Kairos: Redefining Event and Time Prediction with Language Modeling

Akash Gupta AKASHGGU@AMAZON.COM
Amazon, India

Sahil Verma VRSAHIL@AMAZON.COM
Amazon, India

Sumit Bisht SUBISHT@AMAZON.COM
Amazon, India

Nitika Verma NITIKAV@AMAZON.COM
Amazon, India

Purav Aggarwal AGGAP@AMAZON.COM
Amazon, India

Venkat Phanindra Kumar Grandhi PHANINDR@AMAZON.COM
Amazon, India

Abhishek Persad PERSADAP@AMAZON.COM
Amazon, India

Editors: Hung-yi Lee and Tongliang Liu

Abstract

Continuous time-event sequence (CTES) forecasting is essential across diverse domains, from healthcare to finance, requiring accurate prediction of both future event types and their timestamps. Traditionally, CTES forecasting has been driven by Temporal Point Processes (TPPs), which rely on intensity function-based priors. However, these methods often fail to generalize effectively to real-world scenarios as well as perform poorly over longer horizons. While recent diffusion-based approaches are promising, they are limited by the need to define a fixed prior while training, such as number of events to forecast or time horizon, which requires training multiple models for different horizon lengths. We present Kairos, a novel model that reformulates CTES forecasting as a language modeling task. Our model employs a decoder-only transformer architecture with a unified tokenization approach that represents time and events in a shared embedding space. By structuring the input as alternating event and time tokens, the model learns to capture the inherent temporal relationships between events. Through comprehensive experiments on multiple large-scale datasets, we demonstrate that Kairos consistently outperforms state-of-the-art baselines, achieving average improvements of 4.5% and 7.8% in short-term forecasting for event and time respectively and 14.41% improvement in long-term forecasting. Additionally, we conduct extensive ablation studies and qualitative analysis to understand the inner workings of Kairos.

Keywords: Continuous Time-Event Sequences; Language modeling applications; Self-consistency; Temporal reasoning; Event prediction

1. Introduction

Continuous time-event sequences (CTES) are defined as the sequence of discrete events and associated times in the continuous domain. These sequences are present in various domains such as healthcare (Wang et al., 2018), e-commerce (Hernández et al., 2017), social networks (Yang et al., 2011), etc. Forecasting CTES is crucial for various practical applications; for instance, modeling a patient's hospital visits as a CTES and predicting future health events can enable proactive healthcare interventions. CTES forecasting poses unique challenges, as it requires precise predictions of both future events and their associated timestamps, a complexity that distinguishes it from traditional sequence prediction problems (Mei and Eisner, 2017). This task can be further categorized into short-term forecasting, which focuses on predicting the next immediate event and its timestamp, and long-term forecasting, which aims to predict multiple future events along with their timestamps over an extended time horizon.

Temporal point processes (TPPs) have been widely explored in literature (Daley and Vere-Jones, 2007; Hawkes, 1971; Mei and Eisner, 2017; Shchur et al., 2021) to address CTES forecasting. TPPs model the likelihood of each event's instantaneous occurrence using an intensity function, which is computed based on the historical time-event data. Classical TPPs employ Hawkes processes (Laub et al., 2015) to define the intensity function ($\lambda(t)$) for a particular event type (k) as $\lambda_k(t) = \mu_k + \sum_{j:t_j<t} \psi_k(t-t_j)$ where μ_k is the base intensity for that event type and $\psi_k(\cdot)$ is a pre-specified decaying function (e.g., exponential function or power-law function). Intuitively, the intensity function shows that the intensity of an event diminishes with increasing time difference after the previous event of the same type. This indicates that each event occurrence increases the intensity of subsequent events, also referred to as self-excitation (Daley and Vere-Jones, 2007). However, this formulation lacks the flexibility needed for many applications due to its assumption of no interdependence between different types of events. Neural TPPs address this limitation by parameterizing the intensity function with Recurrent Neural Networks (RNNs), making it more adaptable to real-world datasets (Mei and Eisner, 2017). Recent advances have further improved these models through transformer-based architectures which are currently state-of-the-art approaches for modeling CTES with TPPs (Zuo et al., 2020; Yang et al., 2022). However, these approaches often face challenges in scenarios where the distribution of event occurrences does not align with the assumptions underlying intensity-based modeling (Shchur et al., 2021).

Traditional TPP models primarily focus on short-term forecasting. For long-term forecasting, they rely on the thinning algorithm (Lewis and Shedler, 1979) via auto-regressive event sampling which was shown to perform poorly (Deshpande et al., 2021). Xue et al. (2022) have tackled the long-term forecasting challenge in TPPs by generating multiple sequences and employing an energy function to select the most relevant one. However, it still relies on the TPP-based generation, constraining its overall performance. Recently, diffusion-based models have been proposed (Zeng et al., 2024) to overcome these shortcomings by directly learning the joint probability distribution of event types and inter-arrival times. However, these models require the number of events or time horizon to be specified in advance for the diffusion

process training. This constraint becomes problematic in real-world scenarios where the number of future events can vary significantly across sequences, potentially requiring separate models for sequences with varying number of expected events or different time horizons.

To address the limitations of existing models, we propose Kairos, a decoder-only transformer model designed to tackle the CTES forecasting problem using language modeling. Our approach transforms CTES data into a sequence of event and time tokens, reframing the task as a next-token prediction problem. This reformulation eliminates the need for implicit assumptions about the data distribution (as required by TPPs (Shchur et al., 2019)), offering a more flexible framework to model CTES. Kairos learns a joint representation of events and times, similar to diffusion models, but without relying on priors such as the number of events or the time horizon. Furthermore, we leverage recent advances in language modeling, such as top-k and top-p sampling (Holtzman et al., 2019; Radford et al., 2019) and the self-consistency principle (Wang et al., 2022), to enhance long-term forecasting. Kairos generates multiple diverse sequences and refines them to obtain the optimal sequence through consensus decoding (Mei et al., 2019), which helps mitigate error accumulation in long-term forecasting by identifying common patterns across multiple generations.

In summary, the contributions of our work are as follows:

- We propose Kairos, a novel model for CTES forecasting that represents a paradigm shift towards language modeling. By moving beyond intensity function-based formulation of TPPs and diffusion approaches which require a fixed prior, Kairos offers greater flexibility in capturing real-world scenarios.
- We introduce a unified tokenization approach to enable joint representation of time and event in a shared embedding space. This approach facilitates cross-modal learning between temporal and event information. Through structural analysis, we study how this unified representation enhances the model's ability to predict both time and event types accurately.
- We conduct comprehensive benchmarks on multiple large open-source datasets, demonstrating that Kairos outperforms state-of-the-art baselines for CTES forecasting. We further study the workings of the model by analysing the attention patterns and the impact of self consistency.

2. Related Work

TPPs have been widely used for CTES forecasting across various domains (Daley and Vere-Jones, 2007). While classical TPPs like Hawkes processes pioneered CTES forecasting, their intensity-based formulation limited their ability to model complex temporal dependencies. The advent of deep learning architectures catalyzed a shift towards incorporating neural networks to learn complex patterns in CTES, giving birth to neural TPPs. Initial RNN-based approaches (Du et al., 2016; Mei and Eisner, 2017) showed improvements over classical methods but struggled with long-term dependencies. Transformer-based neural TPPs (Yang

et al., 2022; Zuo et al., 2020) addressed this limitation, capturing both short-term and long-term dependencies in CTES modelling. These approaches were developed for next event-time forecasting and were extended to long-term forecasting using auto-regressive generation. The accumulation of errors in auto-regressive generation significantly impacts the models' performance on the long-term forecasting task. This limitation was addressed by recent works like HYPRO (Xue et al., 2022) which employ additional modules to mitigate accumulation of auto-regressive errors. However, neural TPPs remain constrained by their reliance on intensity functions, affecting their cross-domain adaptability (Shchur et al., 2021; Omi et al., 2019).

The limitations of intensity-based methods led to the development of non-intensity based approaches for CTES modeling. Lin et al. (2022) explored various generative approaches, like diffusion, variational inference and GANs for improving the predictive performance of TPPs. Recent advances include probabilistic denoising diffusion models for long-term forecasting (Lüdke et al., 2023) and their extension to marked sequences (Zeng et al., 2024). However, these diffusion-based approaches require pre-specified priors like forecasting horizon, limiting their effectiveness for modelling irregularly occurring events.

Language modeling's success in predicting sequential data has extended beyond natural language to various domains, including computer vision (Chen et al., 2020); protein sequencing (Jumper et al., 2020) and time series forecasting (Ansari et al., 2024). These models excel at learning complex patterns and long-range dependencies, with various decoding strategies like greedy sampling, top-p sampling (Holtzman et al., 2019), top-k sampling (Fan et al., 2018) . These decoding methods were further improved by Wang et al. (2022) by introducing the concept of self-consistency principle, which is based on the intuition that multiple reasoning paths are required for obtaining the correct answer for complex reasoning tasks.

3. Methodology

3.1. Mathematical Formulation

In CTES, we define a sequence of timestamp-event tuples as $S = \{(t_i, e_i)\}_{i=1}^{T}$, where t_i is the timestamp of the occurrence of event e_i. Each event e_i belongs to one of the K classes, $e_i \in \mathbb{E}$, with $|\mathbb{E}| = K$. The sequence of timestamps is monotonically increasing ($t_1 < t_2 < ... < t_T$) and the intervals between events can be irregular, i.e. there may exist $i > 0 : t_{i+1} - t_i \neq t_{i+2} - t_{i+1}$. While formulating the forecasting problem, we transition from using absolute timestamps to inter-event times for representing temporal data, driven by two key considerations: (1) the intensity function, fundamentally depends on the inter-event time, and (2) forecasting inter-event times typically results in a more stationary process than using timestamps (Yang et al., 2022).

Consequently, we redefine the sequence S as $S = \{(\tau_i, e_i)\}_{i=1}^{T}$, where $\tau_i = t_i - t_{i-1}$ for $i > 1$ and $\tau_1 = 0$. Formally, given a historical sequence $S_L = \{(\tau_i, e_i)\}_{i=1}^{L}$ with $L < T$, our primary goal is to learn a function f which predicts the sequence for next N events: $f(S_L) = \{(\tau_i, e_i)\}_{i=(L+1)}^{L+N}$

3.2. Model Details

We solve the CTES forecasting problem of predicting both the next event and inter-event time by transforming it into a next-token prediction (NTP) problem. Specifically, we transform the

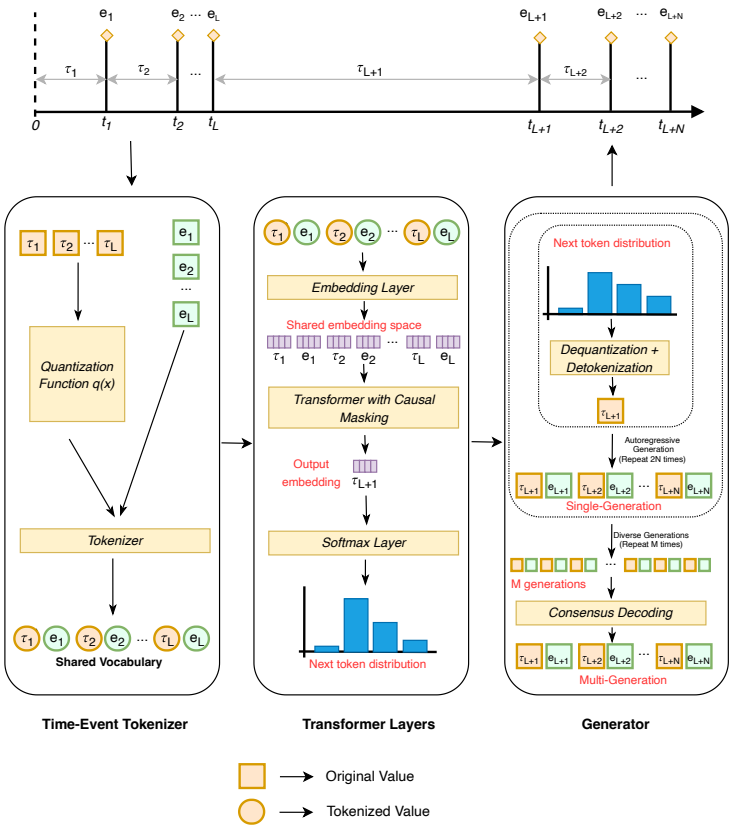

Figure 1: Architectural Overview of Kairos .

original sequence $S=\{(\tau_i,e_i)\}_{i=1}^{T}$ of length T into an alternating sequence of time intervals and event types: $\bar{S}=\tau_1,e_1,\tau_2,e_1,...,\tau_T,e_T$ of length $2T$.

Given $\bar{S}$, our goal is to learn a function f that predicts the probability of the next token u_j in the sequence by minimizing the following loss:

$$\mathcal{L}=\underset{\hat{f}\in F}{\operatorname{argmin}}\sum_{j=1}^{2T}[\delta_{u_j\in\mathbb{E}}\cdot l_{\mathbb{E}}(u_j,\hat{u}j)+\delta_{u_j\in\mathbb{T}}\cdot l_{\mathbb{T}}(u_j,\hat{u}_j)] \tag{1}$$

where $\hat{u}_j=f(u_j|u_1,...,u_{j-1})$, F represents the function set, out of which $\hat{f}$ is the optimal function, $\mathbf{1}$ is an indicator function, δ is the Kronicker delta function (The Kronicker delta function outputs 1 when the condition in the subscript is true and 0 otherwise), $l_{\mathbb{E}}$ and $l_{\mathbb{T}}$ are the categorical cross-entropy loss functions for event and time tokens respectively, and $[u_1,u_2,...,u_{j-1}]$ is the sequential stream of past tokens. Figure 1 illustrates the architecture of our proposed model, Kairos, which consists of three main components: Time-Event Tokenizer, Transformer Layers and Generator.

3.2.1. Time-Event Tokenizer

The first step involves pre-processing the input sequence and converting it into a series of discrete tokens. Kairos embeds both events and time in a shared embedding space. This design choice is aimed at enabling the model to jointly learn the complex relationship between events and time, rather than treating them as two separate streams of information. This architectural choice is further justified in Section 4.3.2.

Specifically, we transform the inter-event time, a continuous value, into the discrete domain. Inspired by previous works that use language modeling for time-series tasks (Ansari et al., 2024), we represent inter-event time in our sequence data as quantized buckets, which can enhance the model's ability to identify meaningful patterns, while reducing noise. Formally, we identify the maximum inter-event time (say τ_m) and generate B equal width bins , where B is a hyperparameter that determines the granularity of time discretization. The quantization function $q:\mathbb{R}\to\{0,1,2,...,B\}$ and dequantization function $d:\{0,1,2,...,B\}\to\mathbb{R}$ used by our model are defined as follows:

$$q(x)=\begin{cases}\text{round}\left(\frac{x\times B}{\tau_m}\right) & \text{if } 0\le x\le\tau_m\\ B & \text{if } x>\tau_m\end{cases} \qquad d(j)=\frac{j\times\tau_m}{B} \tag{2}$$

After quantizing the inter-event time, we obtain a completely discretized sequence. This sequence is tokenized using a unified vocabulary[1] $[\mathbb{E},\mathbb{T}]$, where $|\mathbb{E}|=K$ and $|\mathbb{T}|=B+1$.

3.2.2. Transformer Layers

Tokenized sequences are first passed through an embedding layer with a dimension of d_{model}, where d_{model} denotes the model's capacity. After embedding, the sequence is processed through several transformer layers with causal masking (Vaswani, 2017). Subsequently, an affine layer projects the transformer's output from $\mathbb{R}^{d_{\text{model}}}$ to $\mathbb{R}^{B+K+1}$. A softmax function is then applied to produce a probability distribution over the token universe.

3.2.3. Generator

We employ two strategies for the generator: single-generation and multi-generation.

Single-Generation: In this strategy, we use top-k and top-p sampling at each step. After generating the next token, the token is detokenized to retrieve either the event type or the time gap bin [2]. If the detokenized token corresponds to a time value, we apply the dequantization function (defined in Equation 2) to compute the inter-event time. To predict the next N events along with their respective time gaps, our model generates 2N tokens in an autoregressive manner, followed by detokenization and dequantization to construct the future time-event sequence. The model learns to predict the alternate sequence of time and event tokens, and any mismatch in the order is captured by the long-term forecasting metric.

1. This token universe does not consider special tokens used for training like padding token, start/end of sequence tokens etc.
2. We do not explicitly enforce the alternating pattern of times and events during generation which is a simple pattern learnt by language models from training data (Linzen et al., 2016)

Multi-Generation: Moving beyond the traditional sampling approaches such as top-k and top-p sampling which only generate a single possible sequence, we explore the broader paradigm of self-consistency that has emerged in recent literature (Wang et al., 2022) to increase the search space and reduce auto-regressive error accumulation. This involves generating multiple sequences from language models and then aggregating them to form a more consistent output. Specifically, we implement this methodology through consensus decoding (Mei et al., 2019). Kairos generates M different proposals using top-k and top-p sampling, denoted as $G = \{g_m\}_{m=1}^{M}$. Our goal is to produce a single generation $\hat{g} \in \bigcup_{m=1}^{M} g_m$ that achieves the minimum Bayes risk. The Bayes risk for a generation g is defined as $\sum_{m=1}^{M} w_m L(g, g_m)$, where w_m is the weight of sequence g_m given the input, and $L(g, g_m)$ denotes the loss of g with respect to g_m. We use the Optimal Transport Distance (OTD) (Mei et al., 2019) as the loss metric and employ the inverse of the perplexity (ρ_m^{-1}) for each generation as the weight metric (w_m). The perplexity of a generation $g_m = \{u_i\}_{i=1}^{|g_m|}$ is given by:

$$\rho_m = e^{-\frac{1}{|g_m|}\sum_{i=1}^{|g_m|}\log P(u_i|u_1,u_2,\ldots,u_{i-1})} \tag{3}$$

Consensus decoding (Mei et al., 2019) defines an algorithm using addition, deletion and edit to iteratively align each of the generations g_m to obtain $\hat{g}$, which achieves the minimum Bayes risk. In Section 4.4.2, we demonstrate how consensus decoding operates within Kairos.

Table 1: Performance comparison of different models on short-term forecasting task with following metrics: E_{Acc} (%) in 1st row (higher is better) and T_{RMSE} in 2nd row (lower is better). Best metrics in bold and second best metrics is underlined.

Model	Metrics	Taxi	Taobao	SO-2K	SO-400K	Retweets	Mean Rank
RMTPP	E_{Acc}	89.02 ± 0.90	43.57 ± 0.00	42.50 ± 0.00	17.60 ± 4.49	52.74 ± 2.32	3.5
	T_{RMSE}	0.371 ± 0.00	<u>0.134 ± 0.00</u>	1.374 ± 0.00	2.591 ± 0.00	<u>0.617 ± 0.01</u>	
NHP	E_{Acc}	<u>91.01 ± 0.16</u>	59.57 ± 0.00	42.50 ± 0.00	<u>30.74 ± 0.13</u>	59.44 ± 0.16	2.7
	T_{RMSE}	<u>0.371 ± 0.00</u>	<u>0.134 ± 0.00</u>	1.374 ± 0.00	2.595 ± 0.00	0.619 ± 0.00	
AttNHP	E_{Acc}	88.55 ± 0.9	43.63 ± 0.06	42.09 ± 2.98	20.17 ± 1.89	51.13 ± 2.28	4.1
	T_{RMSE}	0.435 ± 0.01	0.136 ± 0.00	<u>1.366 ± 0.00</u>	2.583 ± 0.01	0.640 ± 0.03	
THP	E_{Acc}	90.85 ± 0.40	<u>59.71 ± 0.29</u>	**44.26 ± 0.29**	25.74 ± 2.48	<u>60.09 ± 0.39</u>	2.3
	T_{RMSE}	<u>0.371 ± 0.00</u>	<u>0.134 ± 0.00</u>	1.374 ± 0.00	<u>2.566 ± 0.02</u>	0.619 ± 0.00	
Kairos	E_{Acc}	**91.08 ± 0.16**	**60.18 ± 0.37**	<u>43.94 ± 0.37</u>	**32.18 ± 0.25**	**60.49 ± 0.05**	**1.1**
	T_{RMSE}	**0.323 ± 0.00**	**0.131 ± 0.00**	**1.282 ± 0.00**	**2.338 ± 0.01**	**0.568 ± 0.01**	

4. Evaluation

4.1. Experimental Setup

We evaluate Kairos against state-of-the-art models on five datasets for both short-term and long-term forecasting. For fair comparison, we maintain comparable parameter counts across all models. Results show mean and standard deviation over five trials. Models were trained on an Nvidia-A10 GPU. Our model uses a two-layer decoder (2 attention heads, dim=16)

Table 2: Optimal Transport Distance (OTD) of different models on long-term forecasting task (lower is better). Best metric is in bold and second best metric is underlined.

Generation Strategies	Model	Taxi	Taobao	SO-2K	SO-400K	Retweets	Mean Rank
Single Generation	RMTPP	32.37 ± 0.04	41.54 ± 0.00	60.79 ± 0.00	45.17 ± 2.19	35.86 ± 0.10	8.0
	NHP	31.46 ± 0.03	38.83 ± 0.16	60.75 ± 0.00	40.26 ± 0.15	31.38 ± 0.61	6.4
	AttNHP	31.50 ± 0.25	36.72 ± 0.94	59.42 ± 0.68	30.11 ± 3.38	22.57 ± 1.89	5.0
	THP	31.64 ± 0.20	38.36 ± 0.23	60.54 ± 0.02	40.14 ± 1.41	33.63 ± 0.04	6.4
	Kairos (M=1)	17.64 ± 0.52	<u>32.99 ± 0.73</u>	37.37 ± 0.21	25.24 ± 0.09	16.76 ± 0.13	2.8
Multi-Generation	HYPRO (M=20)[3]	20.76	35.70	39.07	38.48	18.94	4.2
	CDiff (M=7)	<u>17.22 ± 0.51</u>	33.56 ± 0.20	<u>36.46 ± 0.43</u>	<u>20.37 ± 0.18</u>	<u>13.974 ± 0.17</u>	2.2
	Kairos (M=7)	**14.99 ± 0.19**	**28.89 ± 0.39**	**29.65 ± 0.11**	**14.98 ± 0.04**	**13.967 ± 0.13**	**1.0**

with PyTorch Lightning and HuggingFace transformers, trained with batch size 128, AdamW (lr=0.001), up to 100 epochs (early stopping patience=20). Generation parameters: top-k=20, top-p=0.95, temperature=1.0, OTD cost=1.5.

4.1.1. Datasets

We evaluate Kairos on five real-world datasets: Taobao (Alibaba, 2018) (user browsing, $K=17$, 3hr units, 51 avg. length, 1300/200/500 train/val/test), SO-2k (Leskovec and Krevl, 2014) (Q&A awards, $K=22$, 11-day units, 65 avg. length, 1400/400/400), SO-400k (StackExchange) (Q&A awards, $K=96$, 55-day units, 51 avg. length, 340K/80K/80K), Taxi (Whong, 2014) (NYC taxi events, $K=10$, 1hr units, 37 avg. length, 1400/200/400), and Retweets (Zhao et al., 2015) (retweet sequences, $K=3$, 1hr units, 154 avg. length, 96K/32K/32K). We used preprocessed versions for Taobao, SO-2k and Taxi from Xue et al. (2022).

4.1.2. Evaluation Metrics

For short-term forecasting (N=1 as defined in Section 3.1), we use Next Event Accuracy (E_{Acc}, higher is better) measuring prediction accuracy of the next event type, and Next Time RMSE (T_{RMSE}, lower is better) evaluating time prediction error. For long-term forecasting (N=20 event-timestamp pairs), we use Optimal Transport Distance (OTD), which measures the minimum cost of editing predicted sequences into ground-truth sequences, similar to edit distance (Miller et al., 2009) or dynamic time warping (Furtuna, 2008). Following Xue et al. (2022); Zeng et al. (2024), we report average OTD across cost constants $C=0.05,0.5,1,1.5,2,3,4$ (lower is better), with predictions trimmed beyond a time horizon equal to the forecast horizon multiplied by mean inter-event time.

4.1.3. Baselines

We benchmark against four Neural TPP-based models for short-term forecasting: RNN-based RMTPP (Du et al., 2016), LSTM based NHP (Mei and Eisner, 2017); and transformer-based models: THP (Zuo et al., 2020) and AttNHP (Yang et al., 2022). These have proven to be the state-of-the-art in short-term forecasting as benchmarked by Xue et al. (2024). For long-term forecasting with multiple generations, we compare Kairos (M=7) with HYPRO (Xue et al.,

3. Due to HYPRO's long computation time, we could only run one iteration on the reported datasets.

2022) and CDiff (Zeng et al., 2024). HYPRO (Hybridly normalized neural probabilistic model) combines an autoregressive base model with an energy function to discriminate amongst multiple generations. CDiff is the state-of-the-art diffusion based model for long term forecasting which utilizes joint probability distribution of types and inter-arrival times for multiple events to output future event-time sequence.

For long-term forecasting, baselines employ varying number of generations for benchmarking. Based on this, we evaluate our model in two settings: single-generation and multiple-generation. We compare the TPP-based baselines with Kairos in single generation setting. In multiple generation setting, we use the number of generations (M) as 20 and 7 for HYPRO and CDiff respectively, as specified in their papers, while for Kairos we use $M = 7$.

4.2. Comparison with baselines

4.2.1. Short-Term Forecasting

Table 1 compares our model with baseline methods across various datasets for short-term forecasting. Kairos achieves the highest mean rank, followed by THP, a TPP-based baseline that also utilizes a Transformer architecture. Kairos outperforms THP by 4.5% on E_{Acc} and 7.8% on T_{RMSE}. This improvement can be attributed to two key factors: the adoption of language modeling paradigm and the joint learning of both time and event representations. The language modeling approach helps with learning complex patterns in sequential data, while joint learning allows the model to leverage the underlying relation between temporal and event information. We present an empirical analysis of the contribution of both these changes in Section 4.3.2.

4.2.2. Long-Term Forecasting

Table 2 shows that Kairos outperforms other baselines in both single generation and multiple generation settings, notably outperforming HYPRO using a single generation, even when the latter relies on 20 generations. In multi-generation setting, Kairos outperforms the diffusion based baseline (CDiff) by 14.41%. The superior performance of our approach can be attributed to its ability to leverage the latest sampling strategies from the language modeling paradigm: (1) top-k and top-p sampling for single-generation, which retains high-probability predictions, mitigating exposure bias and ensuring coherence in long event sequences (Holtzman et al., 2019); and (2) usage of self consistency through consensus decoding, which smooths out uncertainties and mitigates outliers, resulting in robust predictions that more accurately align with the true probability distribution of future events and their timing. Further ablations in Section 4.3.1 highlight the importance of these sampling strategies for long-term generation whereas Section 4.4.2 tries to understand how consensus decoding reduces auto-regressive error accumulation.

4.3. Ablation Studies

In this section, we perform ablations to the study the impact of various design choices and structural components of Kairos . Furthermore, in the supplementary material, we study how utilizing additional compute by increasing number of generations and the model capacity can further lead to performance improvement.

Table 3: Design choices and sampling strategies comparison on SO-400K dataset.

Backbone	$E_{\text{Acc}}(\%)$	T_{RMSE}
Encoder-decoder Transformer	30.06	2.27
Decoder-only Transformer	32.18	2.34
State space model	30.84	2.32

Generation Strategy	Sampling Strategy	OTD
Single Generation	Greedy	41.85
	top-k and top-p	25.24
Multi-Generation	Simple Aggregation	24.57
	Consensus Decoding	14.98

4.3.1. Impact of Sampling Strategy on Long-Term Forecasting

We examine various sampling strategies for long-term generation and present our findings on the SO-400K dataset. In the single-generation setting, we compare the performance of greedy decoding with top-k and top-p sampling. For greedy decoding, the token with the highest probability is selected at each step. As shown in Table 3, top-k and top-p sampling outperforms greedy decoding by 39.6% in terms of OTD. This significant performance gap is consistent with findings from Holtzman et al. (2019) and Fan et al. (2018), which highlight that greedy decoding often favors generic and repetitive patterns.

In the multiple-generation setting, we implement a naive aggregation baseline to incorporate self-consistency across generations by computing mean values for time indices and applying majority voting for event indices across multiple generations. Table 3 shows that consensus decoding improves performance by 39% in terms of OTD over simple aggregation. This improvement is attributed to the fact that naive aggregation can result in degenerate sequences that violate conditional constraints, as noted in Wang et al. (2022).

4.3.2. Structural Analysis

A detailed analysis was conducted to understand the incremental benefits of transitioning to language modeling and jointly learning event and time representations. To evaluate this, we implemented a new model using the same decoder-only transformer but with separate embeddings for events and times. These embeddings are concatenated at each step, and predictions are made through two distinct heads: one for time and one for event. The tokenization mechanism and loss remain as described in Section 3. This model represents an intermediate state between the THP model and Kairos, incorporating language modeling without joint representation. Table 4 compares its performance with THP and Kairos on SO-400K dataset, showing that transitioning to language modeling improves E_{Acc} by 10.33% and reduces T_{RMSE} by 5.38%, with joint representation further improving E_{Acc} by 13.63% and reducing T_{RMSE} by 3.38%.

4.4. Understanding working of Kairos

This section provides an in-depth examination of the model's behaviour by studying the attention weights for time/event tokens and the role of consensus decoding in long-term forecasting.

4.4.1. Analysing attention patterns of Kairos

The attention weights of the Kairos model are analyzed to assess its ability to learn meaningful temporal and event relationships for CTES modeling. The initial analysis focuses on the types

Table 4: Structural Analysis of Kairos components.

Models	$E_{\text{Acc}}(\%)$	T_{RMSE}
Intensity Loss (THP)	25.74	2.566
NTP with Separate Representation	28.40	2.428
NTP with Joint Representation (Kairos)	32.27	2.346

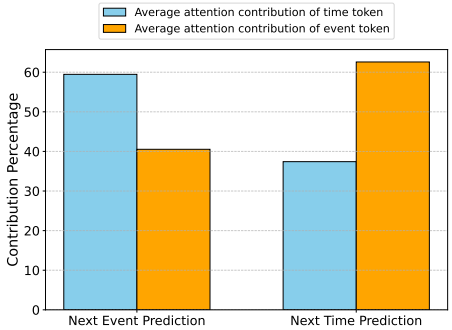

Figure 2: Attention contribution of event and time tokens on each other on SO-400K dataset.

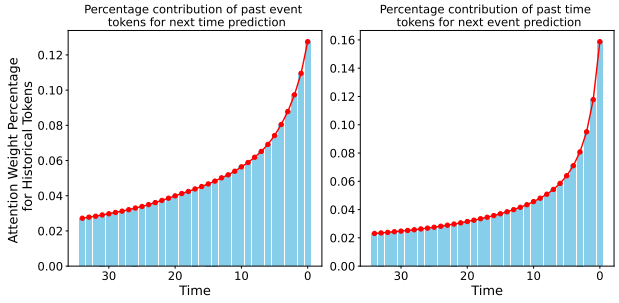

Figure 3: Percentage contribution of historical tokens to future predictions.

of tokens the model attends to while predicting event and time tokens. To understand the types of tokens the model attends to, we aggregate the attention scores separately for event and time tokens at each layer and head, as suggested in Sharma et al. (2022). We then aggregate the total attention scores on basis of type of token attended to and type of token predicted. This score is then normalized with token's total number of occurrences, giving us the average attention contribution of time and event token on each other. Figure 2 shows that, while predicting an event token, Kairos applies 59.46% attention to time tokens, and while predicting time, it relies more on event tokens (62.59% attention). This underscores the importance of jointly learning event and time representations and demonstrates how Kairos enables cross-learning between them.

Additionally, we investigate the positional dependency on past tokens when Kairos predicts a token. To quantitatively understand the positional dependence on past tokens, we aggregate the attention weights on basis of past indices of event token while predicting time token and vice-versa. These attention weights are then normalized with total number of occurrences to provide percentage contribution of historical tokens to future predictions. Figure 3 reveals an exponential decay in the influence of past tokens, resembling the decay function in Hawkes processes. This finding indicates that Kairos captures temporal patterns akin to traditional TPP models without relying on intensity-based loss formulations.

4.4.2. Examining the Impact of Consensus Decoding on Long-Term Forecasting Improvement

First the impact of consensus decoding is analyzed as a function of the forecast horizon length. Figure 4 illustrates that as the forecast horizon length increases, the performance gap between Kairos single-generation and Kairos multiple-generation widens. This occurs because the likelihood of errors in single-generation predictions grows with longer horizons, whereas consensus decoding leverages information from multiple generations to correct some of these errors.

To show this at a more granular level, multiple-generations and the consensus decoding output are analyzed for a sample of users. The 550-day forecast horizon is divided into 36-day bins, and the Intersection over Union (IoU) between predicted and ground truth events is calculated for each bin. Figure 5 shows that consensus decoding consistently achieves equal or higher IoU scores compared to individual generations, confirming that self-consistency across multiple generations effectively reduces errors.

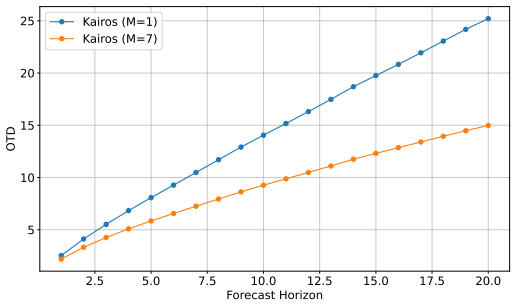

Figure 4: Comparison of Kairos (single generation) and Kairos (multi-generation) at different forecast horizons.

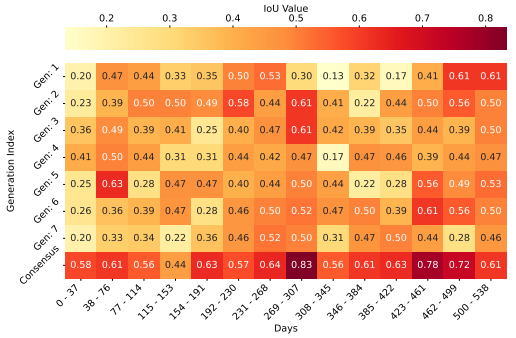

Figure 5: Comparison of Consensus Generation with each individual generation at different timesteps.

4.5. Inference efficiency

Table 5 compares the inference time of Kairos with various baseline models across multiple datasets, for both single (M=1) and multi-generation (M=7) settings. All models are benchmarked on Nvidia L40s GPUs. In the single-generation setting, Kairos achieves comparable inference times to other baselines while delivering superior performance (as shown in Table 1). More notably, in the multi-generation setting, Kairos demonstrates better efficiency, running approximately 10 times faster than CDiff and HYPRO on the SO-400k and Retweets datasets. This speed advantage highlights that our optimizations for long-term forecasting maintain computational efficiency, unlike other long-term baselines such as CDiff, which requires multiple denoising steps that significantly increase computation time. Our analysis on the SO-400k dataset reveals that consensus decoding consumes only 0.025 minutes per 1,000 samples out

Table 5: Comparison of Inference time (minutes per 1000 samples) and number of parameters

Model	Taxi	SO-400k	Retweets
THP	0.06/26.3K	0.19/37.4K	0.09/25.4K
NHP	0.21/59.1K	5.93/70.1K	14.70/58.2K
RMTPP	0.05/2.8K	0.07/8.4K	0.07/2.4K
AttNHP	0.13/11.9K	1.01/49.8K	2.28/11.3K
HYPRO[4](M=20)	28.67/19.3K	31.67/11.1K	57.67/6.5K
CDiff (M=7)	0.51/14.5K	6.58/22.2K	10.16/18.5K
Kairos (M=1)	0.07/26.7K	0.12/46.4K	0.18/27.5K
Kairos (M=7)	0.30/26.7K	0.62/46.4K	0.91/27.5K

of the total processing time of 0.617 minutes per 1,000 samples. This demonstrates that concensus decoding is relatively efficient compared the overall computational requirements and utilizes less than 5% of the total time.

5. Conclusion and Future Work

In this paper, we introduce Kairos, a novel model for CTES forecasting that shifts away from traditional TPP and diffusion-based approaches towards language modeling. The model represents events and time in a shared embedding space as alternating sequences of inter-arrival times and events, leveraging decoder-only transformer-based language models to learn the joint representation between time and event tokens. The paradigm shift makes the formulation flexible and allows us to utilize language modeling enhancements like top-k and top-p sampling. We demonstrate superior performance compared to existing baselines: for short-term forecasting, we achieve a 4.5% improvement in event accuracy (E_{Acc}) and a 7.8% reduction in time root mean square error (T_{RMSE}) compared to THP which is the current state-of-the-art model for short-term forecasting. We further extend the concept of self-consistency to CTES domain by utilizing consensus decoding which improves long-term forecasting performance by reducing the accumulation of auto-regressive errors. We demonstrate in the multiple generation setting, we are able to outperform CDiff by 14.41% in long-term forecasting.

Our structural analysis reveals how Kairos improves upon TPP-based baselines. Through examination of attention weights, we demonstrate the model's ability to learn meaningful temporal and event patterns. Additionally, we show how consensus decoding with multiple generations helps mitigate auto-regressive error accumulation, thereby improving long-term forecasting performance.

While our current implementation uses cross-entropy loss to jointly train the model on time and event tokens, incorporating time-specific losses such as ordinal loss could further enhance the model's temporal pattern learning capabilities. Although we currently employ consensus decoding to leverage additional computational resources, recent advances in test-time computation could enable mid-generation interventions to further improve performance.

4. Batch size=1 for HYPRO due to unavailability of batched inference in official repository.

References

Alibaba. User behavior data from taobao for recommendation, 2018. URL https://tianchi.aliyun.com/dataset/dataDetail?dataId=649.

Abdul Fatir Ansari, Lorenzo Stella, Caner Turkmen, Xiyuan Zhang, Pedro Mercado, Huibin Shen, Oleksandr Shchur, Syama Sundar Rangapuram, Sebastian Pineda Arango, Shubham Kapoor, et al. Chronos: Learning the language of time series. *arXiv preprint arXiv:2403.07815*, 2024.

Mark Chen, Alec Radford, Rewon Child, Jeffrey Wu, Heewoo Jun, David Luan, and Ilya Sutskever. Generative pretraining from pixels. In *International conference on machine learning*, pages 1691–1703. PMLR, 2020.

Daryl J Daley and David Vere-Jones. *An introduction to the theory of point processes: volume II: general theory and structure*. Springer Science & Business Media, 2007.

Prathamesh Deshpande, Kamlesh Marathe, Abir De, and Sunita Sarawagi. Long horizon forecasting with temporal point processes. In *Proceedings of the 14th ACM international conference on web search and data mining*, pages 571–579, 2021.

Nan Du, Hanjun Dai, Rakshit Trivedi, Utkarsh Upadhyay, Manuel Gomez-Rodriguez, and Le Song. Recurrent marked temporal point processes: Embedding event history to vector. In *Proceedings of the 22nd ACM SIGKDD international conference on knowledge discovery and data mining*, pages 1555–1564, 2016.

Angela Fan, Mike Lewis, and Yann Dauphin. Hierarchical neural story generation. *arXiv preprint arXiv:1805.04833*, 2018.

Titus Furtuna. Dynamic programming algorithms in speech recognition. *Informatica Economica Journal*, XII, 01 2008.

Alan G Hawkes. Spectra of some self-exciting and mutually exciting point processes. *Biometrika*, 58(1):83–90, 1971.

Sergio Hernández, Pedro Álvarez, Javier Fabra, and Joaquín Ezpeleta. Analysis of users' behavior in structured e-commerce websites. *IEEE Access*, 5:11941–11958, 2017. doi: 10.1109/ACCESS.2017.2707600.

Ari Holtzman, Jan Buys, Li Du, Maxwell Forbes, and Yejin Choi. The curious case of neural text degeneration. *arXiv preprint arXiv:1904.09751*, 2019.

John Jumper, Richard Evans, Alexander Pritzel, Tim Green, Michael Figurnov, Kathryn Tunyasuvunakool, Olaf Ronneberger, Russ Bates, Augustin Žídek, Alex Bridgland, et al. High accuracy protein structure prediction using deep learning. *Fourteenth critical assessment of techniques for protein structure prediction (abstract book)*, 22(24):2, 2020.

Patrick J. Laub, Thomas Taimre, and Philip K. Pollett. Hawkes processes, 2015. URL https://arxiv.org/abs/1507.02822.

Jure Leskovec and Andrej Krevl. SNAP Datasets: Stanford large network dataset collection. http://snap.stanford.edu/data, June 2014.

P. A. W Lewis and G. S. Shedler. Simulation of nonhomogeneous poisson processes by thinning. *Naval Research Logistics Quarterly*, 26(3):403–413, 1979. doi: https://doi.org/10.1002/nav.3800260304. URL https://onlinelibrary.wiley.com/doi/abs/10.1002/nav.3800260304.

Haitao Lin, Lirong Wu, Guojiang Zhao, Pai Liu, and Stan Z Li. Exploring generative neural temporal point process. *arXiv preprint arXiv:2208.01874*, 2022.

Tal Linzen, Emmanuel Dupoux, and Yoav Goldberg. Assessing the ability of lstms to learn syntax-sensitive dependencies, 2016. URL https://arxiv.org/abs/1611.01368.

David Lüdke, Marin Biloš, Oleksandr Shchur, Marten Lienen, and Stephan Günnemann. Add and thin: Diffusion for temporal point processes. *Advances in Neural Information Processing Systems*, 36:56784–56801, 2023.

Hongyuan Mei and Jason M Eisner. The neural hawkes process: A neurally self-modulating multivariate point process. *Advances in neural information processing systems*, 30, 2017.

Hongyuan Mei, Guanghui Qin, and Jason Eisner. Imputing missing events in continuous-time event streams. In *International Conference on Machine Learning*, pages 4475–4485. PMLR, 2019.

Frederic P. Miller, Agnes F. Vandome, and John McBrewster. *Levenshtein Distance: Information theory, Computer science, String (computer science), String metric, Damerau?Levenshtein distance, Spell checker, Hamming distance*. Alpha Press, 2009. ISBN 6130216904.

Takahiro Omi, Kazuyuki Aihara, et al. Fully neural network based model for general temporal point processes. *Advances in neural information processing systems*, 32, 2019.

Alec Radford, Jeffrey Wu, Rewon Child, David Luan, Dario Amodei, Ilya Sutskever, et al. Language models are unsupervised multitask learners. *OpenAI blog*, 1(8):9, 2019.

Rishab Sharma, Fuxiang Chen, Fatemeh Fard, and David Lo. An exploratory study on code attention in bert. In *Proceedings of the 30th IEEE/ACM International Conference on Program Comprehension*, pages 437–448, 2022.

Oleksandr Shchur, Marin Biloš, and Stephan Günnemann. Intensity-free learning of temporal point processes. *arXiv preprint arXiv:1909.12127*, 2019.

Oleksandr Shchur, Ali Caner Türkmen, Tim Januschowski, and Stephan Günnemann. Neural temporal point processes: A review. *arXiv preprint arXiv:2104.03528*, 2021.

StackExchange. Stack exchange data explorer. URL https://data.stackexchange.com/.

A Vaswani. Attention is all you need. *Advances in Neural Information Processing Systems*, 2017.

Lu Wang, Wei Zhang, Xiaofeng He, and Hongyuan Zha. Supervised reinforcement learning with recurrent neural network for dynamic treatment recommendation, 2018.

Xuezhi Wang, Jason Wei, Dale Schuurmans, Quoc Le, Ed Chi, Sharan Narang, Aakanksha Chowdhery, and Denny Zhou. Self-consistency improves chain of thought reasoning in language models. *arXiv preprint arXiv:2203.11171*, 2022.

Chris Whong. Foiling nyc's taxi trip data, Mar 2014. URL https://chriswhong.com/open-data/foil_nyc_taxi/.

Siqiao Xue, Xiaoming Shi, James Zhang, and Hongyuan Mei. Hypro: A hybridly normalized probabilistic model for long-horizon prediction of event sequences. *Advances in Neural Information Processing Systems*, 35:34641–34650, 2022.

Siqiao Xue, Xiaoming Shi, Zhixuan Chu, Yan Wang, Hongyan Hao, Fan Zhou, Caigao Jiang, Chen Pan, James Y. Zhang, Qingsong Wen, Jun Zhou, and Hongyuan Mei. Easytpp: Towards open benchmarking temporal point processes. 2024. URL https://arxiv.org/abs/2307.08097.

Chenghao Yang, Hongyuan Mei, and Jason Eisner. Transformer embeddings of irregularly spaced events and their participants. *CoRR*, abs/2201.00044, 2022. URL https://arxiv.org/abs/2201.00044.

Shuang-Hong Yang, Bo Long, Alex Smola, Narayanan Sadagopan, Zhaohui Zheng, and Hongyuan Zha. Like like alike: joint friendship and interest propagation in social networks. In *Proceedings of the 20th international conference on World wide web*, pages 537–546, 2011.

Mai Zeng, Florence Regol, and Mark Coates. Interacting diffusion processes for event sequence forecasting, 2024. URL https://arxiv.org/abs/2310.17800.

Qingyuan Zhao, Murat A. Erdogdu, Hera Y. He, Anand Rajaraman, and Jure Leskovec. SEISMIC: A self-exciting point process model for predicting tweet popularity. *CoRR*, abs/1506.02594, 2015. URL http://arxiv.org/abs/1506.02594.

Simiao Zuo, Haoming Jiang, Zichong Li, Tuo Zhao, and Hongyuan Zha. Transformer hawkes process. *CoRR*, abs/2002.09291, 2020. URL https://arxiv.org/abs/2002.09291.