ChatTime:

A Unified Multimodal Time Series Foundation Model Bridging Numerical and Textual Data

Chengsen Wang, Qi Qi, Jingyu Wang, Haifeng Sun,
Zirui Zhuang, Jinming Wu, Lei Zhang, Jianxin Liao

Presenter: Yuefeiyang Li

yuefeiyang.li@stud.uni-heidelberg.de

1. Introduction

2. Method

3. Experiments

4. Conclusion

Introduction

What is a Time Series?

Ordered observations over time

- an ordered sequence of observations over time
- each value is associated with a timestamp
- the temporal order is essential
- **Examples**
  - temperature over days
  - electricity consumption over hours
  - traffic flow over time

What is Time Series Forecasting?

Predicting future values from past observations

- **Goal:** predict future values from past observations
- **Given:** a history window (past)
- **Predict:** a future horizon (future)
- mathematical form:

$$x_{1:C} \longrightarrow x_{C+1:C+H}$$

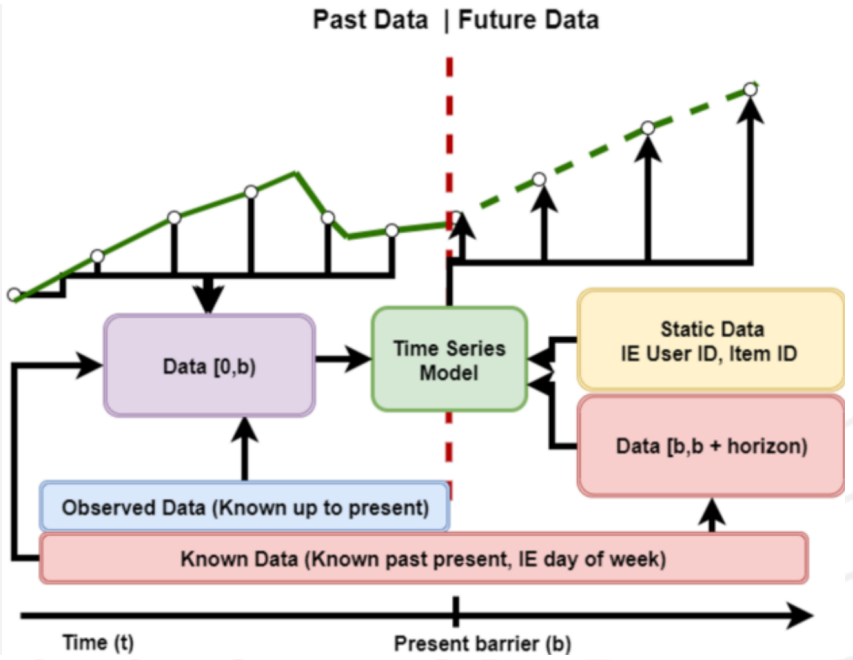


Illustration: forecasting future values from a history window

Zero-Shot TS Forecasting Challenges

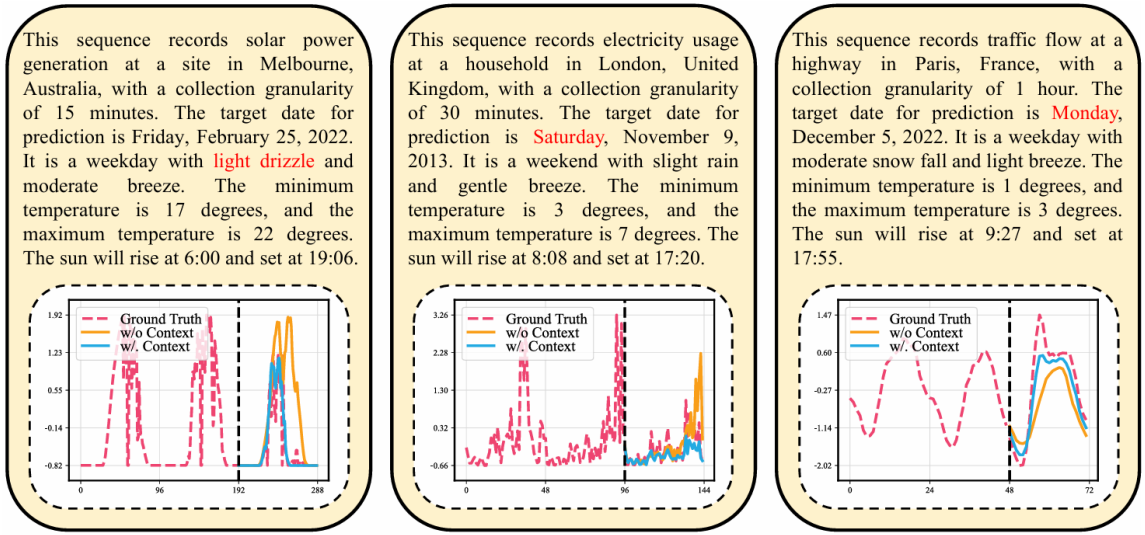
Why generalization across datasets is difficult

- time series differ drastically across datasets:
  - scale
  - frequency
  - noise & seasonality
- most models are trained for:
  - a specific dataset
  - fixed history & prediction lengths
- generalization across datasets is challenging

Numerical-Only Forecasting Limitations

Why purely numeric models are not enough

- Real-world forecasts depend on external context (calendar, weather, background information)
- Same history ⟶ different futures when conditions change (weekday vs. weekend, rain vs. clear sky)
- Unimodal numeric models rely only on past values ⇒ cannot capture external events without textual context



(a) MSPG (b) LEU (c) PTF

Figure 11: The showcase in the context-guided time series forecasting task.

Prior LLM-Based Approaches

Strengths and limitations of prior approaches

- **Prompt-only LLMs (direct prompting)**
  - Representative: LLMTIME
  - Limitation: inefficient numeric tokenization ⟶ high token/context cost
- **LLM + adapters / task-specific modules**
  - Representative: GPT4TS, TimeLLM
  - Limitation: dataset-specific adaptation ⟶ limited zero-shot generalization
- **Time-series foundation models (train from scratch)**
  - Representative: Chronos, TimesFM
  - Limitation: costly to train; primarily TS-focused (not designed for unified TS ↔ Text I/O)

Core Insight

Treating Time Series as a Foreign Language

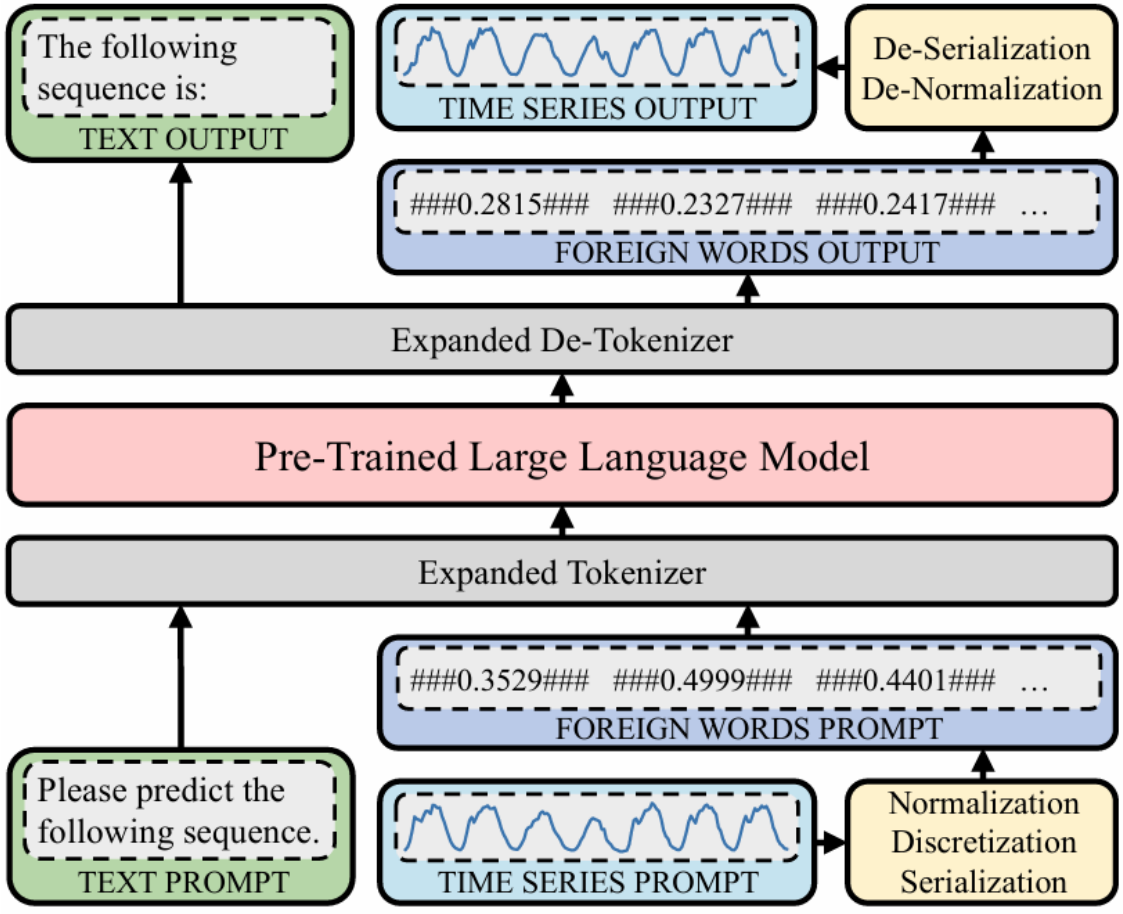
- Time series forecasting and language modeling share the same autoregressive structure: predicting the next value/token given previous context
  - At the core, both can be viewed as an *n-order Markov process*
- The key challenge is representation, not architecture
- **Idea**
  - Treat time series as a foreign language (discrete symbols)
  - Make time series tokenizable by LLMs
  - Reuse pretrained LLMs without modifying their architecture

Method

ChatTime Overview

From time series to LLM-compatible tokens

- **TS ⟶ "foreign words":** normalize + discretize + serialize.
- **Foreign words ⟶ LLM:** expanded tokenizer.
- **LLM ⟶ outputs:** expanded de-tokenizer ⟶ TS output / text output.



(a) ChatTime Architecture

The Key to Making the Pipeline Work

——Representation

- **Key observation:** LLMs operate on **discrete tokens**, not raw numbers.
- Time-series data consists of **continuous numerical values**.
- Directly feeding numbers into LLMs is **inefficient and unstable**.

→ **The core challenge lies in how time series are represented.**

Normalization

- **Why normalization**
  - Time-series scales vary widely across domains
  - Without normalization, discretization becomes dataset-dependent, hurting cross-dataset comparability and zero-shot generalization
- **Design choice**
  - Normalize using history-only min/max to avoid future leakage
  - Apply a shifted min–max normalization with a centered range
  - Reserve headroom as a buffer (no clipping of future values)

$$\tilde{x} = \frac{x - \min(x_{1:C})}{\max(x_{1:C}) - \min(x_{1:C})} - 0{,}5$$

- Result: history $\in [-0{,}5,\ 0{,}5]$; future values may fall outside.

Discretization

- **Binning (10K bins):** uniformly partition the normalized range $[-1, 1]$ into 10,000 bins
- Map each value to its bin and use the **bin center** as a discretized (lossy) value
- **Fix precision (4 decimals):** round discretized values to a fixed 4-decimal precision (same as LLMTIME)
- **Output:** a finite set of discretized numeric symbols (still as numbers/strings)

Serialization

- **Goal:** Convert the discretized series into an ordered “foreign-word” sequence.
- **Input:** Discretized values (bin centers) with fixed precision (4 decimals).
- **Serialization:** Arrange values in temporal order and format them as marked words (e.g., ###0.3529### ###0.4999### ###0.4401### ...)

Tokenization Mismatch

Evidence: numbers get split into many sub-tokens

- **Observation:** standard subword tokenizers split numeric strings bit-by-bit
- **Table 2 takeaway (token consumption):**
  - GPT: **34** tokens
  - LLaMA: **22** tokens
  - ChatTime: **7** tokens
- **Conclusion:** numeric tokenization is expensive → we must change how values are tokenized

Time Series
[0.2835, 0.2285, 0.1587, 0.4001]
GPT (34 tokens)
"2 8 3 5 , 2 2 8 5 , 1 5 8 7 , 4 0 0 1" ['2', '_', '8', '_', '3', '_', '5', '_', '_', '2', '_', '2', '_', '8', '_', '5', '_', '_', '1', '_', '5', '_', '8', '_', '7', '_', '_', '4', '_', '0', '_', '0', '_', '1']
LLaMA (22 tokens)
"2835, 2285, 1587, 4001" ['2', '8', '3', '5', ',', '_', '2', '2', '8', '5', ',', '_', '1', '5', '8', '7', ',', '_', '4', '0', '0', '1']
ChatTime (7 tokens)
"###0.2835### ###0.2285### ###0.1587### ###0.4001###" ['###0.2835###', '_', '###0.2285###', '_', '###0.1587###', '_', '###0.4001###']

Table 2: The comparison of token consumption between LLMTIME and ChatTime.

Tokenization & Vocabulary Expansion

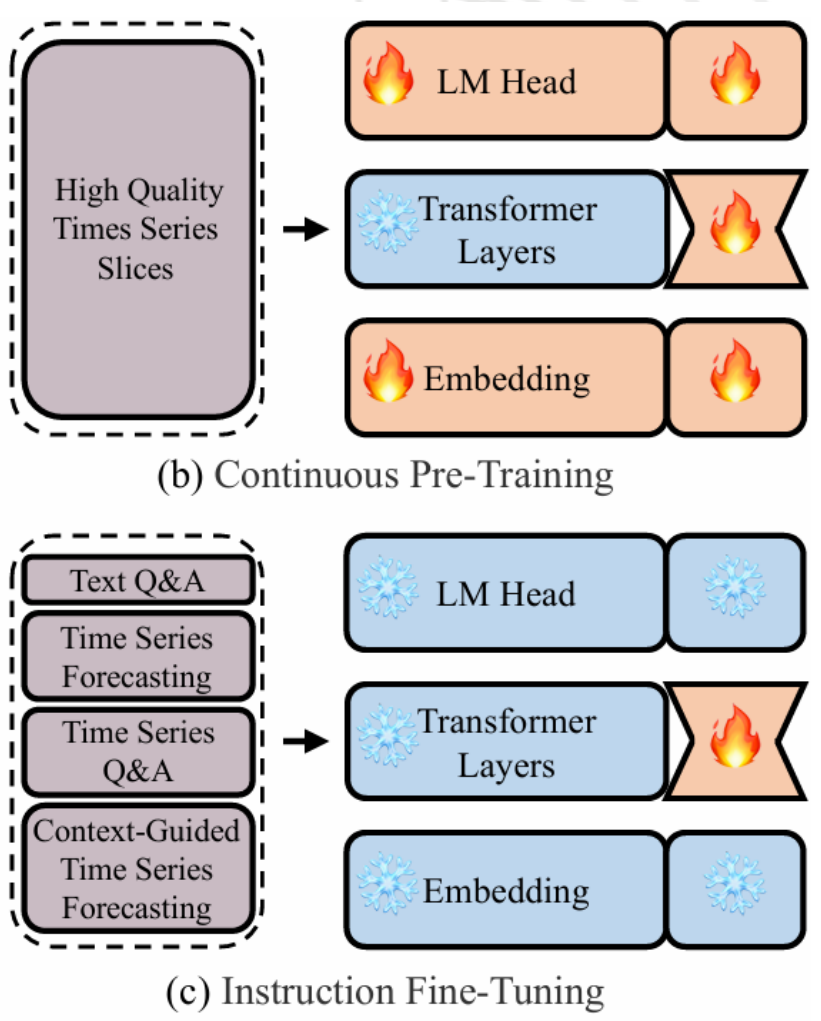
Make each value a single token

- **Mark characters → "foreign words":** wrap each discretized value as `###value###`
- **Vocabulary expansion:** add all `###value###` tokens (from the 10K bin centers) to the tokenizer vocabulary
- Add a special token `###Nan###` for missing values
- **Result:** after expansion, **one token per value** (no more token explosion)
- Training note (Sec. 3.3): after expanding the vocabulary, the **embedding layer** and **output head** must be trained to adapt to the new tokens

Training Strategy

Continuous pre-training + instruction fine-tuning

- **Pre-train:** AR forecasting on **1M high-quality TS slices** ⟶ learn time-series representations.
- **Fine-tune: 25K / task × 4** multimodal instructions ⟶ align **TS** ↔ **Text**.
- **Trained parameters:** Pre-train = Transformer + Embedding / LM head; Fine-tune = Transformer only.



TSQA Prompt Template (Trend)

- **Instruction:** choose (a/b/c)
- **Input:** tokenized series `###value###`
- **Response:** one option (a/b/c)
- Used for instruction tuning

You are a helpful assistant that performs time series analysis. The user will provide a sequence and you will respond to the questions based on this sequence.

Instruction:
Please answer the following question carefully after analyzing the sequence: Given the following definitions:
Constant trend: The time series does not show any significant increase or decrease over time.
Upward trend: The time series consistently increases over time.
Downward trend: The time series consistently decreases over time.
Select one of the following answers that best describes the provided time series:
(a) This time series has a constant trend.
(b) This time series has an upward trend.
(c) This time series has a downward trend.
Only answer (a), (b), or (c).

Input:
###-0.1079### ###0.1905### ###0.4999### ###0.4221### ###0.1359### ###-0.1209### ###0.1909### ###0.4643### ###0.3947### ###0.0713### ###-0.1843### ###0.1787###

Response:
(c)

Figure 6: The prompt of trend feature in the time series question answering task.

Experiments

Experimental Setup

Tasks, baselines, and evaluation

- **Tasks evaluated:**
  - ZSTSF — zero-shot time series forecasting
  - CGTSF — context-guided time series forecasting
  - TSQA — time-series question answering
- **Baselines:**
  - **Forecasting:** DLinear, iTransformer, GPT4TS, TimeLLM, TimeGPT, Moirai, TimesFM, Chronos
  - **CGTSF only:** TGForecaster; ChatTime^{-} (without text)
  - **TSQA:** GPT-4, GPT-3.5, GLM4, LLaMA3-70B
- **Evaluation metrics:**
  - Forecasting: MAE (Avg. Rank for overall comparison)
  - TSQA: Accuracy and qualitative analysis

Zero-shot Time Series Forecasting

ZSTSF

- **Goal**
  - Forecast future values without dataset-specific training
  - Shared model across datasets and domains
- **Setup**
  - 8 benchmark datasets across 4 domains
  - `(Hist, Pred)` settings follow Table 4
- **Evaluation**
  - Metric: MAE (reported as Avg. MAE / Avg. Rank)
  - Baselines: classical TS models, LLM-based methods, and TS foundation models
- **Protocol**
  - Evaluation datasets are excluded from ChatTime training (to avoid data leakage)

ZSTSF Results

Dataset	Hist	Pred	Full-Shot Forecast				Zero-Shot Forecast				
			DLinear	iTransformer	GPT4TS	TimeLLM	TimeGPT	Moirai	TimesFM	Chronos	ChatTime
ETTh1	48	24	0.1462	0.1650	**0.1389**	0.1467	**0.1604**	0.1694	0.2021	0.1634	0.1698
	72	24	**0.1358**	0.1852	0.1469	0.1439	0.1603	0.1796	0.1599	**0.1372**	0.1403
	96	24	**0.1398**	0.1964	0.1447	0.1473	0.1577	0.1433	0.1454	**0.1374**	0.1374
	120	24	**0.1371**	0.1971	0.1414	0.1513	0.1594	0.1492	0.1502	**0.1348**	0.1431
ETTh2	48	24	**0.2724**	0.2937	0.2742	0.2758	**0.2874**	0.2963	0.3360	0.3128	0.2906
	72	24	0.2756	0.3118	**0.2717**	0.2972	0.2888	0.3109	**0.2880**	0.3045	0.3092
	96	24	**0.2831**	0.3417	0.2900	0.2864	**0.2902**	0.3139	0.3144	0.3158	0.2917
	120	24	0.2863	0.3299	**0.2854**	0.3175	0.3026	**0.2905**	0.3311	0.3150	0.3124
ETTm1	192	96	0.1479	0.1608	**0.1384**	0.1503	0.1921	0.1608	0.1719	0.1604	**0.1442**
	288	96	0.1400	0.1813	**0.1345**	0.1425	0.1715	0.1848	0.1650	**0.1452**	0.1587
	384	96	**0.1428**	0.1680	0.1518	0.1452	0.1616	0.1619	0.1584	0.1463	**0.1393**
	480	96	**0.1406**	0.2001	0.1472	0.1527	0.1570	0.1703	0.1582	**0.1401**	0.1802
ETTm2	192	96	0.2793	0.3397	**0.2792**	0.2918	0.4294	0.4206	0.3405	0.3759	**0.3135**
	288	96	**0.2881**	0.3623	0.2918	0.2904	0.3625	0.3882	**0.3277**	0.3472	0.3340
	384	96	0.2947	**0.2880**	0.3089	0.3003	**0.3389**	0.3742	0.3562	0.3589	0.3434
	480	96	0.3014	0.3725	**0.2945**	0.3054	**0.3242**	0.3597	0.3679	0.3353	0.4213
Electric	48	24	0.5719	0.5951	**0.5008**	0.5733	**0.5276**	0.6617	0.6005	0.6098	0.6083
	72	24	0.5486	0.5619	**0.4896**	0.4989	**0.4953**	0.6018	0.5454	0.5914	0.6238
	96	24	0.5536	0.5290	**0.4432**	0.4816	0.4971	0.5260	0.5276	0.5139	**0.4951**
	120	24	0.4714	0.5622	**0.4540**	0.4848	0.5196	0.4963	**0.4900**	0.5031	0.5101
Exchange	14	7	0.0543	**0.0526**	0.0533	0.0531	0.0620	0.0784	0.0647	0.0555	**0.0540**
	21	7	0.0571	0.0547	**0.0505**	0.0505	0.0599	0.0812	0.0743	0.0635	**0.0556**
	28	7	0.0595	0.0581	**0.0508**	0.0511	0.0610	0.0844	0.0652	0.0595	**0.0559**
	35	7	0.0615	0.0607	**0.0493**	0.0524	0.0629	0.0677	0.0632	0.0598	**0.0558**
Traffic	48	24	0.4662	0.5000	0.4557	**0.4473**	0.4668	0.4887	0.4483	0.4718	**0.4220**
	72	24	0.4475	0.4443	**0.4116**	0.4252	0.4635	0.4581	0.4196	**0.3725**	0.3873
	96	24	0.4438	0.4348	0.4190	**0.4064**	0.4332	0.4082	**0.3714**	0.3787	0.4074
	120	24	0.4190	0.4149	**0.3416**	0.4279	0.4161	**0.3539**	0.3542	0.3908	0.4125
Weather	288	144	**0.0339**	0.0367	0.0364	0.0352	0.0331	**0.0305**	0.0354	0.0343	0.0352
	432	144	**0.0366**	0.0404	0.0401	0.0395	0.0321	0.0302	**0.0298**	0.0346	0.0356
	576	144	**0.0364**	0.0379	0.0399	0.0377	0.0328	0.0331	0.0321	0.0349	**0.0284**
	720	144	**0.0371**	0.0395	0.0392	0.0392	**0.0323**	0.0353	0.0369	0.0335	0.0332
Avg. MAE			0.2409	0.2661	**0.2286**	0.2390	0.2544	0.2659	0.2541	**0.2512**	0.2515
Avg. Rank			3.7500	6.9688	**3.0000**	3.9688	5.5625	6.5000	5.7500	4.8438	**4.4688**

Table 4: The evaluation result in the traditional unimodal time series forecasting task. The lower values for all metrics represent the better performance. The best results among full-shot and zero-shot forecasting methods are highlighted in bold, respectively.

Context-Guided Time Series Forecasting

CGTSF

- **Goal**
  - Forecast with textual context (time series + text)
- **Setup**
  - Datasets: MSPG, LEU, PTF (chronological split 6:2:2)
  - Same `(Hist, Pred)` settings and metric as ZSTSF
- **Evaluation**
  - Metric: MAE (reported as Avg. MAE / Avg. Rank)
  - Baselines: TGForecaster and ChatTime^{-} (without text)
- **Note**
  - Instruction tuning uses partial training data (not strict zero-shot)
  - Model weights are shared across scenarios

CGTSF Results

Dataset	Hist	Pred	Dataset-Specific Forecast				Dataset-Shared Forecast				
			DLinear	GPT4TS	TimeLLM	TGForecaster	Moirai	TimesFM	Chronos	ChatTime-	ChatTime
MSPG	192	96	**0.7136**	0.7558	0.7697	0.7595	0.8108	0.8362	0.7427	0.7606	**0.7346**
	288	96	**0.7083**	0.7464	0.7959	0.7610	0.7849	0.7896	0.7408	0.7606	**0.7353**
	384	96	**0.7014**	0.7388	0.7672	0.7638	0.7749	0.7811	0.7352	0.7607	**0.7330**
	480	96	**0.7018**	0.7311	0.7632	0.7695	0.7664	0.7667	0.7344	0.7607	**0.7292**
LEU	96	48	0.6676	0.6697	0.6531	**0.6181**	**0.6228**	0.6670	0.6571	0.6496	0.6305
	144	48	0.6495	0.6567	0.6474	**0.6355**	**0.6085**	0.6475	0.6597	0.6506	0.6231
	192	48	0.6407	0.6771	**0.6329**	0.6458	**0.6008**	0.6490	0.6645	0.6407	0.6111
	240	48	**0.6316**	0.6383	0.6356	0.6329	**0.5968**	0.6333	0.6631	0.6377	0.6085
PTF	48	24	0.5204	0.4373	**0.4211**	0.4411	0.5981	0.4851	**0.4813**	0.5155	0.4849
	72	24	0.5075	0.4253	0.4031	**0.3943**	0.5776	**0.4258**	0.4276	0.4436	0.4307
	96	24	0.4965	0.3921	0.4392	**0.3653**	0.5179	0.4054	0.4336	0.4172	**0.3920**
	120	24	0.4796	0.3713	0.3594	**0.3594**	0.5245	0.3807	0.3902	0.3943	**0.3480**
Avg. MAE			0.6182	0.6033	0.6073	**0.5955**	0.6487	0.6223	0.6109	0.6160	**0.5884**
Avg. Rank			4.7500	5.0833	4.9167	**3.9167**	5.9167	6.4167	5.5000	5.7500	**2.5833**

Table 5: The evaluation result in the context-guided time series forecasting task. The lower values for all metrics represent the better performance. The best results among dataset-specific and dataset-shared methods are highlighted in bold, respectively.

Time Series Question Answering

TSQA

- **Goal**
  - Answer natural-language questions about time series
  - Focus on high-level properties rather than exact numeric values
- **Setup**
  - Synthetic dataset generated via KernelSynth
  - Four evaluated features: trend, volatility, seasonality, outliers
- **Evaluation**
  - Metric: Accuracy (Acc), summarized as Avg. Acc / Avg. Rank
  - Baselines: GPT-4, GPT-3.5, GLM-4, LLaMA-3-70B
  - Time-series input uses two LLMTIME prompting strategies
- **Protocol**
  - Exclude the 25K instruction-tuning samples from evaluation

TSQA Results

Feat	Len	GPT4	GPT3.5	GLM4	LLaMA3	ChatTime
Trend	64	0.6532	0.3507	0.7319	0.6799	**0.9011**
	128	0.7015	0.5846	0.7574	0.5855	**0.9068**
	256	0.7482	0.5028	0.6377	0.6143	**0.8843**
	512	0.6346	0.5903	0.6697	0.6753	**0.8234**
Volatility	64	0.5585	0.5633	0.6797	0.6373	**0.7874**
	128	0.4979	0.3839	0.4770	0.4756	**0.6954**
	256	0.4624	0.4894	0.5418	0.5246	**0.6228**
	512	0.3169	0.3796	0.4549	0.5261	**0.5736**
Season	64	0.3518	0.3428	0.3366	0.3484	**0.6639**
	128	0.3515	0.3952	0.3464	0.3958	**0.6517**
	256	0.5283	0.5089	0.3892	0.4120	**0.6463**
	512	0.4457	0.4889	0.3892	0.4127	**0.6244**
Outlier	64	0.7230	0.4325	0.5359	0.7051	**0.8773**
	128	0.6327	0.5940	0.5298	0.5694	**0.9032**
	256	0.6795	0.4579	0.5019	0.5073	**0.8593**
	512	0.6219	0.4996	0.2822	0.4085	**0.7478**
Avg. Acc		0.5567	0.4728	0.5163	0.5299	**0.7605**
Avg. Rank		3.0625	4.0625	3.6250	3.2500	**1.0000**

Table 6: The evaluation result in the time series question answering task. Higher values mean better performance for all metrics, except Rank, which is better when lower. The best results are highlighted in bold.

Ablation Study

What makes ChatTime work?

- **Variants:** w/o AR = replace 1M continuous pre-training with 100K instruction data (epochs ×10); w/o CL = random slices instead of clustering-selected 1M; w/o TQA = remove text QA task (25K Alpaca) in instruction tuning.
- **Fig. 2 takeaway:** removing any component hurts performance — MAE ↑ on ZSTSF & CGTSF, Accuracy ↓ on TSQA.

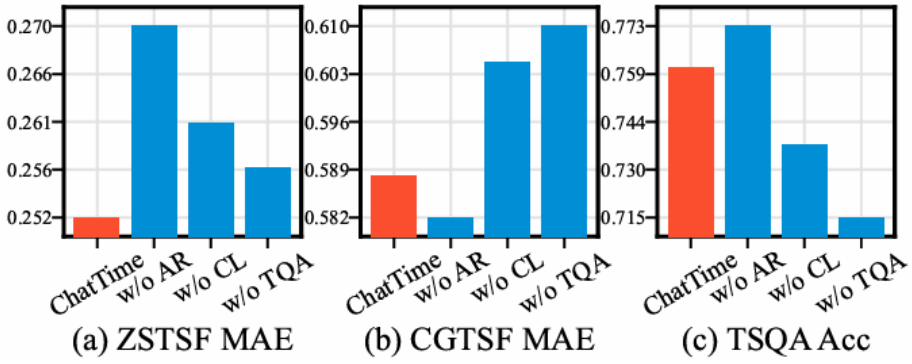


Figure 2: The evaluation result between ChatTime and variants. Lower values are better for ZSTSF and CGTSF, while higher values are better for TSQA.

Figure 2: Component-wise ablation results.

Conclusion

Conclusion
Key takeaways

- time series capture real-world temporality beyond language
- naïve LLMs struggle with numeric temporal signals
- ChatTime treats time series as a foreign language
- Enables a unified framework for:
  - TS $\longrightarrow$ TS
  - TS $\longrightarrow$ Text
  - Text $\longrightarrow$ TS
  - Text $\longrightarrow$ Text
- **ChatTime provides a unified framework for time series–text multimodal modeling.**

Discussion & Future Work

What the paper explicitly claims beyond results

- **Strengths**
  - No architecture change / no training from scratch: reuse a pre-trained LLM via vocabulary expansion
  - Unified processing of time series and text: supports bimodal inputs & outputs (TS $\leftrightarrow$ Text)
  - Lower training cost + extra textual capability compared to training TS foundation models from scratch
- **Limitation & future work**
  - Limitation: due to resource constraints, ChatTime has not yet reached saturation
  - Future work: scale data & compute; extend to anomaly detection, classification, and summarization

Thank you for listening!

Any questions?