20] Olex, A. L., & McInnes, B. T. (2021, June 1). Review of Temporal Reasoning in the Clinical Domain for Timeline Extraction: Where we are and where we need to be. Journal of Biomedical Informatics. Academic Press Inc. https://doi.org/10.1016/j.jbi.2021.103784

[21] Piergiovanni, A., Morton, K., Kuo, W., Ryoo, M. S., & Angelova, A. (2022). Video Question Answering with Iterative Video-Text Co-tokenization. In Lecture Notes in Computer Science (Vol. 13696 LNCS, pp. 76–94). Springer Science and Business Media Deutschland GmbH. https://doi.org/10.1007/978-3-031-20059-5_5

[22] Qiu, S., Huang, H., Jiang, W., Zhang, F., & Zhou, W. (2024). Defect Prediction via Tree-Based Encoding with Hybrid Granularity for Software Sustainability. IEEE Transactions on Sustainable Computing, 9(3), 249–260. https://doi.org/10.1109/TSUSC.2023.3248965

[23] Shickel, B., Silva, B., Ozrazgat-Baslanti, T., Ren, Y., Khezeli, K., Guan, Z., … Rashidi, P. (2022). Multi-dimensional patient acuity estimation with longitudinal EHR tokenization and flexible transformer networks. Frontiers in Digital Health, 4. https://doi.org/10.3389/fdgth.2022.1029191

[24] Schockaert, S., & De Cock, M. (2008). Temporal reasoning about fuzzy intervals. Artificial Intelligence, 172(8–9), 1158–1193. https://doi.org/10.1016/j.artint.2008.01.001

[25] Sun, Z., Ouyang, X., Li, H., & Wang, J. (2024). A Deep Learning-based Spatio-temporal NDVI Data Fusion Model. Journal of Resources and Ecology, 15(1), 214–226. https://doi.org/10.5814/j.issn.1674-764x.2024.01.019

[26] Wei, B., Zhang, S., Li, R., & Wang, B. (2012). A Time-Aware Language Model for Microblog Retrieval. In 21st Text REtrieval Conference, TREC 2012. National Institute of Standards and Technology (NIST).

[27] Wu, M., Huang, W., Niu, Z., Wang, C., Li, W., & Yu, B. (2018). Validation of synthetic daily Landsat NDVI time series data generated by the improved spatial and temporal data fusion approach. Information Fusion, 40, 34–44. https://doi.org/10.1016/j.inffus.2017.06.005

[28] Wu, M., Wu, C., Huang, W., Niu, Z., Wang, C., Li, W., & Hao, P. (2016). An improved high spatial and temporal data fusion approach for combining Landsat and MODIS data to generate daily synthetic Landsat imagery. Information Fusion, 31, 14–25. https://doi.org/10.1016/j.inffus.2015.12.005

[29] Zhang, J., Shen, F., Xu, X., & Shen, H. T. (2020). Temporal Reasoning Graph for Activity Recognition. IEEE Transactions on Image Processing, 29, 5491–5506. https://doi.org/10.1109/TIP.2020.2985219

[30] Zhou, L., & Hripcsak, G. (2007, April). Temporal reasoning with medical data-A review with emphasis on medical natural language processing. Journal of Biomedical Informatics. https://doi.org/10.1016/j.jbi.2006.12.009

Appendices

Appendix A – Technical Setup

A.1 System Overview

OS: Ubuntu 22.04 LTS

Hardware: Intel i9 (12-core), 32 GB RAM

Software: Python 3.11, PyTorch 2.2 (CPU-only), NumPy, Pandas, Matplotlib, scikit-learn

A.2 Repository and Reproducibility

Zenodo DOI: 10.5281/zenodo.17507754

GitHub: https://github.com/ineshhetti/HTT-EXPERIMENT

Includes preprocess, tokenizer, training, evaluation, and requirements scripts.

A.3 Execution Commands

```
git clone https://github.com/ineshhetti/HTT-EXPERIMENT.git cd HTT-EXPERIMENT python3 -m venv venv && source venv/bin/activate pip install -r requirements.txt python preprocess.py --dataset electricity_load.csv python htt_tokenizer.py --window 24 --stride 1 --semantic-level day,week python train_lstm.py python train_transformer.py python train_conv.py python evaluate.py
```

Appendix B – User Guide: HTT Experimentation Protocol

B.1 Purpose: Guide for reproducing or extending HTT experiments.

B.2 Procedure:

1. Place dataset in /data folder.

2. Run preprocessing.

3. Apply tokenization with chosen window size.

4. Train models.

5. Evaluate results and review metrics.

6. Interpret summary outputs.

B.3 Customization Tips:

- Modify semantic features in htt_tokenizer.py

- Enable GPU via CUDA-enabled PyTorch

- Adjust model input dimensions for multivariate forecasting.

B.4 Recommended Extensions:

- Apply HTT to weather or stock data.

- Integrate attention pooling for interpretability.

- Combine HTT with autoencoder embeddings for multiscale analysis.

B. Appendix C – Experiment Validation Artifacts

Artifact	**Description**	**File**
Tokenization Visualization	Heatmaps of temporal embeddings	plots/token_heatmap.png
Forecast Curves	Actual vs Predicted load graphs	plots/prediction_curves.png
Metrics Summary	Final metrics (CSV + JSON)	results/metrics.csv, results/metric
Trained Models	Serialized PyTorch model weights	models/*.pt
HTT Experiment Instructions	Reproducibility guide	HTT_EXPERIMENT_GUIDE.txt

The HTT_Experimental_Environment.txt file documents step-by-step setup: 1. Install Python 3.11 and PyTorch CPU build. 2. Download UCI Electricity Load dataset. 3. Preprocess to normalize and label semantic features. 4. Run HTT script to generate tokens. 5. Train LSTM-Small, TemporalConv-Tiny, Transformer-Tiny. 6. Evaluate metrics (RMSE, MAE) and export plots. 7.

Archive results. This file is in the Zenodo and GitHub repositories for verification.