Towards Out-of-Distribution Sequential Event Prediction: A Causal Treatment

Chenxiao Yang[1], **Qitian Wu**[1], **Qingsong Wen**[2], **Zhiqiang Zhou**[2], **Liang Sun**[2], **Junchi Yan**[1*]
[1]Department of Computer Science and Engineering, Shanghai Jiao Tong University
[2]DAMO Academy, Alibaba Group
{chr26195,echo740,yanjunchi}@sjtu.edu.cn,
{qingsong.wen,zhouzhiqiang.zzq,liang.sun}@alibaba-inc.com

Abstract

The goal of sequential event prediction is to estimate the next event based on a sequence of historical events, with applications to sequential recommendation, user behavior analysis and clinical treatment. In practice, the next-event prediction models are trained with sequential data collected at one time and need to generalize to newly arrived sequences in remote future, which requires models to handle temporal distribution shift from training to testing. In this paper, we first take a data-generating perspective to reveal a negative result that existing approaches with maximum likelihood estimation would fail for distribution shift due to the latent context confounder, i.e., the common cause for the historical events and the next event. Then we devise a new learning objective based on backdoor adjustment and further harness variational inference to make it tractable for sequence learning problems. On top of that, we propose a framework with hierarchical branching structures for learning context-specific representations. Comprehensive experiments on diverse tasks (e.g., sequential recommendation) demonstrate the effectiveness, applicability and scalability of our method with various off-the-shelf models as backbones.

1 Introduction

Real-world problem scenarios are flooded with sequential event data consisting of chronologically arrived events which reflect certain behaviors, activities or responses. A typical example is user activity prediction [55, 12, 61, 20, 45, 63] that aims to harness user's recent activities to estimate future ones, which could help downstream target advertisement in online platforms (e.g., e-commerce or social networks). Predicting future events also plays a central role in some real situations such as epidemic control [43] and clinical treatment [3] for promoting social welfare.

A common nature of (sequential) event prediction lies in the different time intervals within which training and testing data are generated. Namely, models trained with data collected at one time are supposed to predict events in the future [58, 36, 41] where the underlying data-generating distributions may have gone through variation due to environmental changes. However, most existing approaches [20, 14, 37, 24, 45, 39, 19, 18] overlook this issue in both problem formulation and empirical evaluation, which may leave the real problems under-resolved with model mis-specification, and result in over-estimation for model performance on real data.

As a concrete example in sequential recommenders [11], the data-generating distribution for user clicking behaviors over items is highly dependent on user preferences that are normally correlated

*The SJTU authors are also with MoE Key Lab of Artificial Intelligence, SJTU. Junchi Yan is the correspondence author who is also with Shanghai AI Laboratory.

36th Conference on Neural Information Processing Systems (NeurIPS 2022).

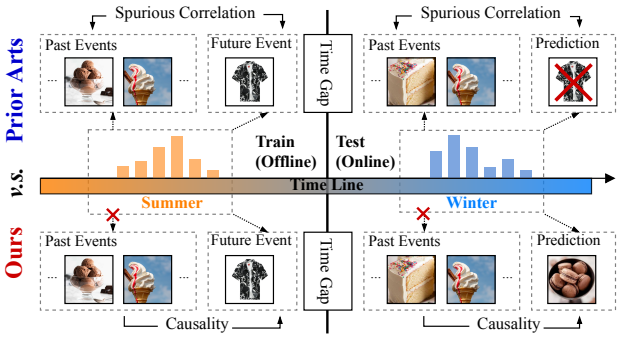


Figure 1: A toy example in recommendation. Prior art would spuriously correlate non-causal items ('ice cream' and 'beach T-shirt') and produce undesired results under a new environment in the future. Our approach endeavors to alleviate the issue via counteracting the effects of contexts (seasons).

with some external factors, like the dynamic fashion trends in different years [13] or seasonal influences on item popularity [44]. These highly time-sensitive external factors would induce distinct user preferences leading to different behavioral data as time goes by. This issue could also partially explain the common phenomenon of model performance drop after adaptation from offline to online environments [41, 36, 22]. Nevertheless, handling distribution shift in sequential event data poses several non-trivial challenges. **First**, the temporal shift requires models' capability for out-of-distribution (OoD) generalization [30], i.e., extrapolating from training environments to new unseen environments in remote future. Prior art that focus on model learning and evaluation over in-distribution data may yield sub-optimal results on OoD testing instances. **Second**, as mentioned before, there exist external factors that impact the generation of events. These external factors, which we term as *contexts*, are unobserved in practice. To eliminate their effects, one may also need the latent distributions that characterize how the contexts affect events generation. Unfortunately, such information is often inaccessible due to constrained data collection and privacy issues, which requires the models to learn from pure observed sequences.

1.1 Our Contributions

To resolve these difficulties, in this paper, we adopt a generative perspective to investigate the *temporal distribution shift* problem and propose a new variational context adjustment approach with instantiations to solve the issue for sequential event prediction.

A generative perspective on temporal distribution shift. We use proof-of-concept Structural Causal Models (SCMs) to characterize the dependency among contexts, historical sequences and labels (i.e., next event type) in terms of both data generation and model prediction. We show that contexts essentially act as a confounder, which leads the model to leverage spurious correlations and fail to generalize to data from new distributions. See Fig. 1 for a toy example to illustrate the issue.

Variational context adjustment. We propose a variational context adjustment approach to resolve the issue. The resulting new learning objective has two-fold effects: 1) helping to uncover the underlying relations between latent contexts and sequences in a data-driven way, and 2) canceling out the confounding effect of contexts to facilitate our model exploring the true causality of interest (as illustrated in Fig. 1). We also propose a mixture of variational posteriors to approximate context prior via randomly simulating pseudo input sequences.

Instantiation by a flexible framework. We propose a framework named as CaseQ to instantiate ingredients in the objective, which could combine with most off-the-shelf sequence backbone models. To accommodate temporal patterns under different environments, we devise a novel hierarchical branching structure for learning context-specific representations of sequences. It could dynamically evolves its architecture to adapt for variable contexts.

Empirical results. We carry out comprehensive experiments on three sequential event prediction tasks with valuation protocols designed for testing model performance under temporal distribution shift. Specifically, when we enlarge time gap between training and testing data, CaseQ can alleviate performance drop by 47.77% w.r.t. Normalized Discounted Cumulative Gain (NDCG) and 35.73% w.r.t. Hit Ratio (HR) for sequential recommendation, which shows its robustness against temporal distribution shift. The implementation codes will be made publicly accessible upon publication.

1.2 Related Works

Sequential Event Prediction [25] aims to predict the next event given a historical event sequence with applications including item recommendation in online commercial platforms, user behavior prediction in social networks, and epidemic control in real situations. Early works [39, 19, 18] rely on Markov chains and Bayesian networks to learn event associations. Later, deep learning based methods are proposed to capture non-linear temporal dynamics using CNN [46], RNN [20, 27], self-attention [24, 51], etc. These models are usually trained offline using event sequences collected during a certain period of time. Due to temporal distribution shifts, they may not generalize well to online environments [41, 36, 22]. Our work develops a principled approach for enhancing sequential event prediction with better robustness w.r.t. distribution shifts via pursing causality behind data.

Out-of-Distribution Generalization [30, 4, 31, 62] deals with distinct training and testing distributions. It is a common yet challenging problem, and has received increasing attention due to its significance [26, 42, 52, 54]. For event prediction, distribution shift inherently exists because of different time intervals from which training and testing data are generated. Most existing sequence learning models assume that data are independent and identically distributed [20, 14, 37]. Despite various approaches of OoD generalization designed for machine learning problems in other research areas (e.g., vision and texts), it still remains under-explored for sequential event prediction and user behavior modeling.

Causal Inference [33, 35] is a fundamental way to identity causal relations among variables, and pursuit stable and robust learning and inference. It has received wide attention and been applied in various domains e.g., computer vision [56, 28], natural language processing [40, 32] and recommender systems [57, 1]. Some existing causal frameworks for user behavior modeling [38, 16, 59, 60, 17] aim to extract causal relations based on observed or predefined patterns, yet they often require domain knowledge or side information for guidance and also do not consider temporal distribution shift. [1, 2, 6, 49] adopt counterfactual learning to overcome the effect of an ad-doc bias in recommendation task (e.g., exposure bias, popularity bias) or mitigate clickbait issue, whereas they do not focus on modeling sequential events and are still based on MLE as learning objectives.

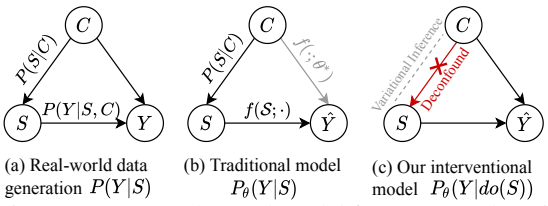
2 Problem and Model Formulation

We denote $\mathcal{X} = \{1, 2, \cdots, M\}$ as the space of event types, and assume each event is assigned with an event type $x_i \in \mathcal{X}$. Events occurred in chronological order consist of a sequence $\mathcal{S} = \{x_1, x_2, \cdots, x_{|\mathcal{S}|}\}$. As mentioned before, the data distribution is normally affected by time-dependent external factors, i.e., *context* c. We use S, Y, $\hat{Y}$ and C to denote the random variables of historical event sequence $\mathcal{S}$, ground-truth next event y, predicted next event $\hat{y}$ and context c, respectively. The data distribution can be characterized as $P(S, Y|C) = P(S|C)P(Y|S, C)$.

Problem Formulation. Given training data $\{(\mathcal{S}_i, y_i)\}_{i=1}^{N}$ generated from data distributions with $(\mathcal{S}_i, y_i) \sim P(S, Y|C = c_{tr}^{(i)})$, where $c_{tr}^{(i)}$ denotes the specific context when the i-th training sample is generated, we are to learn a prediction model $\hat{y}_i = f(\mathcal{S}_i; \theta)$ that can generalize to testing data $\{(\mathcal{S}_j, y_j)\}_{j=1}^{N'}$ from new distributions with $(\mathcal{S}_j, y_j) \sim P(S, Y|C = c_{te}^{(j)})$, where $c_{te}^{(j)}$ denotes the specific context when the j-th testing sample is generated. The distribution shift stems from different contexts that change over time, which we call *temporal distribution shift* in this paper.

2.1 Understanding the Limitations of Maximum Likelihood Estimation

Most of existing approaches target maximizing the likelihood $P_\theta(y|\mathcal{S})$ as the objective for optimization. Here we use $P_\theta(\cdot)$ to denote the distribution induced by prediction model f_θ. Based on the definitions, we can build two Structural Causal Models (SCMs) that interpret the causal relations among 1) C, S, Y (given by data-generating process) and 2) C, S and $\hat{Y}$ (given by model learning), as shown in



(a) Real-world data generation $P(Y|S)$ (b) Traditional model $P_\theta(Y|S)$ (c) Our interventional model $P_\theta(Y|do(S))$

Figure 2: Structural causal model for sequence learning.

Fig. 2(a) and (b). For Fig. 2(a), the three causal relations are given by the definitions for data generation, i.e., $P(S, Y|C) = P(S|C)P(Y|S, C)$. We next illustrate the rationales behind the other two causal relations $S \rightarrow \hat{Y}$ and $C \rightarrow \hat{Y}$.

$\boldsymbol{S \rightarrow \hat{Y}}$: This relation is induced by the prediction model $\hat{y} = f(\mathcal{S}; \theta)$ that takes a historical event sequence $\mathcal{S}$ as input and outputs the prediction for next event $\hat{y}$. The relation from $\mathcal{S}$ to $\hat{y}$ is deterministic given fixed model parameters θ.

$\boldsymbol{C \rightarrow \hat{Y}}$: This relation is implicitly embodied in the learning process that optimizes the model parameters with a given dataset collected at one time. By our definition, the training dataset is generated from a latent distribution affected by context c_{tr} and the MLE algorithm yields

$$\theta^* = \arg\min_{\theta} \mathbb{E}_{(\mathcal{S},y)\sim P(S,Y|C=c_{tr})}[l\left(f(\mathcal{S};\theta), y\right)], \tag{1}$$

where $l(\cdot, \cdot)$ denotes a certain loss function, e.g., cross-entropy. This indicates that the learned model parameters θ^* is dependent on the distribution of c_{tr}. Also, due to the fact $\hat{y} = f(\mathcal{S}; \theta)$, we conclude the proof for relation from C to $\hat{Y}$. In short, intuitive explanations for such a causal relation lie in two facts: 1) C affects the generation of data used for model training, and 2) $\hat{Y}$ is the output of the trained model given input sequence S.

Confounding Effect of C. The key observation is that C acts as the confounder in both Fig. 2(a) and (b), which could play a crucial role leading to undesirable testing performance of MLE-based approaches once the distribution is shifted. As implied by the causal relations in Fig. 2(a), there exists partial information in S that is predictive for Y yet highly sensitive to C, a usually fast-changing variable in real-world scenarios. As a result, the correlation between S and Y in previous contexts may become spurious in future ones. This also explains the failure of MLE-based models for generalizing to data from new distributions, according to the similar causal pattern in Fig. 2(b).

We can reuse the toy example in Fig. 1 as an intuitive interpretation for the failure. The 'summer' season (a context) acts as a confounder that correlates buying 'ice cream' (a historical event) and buying 'T-shirt' (a label), between which exists no obvious causal relation. However, the model would 'memorize' their correlation and tend to improperly recommend 'T-shirt' in the 'winter' season (new context). Whereas in fact the user purchases ice cream in winter most possibly because he/she is a dessert lover, in which case recommending other desserts would be a better decision.

Intervention. To address the confounding effect of C and endow the model with robustness to temporal distribution shift, we propose to target model learning with $P_\theta(Y|do(S))$ instead of the conventional $P_\theta(Y|S)$. As shown in Fig. 5(c), the do-operator cuts off the arrow (i.e., causal relation) coming from C to S, which essentially simulates an ideal data-generating process where sequences are generated independently from contexts. This operation blocks the backdoor path $S \leftarrow C \rightarrow \hat{Y}$ that spuriously correlates S and Y, and enables the model to learn the desired causal relation $S \rightarrow Y$ which is invariant to environmental change.

2.2 Variational Context Adjustment

An ideal way to compute $P_\theta(Y|do(S))$ is to carry out randomized controlled trial (RCT) [34] by recollecting data from a prohibitively large quantity of random samples under any possible context, which is infeasible since we could neither control the environment nor collect data in the future. Fortunately, there exists a statistical estimation of $P_\theta(Y|do(S))$ by leveraging backdoor adjustment [34], wherein the confounder C is stratified into discrete pieces $\mathcal{C} = \{c_i\}_{i=1}^{|\mathcal{C}|}$. By using basic rules induced by the do-operator (see derivation in Appendix A), we have:

$$P_\theta(Y|do(S)) = \sum_{i=1}^{|\mathcal{C}|} P_\theta(Y|S, C = c_i)P(C = c_i). \tag{2}$$

Intuitively, the backdoor adjustment approximates the ideal situation via enumerating c_i according to the context prior $P(C)$ independent from input sequences, which could counteract the effect of C on the generation of S. Nevertheless, optimizing with Eq. (2) is intractable since c_i is unobserved and its prior distribution $P(C)$ is unknown. Furthermore, computing Eq. (2) requires the relation between contexts and generated sequences, which is implicit in the data-generating process behind the data.

To address the difficulty for targeting Eq. (2), we introduce a variational distribution $Q(C|S)$ as the estimation for latent contexts given input sequences. By treating C as a latent variable and using variational inference technique, we can obtain the following tractable evidence lower bound (ELBO) as learning objective (see derivation in Appendix B):

$$\begin{aligned} &\log P_\theta(Y|do(S=\mathcal{S})) \\ \geq &\mathbb{E}_{c\sim Q(C|S=\mathcal{S})}\left[\log P_\theta(Y|S=\mathcal{S}, C=c)\right] - \mathcal{D}_{KL}\left(Q(C|S=\mathcal{S})\|P(C)\right), \end{aligned} \tag{3}$$

where the last step is given by Jensen's Inequality and the equality holds if and only if $Q(C|S)$ exactly fits the true posterior $P(C|S,Y)$, which suggests it successfully uncovers the latent context from observed data. The first component in Eq. (3) is the negative reconstruction error (i.e. prediction error). The second component is the Kullback–Leibler (KL) divergence of the variational distribution and context prior distribution. Like [29], we could re-write the second term as a summation of the entropy of $Q(C|S)$, i.e., $\mathbb{H}[Q(C|S=\mathcal{S})]$ and the cross-entropy between $Q(C|S)$ and the prior distribution, i.e., $\mathbb{H}[Q(C), P(C)]$. The former enforces high variance of contexts for each event sequence, and the latter aligns the aggregated posterior with the prior.

Learnable Prior via Mixture of Posteriors. The distribution $P(C)$ characterizes the true context prior in real world. It serves to regularize $Q(C|S)$ through the KL term in Eq. (3). Choosing an appropriate $P(C)$ is important yet challenging. One straightforward solution is to adopt a pre-defined prior [8, 56], e.g., uniform distribution. However, such a simplistic distribution may not reflect the true context prior, and potentially lead to over-regularization [5]. An alternative way is to estimate the prior by computing the average of the variational posterior [21], i.e., $P(C) \approx 1/N \sum_{i=1}^{N} Q(C|S=\mathcal{S}_i)$. However, such a method is computational expensive [47] and would often result in biased estimation given a limited quantity of training data collected within a certain time interval. Inspired by [47] using mixture of Gaussian as a flexible and learnable prior, we propose to use a mixture of pseudo variational posteriors as an estimation:

$$\hat{P}(C) = \frac{1}{R}\sum_{j=1}^{R} Q(C|S=\mathcal{S}'_j). \tag{4}$$

where $\mathcal{S}'_j$ is a randomly generated pseudo event sequence and we set $R \ll N$ to reduce the computational cost. Note that the variational distribution $Q(C|S)$ is given by the prediction model (see details in Section 3). Therefore, the prior estimation by Eq. (4) is a general reflection of how the model recognizes each type of context given uninformative inputs. Also, the estimated prior is learned with the model in fully data-driven manner.

3 Model Instantiations

We next parameterize the components in Eq. (3), i.e., $P_\theta(Y|S,C)$ and $Q(C|S)$, and propose a flexible framework named as CaseQ by taking 'cas' from 'causal' and 'seq' from 'sequence'.

Event Embedding. Given an event sequence $\mathcal{S} = \{x_1, x_2, \cdots, x_t\}$ as input, we use an embedding to represent each type of event. We consider a global embedding matrix $\mathbf{H}_x \in \mathbb{R}^{M\times d}$ to map each type of event into a d-dimensional embedding space:

$$\mathbf{h}^1_m = OneHot(x_m)^\top \mathbf{H}_x, \tag{5}$$

where M denotes the number of event types, $OneHot(\cdot) : \mathbb{Z}^+ \to \{0,1\}^M$ transforms an event into a one-hot column vector, and $\mathbf{h}^1_m \in \mathbb{R}^d$ denotes the representation for an event x_m in the sequence.

Inference Unit: Context-Specific Encoder. To encode input sequences and accommodate the effect of C on prediction, we consider an encoder $\Phi(\cdot)$ conditioned on a specific context. It takes a sequence of event embeddings and estimated context representation $\mathbf{c}_{(t)}$ at time step t (given by the branching unit detailed next) as input, and outputs a sequence of hidden states:

$$[\mathbf{h}^2_1, \mathbf{h}^2_2, \cdots, \mathbf{h}^2_t] = \Phi([\mathbf{h}^1_1, \mathbf{h}^1_2, \cdots, \mathbf{h}^1_t], \mathbf{c}_{(t)}; \Theta), \tag{6}$$

where $\mathbf{h}^2_m$ is the m-th hidden state for the event sequence. Assume that there are K types of contexts, i.e., $\mathcal{C} = \{\mathbf{c}_k\}_{k=1}^K$ and $c_{(t)} \in \mathcal{C}$. Each type of context c_k is represented as a K-dimensional one-hot

column vector $\mathbf{c}_k \in \{0,1\}^K$ whose k-th dimension is 1. Based on the SCM in Fig. 2, the context would change the underlying mechanism of how the next event is generated in real-world. Therefore, we consider context-specific *inference units* $\{\Phi_k(\cdot\,;\theta_k)\}_{k=1}^K$ as sub-networks of the encoder $\Phi(\cdot)$ to learn context-aware sequence representations, where $\Theta = \{\theta_k\}_{k=1}^K$. Then, we have

$$\Phi([\mathbf{h}_1^1, \mathbf{h}_2^1, \cdots, \mathbf{h}_t^1], \mathbf{c}_{(t)}; \Theta) = \sum_{k=1}^{K} \mathbf{c}_{(t)}[k] \cdot \Phi_k([\mathbf{h}_1^1, \cdots, \mathbf{h}_t^1]; \theta_k), \tag{7}$$

where $\mathbf{c}_{(t)}[k]$ returns the k-th entry of the vector $\mathbf{c}_{(t)}$. The inference unit can be specified as arbitrary off-the-shelf sequence models such as recurrent neural network (RNN) or self-attention (SA) [48].

Branching Unit: Dynamic Model Selection. We next introduce a *branching unit* $\Psi(\cdot)$ to parameterize the variational posterior $Q(C|S)$ that aims to obtain $\mathbf{c}_{(t)}$ given the sequence. At each time step, it takes the sequence representation as input and outputs a probability vector $\mathbf{q}_t \in [0,1]^K$, whose k-th entry represents the probability of the corresponding context c_k, i.e.,

$$\mathbf{q}_t = \Psi(\mathbf{h}_t^1; \Omega), \text{ where } \sum_{k=1}^{K} \mathbf{q}_t[k] = 1. \tag{8}$$

We use a d'-dimensional *context embedding* $\mathbf{w}_k \in \mathbb{R}^{d'}$ to accommodate the information of each context $\mathbf{c}_k$ by an embedding matrix $\mathbf{H}_c \in \mathbb{R}^{K \times d'}$, where $\mathbf{w}_k = \mathbf{c_k}^\top \mathbf{H}_c$ and $d' = d(d+1)$. Then, we split each context embedding into several fix-sized parameters $\mathbf{W}_k \in \mathbb{R}^{d \times d}$, $\mathbf{a}_k \in \mathbb{R}^d$:

$$\mathbf{W}_k = \mathbf{w}_k[:d^2].reshape(d,d),\ \mathbf{a}_k = \mathbf{w}_k[d^2 : d(d+1)]. \tag{9}$$

The attribution score s_{tk} that measures how likely a sequence up to time step t belongs to context c_k can be calculated via

$$s_{tk} = \mathbf{a}_k \cdot Tanh(\mathbf{W}_k \mathbf{h}_t). \tag{10}$$

Namely, we first project the sequence representation into a new d-dimensional space and then use dot product to measure the similarity. The variational distribution $Q(C|S = \mathcal{S})$ can be specified as a probability vector $\mathbf{q}_t$ that is computed by using Softmax function over s_{tk}, i.e., $\mathbf{q}_t = Softmax([s_{tk}]_{k=1}^K; \tau)$, where τ controls the confidence level. To implement the random sampling procedure $c \sim Q(C|S)$ in Eq. (3), one can also apply the categorical reparameterization trick that uses differentiable samples from Gumbel-Softmax [23] distribution, i.e.,

$$\mathbf{q}_t[k] = \frac{\exp\left((s_{tk} + g)/\tau\right)}{\sum_{k=1}^{K} \exp\left((s_{tk} + g)/\tau\right)}, \quad g \sim Gumbel(0,1). \tag{11}$$

Then, we can revise the right term of Eq. (7) as $\sum_{k=1}^{K} \mathbf{q}_t[k] \Phi_k([\mathbf{h}_1^1, \mathbf{h}_2^1, \cdots, \mathbf{h}_t^1]; \theta_k)$.

CaseQ: a Hierarchical Branching Structure. The aforementioned single-layered architecture requires an independent inference unit for each type of context, i.e., $|\mathcal{C}| = K$, which has several drawbacks: 1) It is impractical to accommodate a large amount of context types due to limited computational resources; 2) Real-world contexts are not entirely isolated, and assuming independent parameterization may lead to over-fitting and undesired generalizability given limited training data. To address these limitations, we further devise a hierarchical branching structure for CaseQ framework. Instead of using a one-hot vector, we denote a context as a 0-1 matrix, i.e.,

$$\mathbf{c}_k = stack([\mathbf{c}_k^1, \mathbf{c}_k^2, \cdots, \mathbf{c}_k^D]) \in \{0,1\}^{D \times K}, \tag{12}$$

where D denotes the number of layers and K denotes the number of parallel inference units in each layer. Each row in $\mathbf{c}_k$ is a one-hot vector indicating a certain inference unit in a layer, and each type of context corresponds to a certain combination of inference units. The entire network could then be represented as the stack of multiple parallel inference units and branching units:

$$[\mathbf{h}_1^{l+1}, \mathbf{h}_2^{l+1}, \cdots, \mathbf{h}_t^{l+1}] = \sum_{k=1}^{K} \mathbf{q}_t^l[k]\ \Phi_k^l([\mathbf{h}_1^l, \mathbf{h}_2^l, \cdots, \mathbf{h}_t^l]; \theta_k), \tag{13}$$

where $\mathbf{q}_t^l = \Psi^l(\mathbf{h}_t^l)$, and $l \in \{1, 2, \cdots, D\}$. The final prediction result at each time step can be obtained by ranking the relevance scores of current hidden states and event embeddings:

$$\hat{y}_t = \text{argmax}_{m \in \{1, \cdots, M\}} (\mathbf{H}_x \cdot \mathbf{h}_t^{D+1})[m]. \tag{14}$$

The above computation acts as a realization of $P_\theta(Y|S=\mathcal{S}, C=c)$ and $c \sim Q(C|S=\mathcal{S})$, required by the first term in Eq. (3). Besides, the KL term in Eq. (3) also requires density estimation from $Q(C|S=\mathcal{S})$. To achieve this, we use the continued Kronecker product of the probability vector $\mathbf{q}_t^l$ in each layer to represent the final variational posterior:

$$\mathbf{q}_t = Flatten\Big(\bigotimes_{l=1}^{D} \mathbf{q}_t^l\Big) \in \mathbb{R}^{K^D}, \tag{15}$$

where $\mathbf{q}_t^l \in \mathbb{R}^K$ and $\otimes$ denotes Kronecker product. The produced variational posterior is a K^D-dimensional vector, where each entry is the probability of a certain path (i.e., a combination of inference units across layers) in the network. This design has some merits. First, we could accommodate $|\mathcal{C}| = K^D$ types of contexts that grow exponentially w.r.t. model depth D. This helps to endow the model with larger capacity for learning diversified contexts. Second, different contexts may associate with each other by sharing the same inference unit in some layers, which could help to bridge their connection and plays as an effective *inductive bias* that guides the model to generalize to new environments (see more details and justifications in Appendix C.

According to Eq. (3), the final loss function can be written as

$$\sum_{\mathcal{S}} \sum_{t=1}^{|\mathcal{S}|-1} \left[l(y_t, \hat{y}_t) + \alpha \mathcal{D}_{KL} \left(\mathbf{q}_t \| \frac{1}{R} \sum_{j=1}^{R} \mathbf{q}(\mathcal{S}'_j) \right) \right]. \tag{16}$$

where α is a weight parameter for balance, $\mathbf{q}(\mathcal{S}'_j)$ denotes the produced variational posterior distribution with a pseudo sequence $\mathcal{S}'_j$ as input, and $l(\cdot,\cdot)$ can be arbitrary loss functions (e.g., cross-entropy) depending on specific tasks.

4 Experiments

The goal of our experiments is to test whether CaseQ can effectively handle temporal event prediction under distribution shift. We thus consider several temporal prediction tasks with particular evaluation protocols for the purpose.

Datasets. We evaluate the effectiveness of CaseQ with different applications for event prediction, including sequential recommendation, ATM maintenance and user awarded badges prediction. We use four datasets with variable length and number of event types: *Movielens*, *Yelp*, *Stack Overflow* and *ATM*. The detailed description of datasets and their statistics are given in Appendix E.

Evaluation Protocol. The commonly adopted *leave-one-out* evaluation for sequential event prediction [24, 45, 46] uses the latest event for testing, the event right before the last for validation, and the rest for training. To test the effectiveness of CaseQ for handling temporal distribution shift, we generalize such evaluation protocol by enlarging the time gap between training and testing data. We use the last $G+1$ events for testing, the first $|\mathcal{S}|-G-2$ events for training, and the $(|\mathcal{S}|-G-1)$-th event for validation. We call G *maximal gap size* and $g \in [0, G]$ *gap size* that means the number of events between the validation event and a specific testing event. When $G=0$ and $g=0$, it degrades to leave-one-out evaluation. By increasing g in our case, we could enlarge the time gap between training and testing data, so as to simulate temporal shift in real-world scenarios where data collection and online serving are during different periods of time. More details are in Appendix G.

Evaluation Metrics. For recommendation datasets, i.e., Movielens and Yelp, we adopt two widely used evaluation metrics: *Hit Ratio at K* (HR@K), and *Normalized Discounted Cumulative Gain at K* (NDCG@K). For other two low-dimensional datasets with fewer number of event types, i.e., Stack Overflow and ATM, we use Accuracy (i.e., Precision) as the metric.

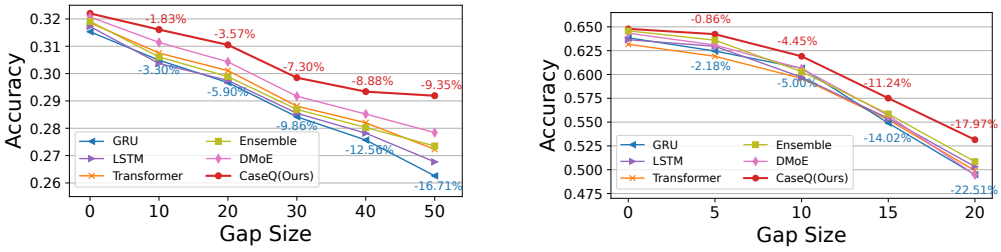
4.1 Sequential Recommendation

We adopt three sequence models as backbones for sequential recommendation: 1) GRU4Rec [20] that uses RNNs to model users' sequential behaviors and predict the next clicked item; 2) SASRec [24]

	Gap Size	GRU4Rec [20]		**CaseQ (GRU)**		SASRec [24]		**CaseQ (SA)**		SSE-PT [50]		**CaseQ (PT)**	
		NDCG	HR	NDCG	HR	NDCG	HR	NDCG	HR	NDCG	HR	NDCG	HR
Movielens	0	0.436	0.730	0.448 (+1.1%)[1]	0.736 (+0.8%)	0.453	0.742	0.462 (+2.0%)	0.752 (+1.3%)	0.465	0.753	0.476 (+2.3%)	0.763 (+1.3%)
	30	0.371	0.603	0.406 (+8.7%)	0.660 (+8.7%)	0.384	0.627	0.429 (+10.5%)	0.675 (+7.1%)	0.397	0.642	0.443 (+10.3%)	0.680 (+5.5%)
	Drop (%)	↓ 15.0	↓ 17.4	↓ 9.3 (−38.5%)[2]	↓ 10.3 (−40.7%)	↓ 15.2	↓ 15.6	↓ 7.1 (−53.1%)	↓ 9.3 (−40.6%)	↓ 14.6	↓ 14.7	↓ 7.0 (−51.8%)	↓ 10.9 (−25.9%)
Yelp	0	0.428	0.714	0.440 (+2.8%)	0.731 (+2.3%)	0.436	0.734	0.452 (+3.7%)	0.744 (+1.4%)	0.451	0.745	0.462 (+2.3%)	0.751 (+0.9%)
	20	0.395	0.663	0.426 (+7.2%)	0.702 (+5.5%)	0.402	0.682	0.433 (+7.0%)	0.719 (+5.2%)	0.405	0.685	0.440 (+8.0%)	0.717 (+4.5%)
	Drop (%)	↓ 7.6	↓ 7.2	↓ 3.2 (−57.9%)	↓ 4.1 (−43.1%)	↓ 7.6	↓ 7.1	↓ 4.3 (−43.3%)	↓ 3.4 (−52.5%)	↓ 10.3	↓ 8.0	↓ 4.7 (−54.5%)	↓ 4.5 (−43.7%)

[1] The percentage measures the improvement of CaseQ over the counterpart baseline with respect to one metric (e.g., NDCG).
[2] The percentage measures the degree of CaseQ helping to alleviate the performance drop compared to the counterpart.

Table 1: Experiment results of CaseQ with different base models. The *Drop* measures the percentage of decrease of a metric when time gap size increases.



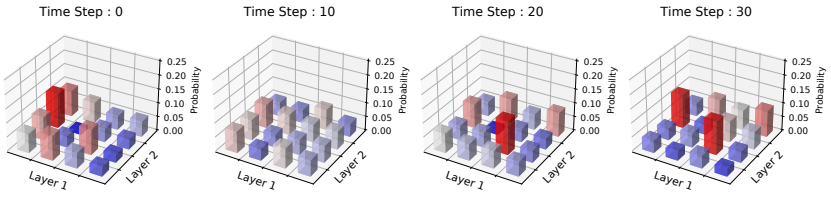
(a) Stack Overflow (b) ATM

Figure 3: Performance comparison of CaseQ and baselines w.r.t. different gap sizes.

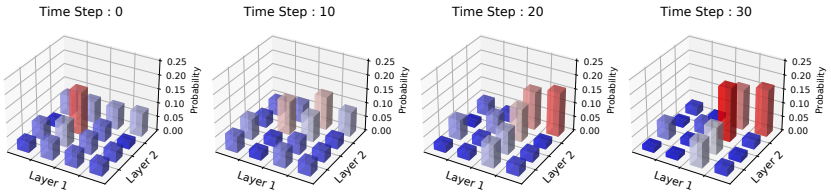
that adopts Self-Attention (SA) in Transformer's encoder for sequential recommendation; 3) SSE-PT [50] that further considers user's personalized embedding which incorporates user information into the input sequence. We use MLE for training these models. For fairness, our model uses the same number of layers as the baselines. We set gap size g as $0, 30$ for Movielens and $0, 20$ for Yelp (as the average sequence length of Movielens is greater than Yelp's). As shown in Table 1, our approach consistently outperforms the counterparts with different backbones throughout all the cases. While the performance degrades when we increase the gap size, CaseQ is significantly more robust to temporal distribution shift than the models trained with MLE. The percentage of the performance drop for CaseQ is nearly half of that for the counterpart models. Also, CaseQ is agnostic to the sequential prediction backbones and is promising for combining with other advanced architectures.

4.2 Event Prediction

We further study the performance of CaseQ in Stack Overflow and ATM datasets, which have distinct statistical features compared to sequential recommendation datasets. We consider three commonly adopted models for predicting discrete sequential events, i.e., GRU, LSTM and Transformer [48]. We also consider two ensemble frameworks that has the same model size with ours and are built on parallel structures: 1) *Ensemble* [7] adopts mean pooling over the hidden states of different inference units; 2) *DMoE* [10] combines the hidden states by learning a gating network in each layer. Also, we use multiple heads for Transformer to equip it with the same model size as CaseQ. For Stack Overflow, the gap size varies from 0 to 50. For ATM, it varies from 0 to 20. As shown in Fig. 3, CaseQ achieves the best results throughout all the cases in both datasets. As the gap size increases, the overall accuracy of baseline models exhibit a sharp downward trend in general, which shows the severe negative impact of temporal distribution shift. Despite with larger model size, both ensemble-based baselines show little improvement over the single models and also suffer a serious performance drop with larger gap sizes. In comparison, when the gap size goes large, the performance improvement of CaseQ over other competitors becomes more significant. This suggests that our approach is indeed effective for improving the robustness of sequence models w.r.t. temporal shift.



(a) LSTM as base model



(b) Transformer encoder as base model

Figure 4: Visualization of probabilities for different contexts recognized by well-trained CaseQ ($D = 2, K = 4, \tau = 1$). The results reveal the temporal distribution shift behind data.

4.3 Case Study

Visualization of Temporal Shift. To further investigate temporal distribution shift, in Fig. 4 we visualize the probabilities for each type of context at different time steps of a sequence. We set $K = 4, D = 2$. The two sub-figures show the results with different base models (LSTM and Transformer). The probability value reflects the intensity of a certain context at one time. As shown in the figure, while some contexts dominate at the beginning, they exhibit clear variation over time. The results imply that there indeed exists temporal distribution shift behind the data collected at different time intervals. Therefore, prior art use the past sequential data for training may not successfully transfer to new environments in the future. By contrast, our approach could first identifies latent contexts behind data and then alleviate its impact on learning a stable prediction model.

Learning of Context Embeddings. Context embeddings are latent representations for each type of context and dynamically affect the branching unit for selecting the proper inference unit. Conceptually, each type of context is expected to represent an environment in physical world, so each context embedding is supposed to be informative and distinctive. To verify whether our model could learn meaningful context representations without collapse or over-regularization, we visualize the context embeddings ($K = 3, D = 2$) through the training process in Fig. 5 by using dimensional reduction. Each color corresponds to the embedding of a certain context. While the context embeddings have the same initialized positions, they gradually separate and move towards different directions as the epochs increase. This indicates that the model manages to learn informative context representations.

4.4 Scalability Test

We further test the scalability of CaseQ w.r.t. the number of context types. Fig. 6 shows the training time and inference time per mini-batch of data. We use a single layer when the number of context types is within $[1, 6]$ and two layers when it further increases. As we can see in the range of $[1, 6]$, the training and inference time exhibit linear increase with a single-layer architecture. The increasing trend goes down when we increase the model depth. The results suggest CaseQ's decent scalability.

4.5 Hyper-Parameter Analysis

An appropriate number of context types are important for guaranteeing the performance and generalizability of our model. Recall that there are in total K^D types of contexts, given a model whose depth (i.e. number of layers) is D and width (i.e. number of parallel inference units in a layer) is K. Fig. 7 shows the performance of CaseQ w.r.t. different K and D in sequential recommendation task. The results show that increasing K and D in a certain range is effective for improving the performance.

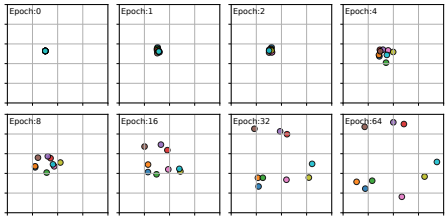


Figure 5: Visualization of context embeddings (reduced into a 2D space, $D = 3, K = 3$) at different training epochs.

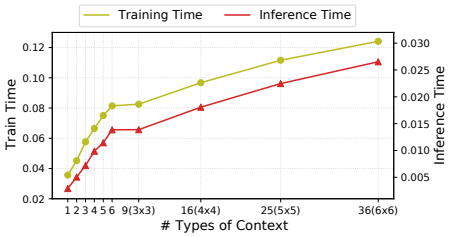
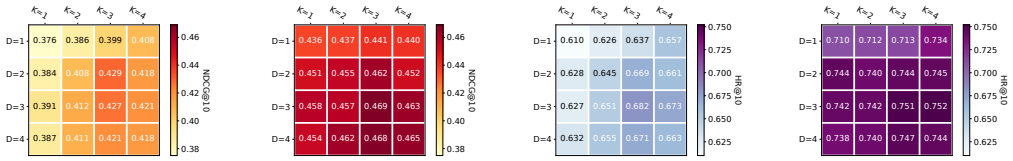


Figure 6: Train and inference time (s) of CaseQ w.r.t. number of context types.



(a) Gap Size = 30 (b) Gap Size = 0 (c) Gap Size = 30 (d) Gap Size = 0

Figure 7: Performance of CaseQ on recommendation task w.r.t. different K and D.

Notably, the improvements are more significant with larger gap size, which again verifies that CaseQ is helpful for addressing temporal shift. Still, out-of-range K and D would harm the performance, which is presumably due to over-fitting.

5 Limitations and Conclusion

Limitations and Potential Impacts. One of the main limitations of our variational context adjustment framework is that it stratifies discrete contexts (which may be infinitely many) into a fixed context set, in order to make the objective tractable. However, this remains a common issue for all works that are dealing with infinitely many discrete confounders, and our framework has already made a progress in this direction increasing the size of context set to K^D. We expect to see more works to be done in this direction. Also, due to the nature of the problem setting where the context is defined as a latent and abstract factor (which is a major challenge we aim to tackle and serves as our contribution), it would be hard or even impossible to identify its meaning in the real-world. Finally, we do not foresee any direct negative societal impacts.

Conclusion. In this paper, we endeavor to deal with temporal distribution shift in sequential event prediction. Namely, the training and testing data are generated from distinct distributions with different contexts. We first show that existing approaches with maximum likelihood estimation would leverage spurious correlation and produce undesirable prediction. Then we propose a new learning objective with backdoor adjustment and variation inference techniques. Based on this, we devise a novel model framework for learning context-specific representations. Extensive experiments demonstrate the practical efficacy of proposed method. We hope our work could enlighten new research focus for subsequent works and inspire more frontier research in sequential event prediction against distribution shift.

Acknowledgement

This work was partly supported by National Key Research and Development Program of China (2020AAA0107600), National Natural Science Foundation of China (61972250, 72061127003), and Shanghai Municipal Science and Technology (Major) Project (2021SHZDZX0102, 22511105100).

References

[1] Aman Agarwal, Kenta Takatsu, Ivan Zaitsev, and Thorsten Joachims. A general framework for counterfactual learning-to-rank. In *SIGIR*, pages 5–14, 2019.

[2] Aman Agarwal, Ivan Zaitsev, Xuanhui Wang, Cheng Li, Marc Najork, and Thorsten Joachims. Estimating position bias without intrusive interventions. In *WSDM*, pages 474–482, 2019.

[3] Juan-Jose Beunza, Enrique Puertas, Ester García-Ovejero, Gema Villalba, Emilia Condes, Gergana Koleva, Cristian Hurtado, and Manuel F Landecho. Comparison of machine learning algorithms for clinical event prediction (risk of coronary heart disease). *JBI*, 97:103257, 2019.

[4] Gilles Blanchard, Gyemin Lee, and Clayton Scott. Generalizing from several related classification tasks to a new unlabeled sample. In *Advances in Neural Information Processing Systems*, pages 2178–2186, 2011.

[5] Yuri Burda, Roger Grosse, and Ruslan Salakhutdinov. Importance weighted autoencoders. *arXiv:1509.00519*, 2015.

[6] Konstantina Christakopoulou, Madeleine Traverse, Trevor Potter, Emma Marriott, Daniel Li, Chris Haulk, Ed H Chi, and Minmin Chen. Deconfounding user satisfaction estimation from response rate bias. In *RecSys*, pages 450–455, 2020.

[7] Thomas G Dietterich. Ensemble methods in machine learning. In *International workshop on multiple classifier systems*, pages 1–15, 2000.

[8] Laurent Dinh, David Krueger, and Yoshua Bengio. Nice: Non-linear independent components estimation. *International Conference on Learning Representations*, 2015.

[9] 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 *SIGKDD*, pages 1555–1564, 2016.

[10] David Eigen, Marc'Aurelio Ranzato, and Ilya Sutskever. Learning factored representations in a deep mixture of experts. *arXiv:1312.4314*, 2013.

[11] Hui Fang, Danning Zhang, Yiheng Shu, and Guibing Guo. Deep learning for sequential recommendation: Algorithms, influential factors, and evaluations. *TOIS*, 39(1):1–42, 2020.

[12] Shanshan Feng, Xutao Li, Yifeng Zeng, Gao Cong, Yeow Meng Chee, and Quan Yuan. Personalized ranking metric embedding for next new poi recommendation. In *IJCAI*, 2015.

[13] Wei Gong and Laila Khalid. Aesthetics, personalization and recommendation: A survey on deep learning in fashion. *arXiv:2101.08301*, 2021.

[14] Mark Granroth-Wilding and Stephen Clark. What happens next? event prediction using a compositional neural network model. In *AAAI*, volume 30, 2016.

[15] F Maxwell Harper and Joseph A Konstan. The movielens datasets: History and context. pages 1–19, 2015.

[16] Chikara Hashimoto, Kentaro Torisawa, Julien Kloetzer, and Jong-Hoon Oh. Generating event causality hypotheses through semantic relations. In *AAAI*, 2015.

[17] Chikara Hashimoto, Kentaro Torisawa, Julien Kloetzer, Motoki Sano, István Varga, Jong-Hoon Oh, and Yutaka Kidawara. Toward future scenario generation: Extracting event causality exploiting semantic relation, context, and association features. In *ACL*, pages 987–997, 2014.

[18] Ruining He, Wang-Cheng Kang, and Julian McAuley. Translation-based recommendation. In *RecSys*, pages 161–169, 2017.

[19] Ruining He and Julian McAuley. Fusing similarity models with markov chains for sparse sequential recommendation. In *ICDM*, pages 191–200. IEEE, 2016.

[20] Balázs Hidasi and Alexandros Karatzoglou. Recurrent neural networks with top-k gains for session-based recommendations. In *CIKM*, pages 843–852, 2018.

[21] Matthew D Hoffman and Matthew J Johnson. Elbo surgery: yet another way to carve up the variational evidence lower bound. In *Workshop in Advances in Approximate Bayesian Inference, NeurIPS*, 2016.

[22] Guangda Huzhang, Zhenjia Pang, Yongqing Gao, Yawen Liu, Weijie Shen, Wen-Ji Zhou, Qing Da, Anxiang Zeng, Han Yu, Yang Yu, et al. Aliexpress learning-to-rank: maximizing online model performance without going online. *TKDE*, 2021.

[23] Eric Jang, Shixiang Gu, and Ben Poole. Categorical reparameterization with gumbel-softmax. *International Conference on Learning Representations*, 2017.

[24] Wang-Cheng Kang and Julian McAuley. Self-attentive sequential recommendation. In *ICDM*, pages 197–206, 2018.

[25] Benjamin Letham, Cynthia Rudin, and David Madigan. Sequential event prediction. *Machine learning*, 93(2):357–380, 2013.

[26] Da Li, Yongxin Yang, Yi-Zhe Song, and Timothy M Hospedales. Learning to generalize: Meta-learning for domain generalization. In *AAAI*, 2018.

[27] Jing Li, Pengjie Ren, Zhumin Chen, Zhaochun Ren, Tao Lian, and Jun Ma. Neural attentive session-based recommendation. In *CIKM*, pages 1419–1428, 2017.

[28] Sara Magliacane, Thijs van Ommen, Tom Claassen, Stephan Bongers, Philip Versteeg, and Joris M Mooij. Domain adaptation by using causal inference to predict invariant conditional distributions. *Advances in Neural Information Processing Systems*, 2018.

[29] Alireza Makhzani, Jonathon Shlens, Navdeep Jaitly, Ian Goodfellow, and Brendan Frey. Adversarial autoencoders. *arXiv:1511.05644*, 2015.

[30] Yishay Mansour, Mehryar Mohri, and Afshin Rostamizadeh. Domain adaptation: Learning bounds and algorithms. In *COLT*, 2009.

[31] Krikamol Muandet, David Balduzzi, and Bernhard Schölkopf. Domain generalization via invariant feature representation. In *ICML*, pages 10–18, 2013.

[32] Yulei Niu, Kaihua Tang, Hanwang Zhang, Zhiwu Lu, Xian-Sheng Hua, and Ji-Rong Wen. Counterfactual vqa: A cause-effect look at language bias. In *CVPR*, pages 12700–12710, 2021.

[33] Judea Pearl et al. Models, reasoning and inference. *Cambridge University Press*, 19, 2000.

[34] Judea Pearl, Madelyn Glymour, and Nicholas P Jewell. *Causal inference in statistics: A primer*. John Wiley & Sons, 2016.

[35] Judea Pearl and Dana Mackenzie. *The book of why: the new science of cause and effect*. Basic Books, 2018.

[36] Ladislav Peska and Peter Vojtas. Off-line vs. on-line evaluation of recommender systems in small e-commerce. In *HT*, pages 291–300, 2020.

[37] Zhi Qiao, Shiwan Zhao, Cao Xiao, Xiang Li, Yong Qin, and Fei Wang. Pairwise-ranking based collaborative recurrent neural networks for clinical event prediction. In *IJCAI*, 2018.

[38] Kira Radinsky, Sagie Davidovich, and Shaul Markovitch. Learning causality for news events prediction. In *WWW*, pages 909–918, 2012.

[39] Steffen Rendle, Christoph Freudenthaler, and Lars Schmidt-Thieme. Factorizing personalized markov chains for next-basket recommendation. In *WWW*, 2010.

[40] Margaret E Roberts, Brandon M Stewart, and Richard A Nielsen. Adjusting for confounding with text matching. *American Journal of Political Science*, 64(4):887–903, 2020.

[41] Marco Rossetti, Fabio Stella, and Markus Zanker. Contrasting offline and online results when evaluating recommendation algorithms. In *RecSys*, pages 31–34, 2016.

[42] Shiv Shankar, Vihari Piratla, Soumen Chakrabarti, Siddhartha Chaudhuri, Preethi Jyothi, and Sunita Sarawagi. Generalizing across domains via cross-gradient training. *arXiv:1804.10745*, 2018.

[43] Afshin Shoeibi, Marjane Khodatars, Roohallah Alizadehsani, Navid Ghassemi, Mahboobeh Jafari, Parisa Moridian, Ali Khadem, Delaram Sadeghi, Sadiq Hussain, Assef Zare, et al. Automated detection and forecasting of covid-19 using deep learning techniques: A review. *arXiv:2007.10785*, 2020.

[44] Henrik Stormer. Improving e-commerce recommender systems by the identification of seasonal products. In *AAAI*, pages 92–99, 2007.

[45] Fei Sun, Jun Liu, Jian Wu, Changhua Pei, Xiao Lin, Wenwu Ou, and Peng Jiang. Bert4rec: Sequential recommendation with bidirectional encoder representations from transformer. In *CIKM*, pages 1441–1450, 2019.

[46] Jiaxi Tang and Ke Wang. Personalized top-n sequential recommendation via convolutional sequence embedding. In *WSDM*, pages 565–573, 2018.

[47] Jakub Tomczak and Max Welling. Vae with a vampprior. In *AISTATS*, 2018.

[48] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. In *Advances in Neural Information Processing Systems*, pages 5998–6008, 2017.

[49] Wenjie Wang, Fuli Feng, Xiangnan He, Hanwang Zhang, and Tat-Seng Chua. " click" is not equal to" like": Counterfactual recommendation for mitigating clickbait issue. *SIGIR*, 2021.

[50] Liwei Wu, Shuqing Li, Cho-Jui Hsieh, and James Sharpnack. Sse-pt: Sequential recommendation via personalized transformer. In *RecSys*, pages 328–337, 2020.

[51] Qitian Wu, Yirui Gao, Xiaofeng Gao, Paul Weng, and Guihai Chen. Dual sequential prediction models linking sequential recommendation and information dissemination. In *Proceedings of the 25th ACM SIGKDD international conference on knowledge discovery & data mining*, pages 447–457, 2019.

[52] Qitian Wu, Hengrui Zhang, Junchi Yan, and David Wipf. Handling distribution shifts on graphs: An invariance perspective. In *International Conference on Learning Representations*, 2022.

[53] Shuai Xiao, Junchi Yan, Xiaokang Yang, Hongyuan Zha, and Stephen Chu. Modeling the intensity function of point process via recurrent neural networks. In *AAAI*, volume 31, 2017.

[54] Nianzu Yang, Kaipeng Zeng, Qitian Wu, Xiaosong Jia, and Junchi Yan. Learning substructure invariance for out-of-distribution molecular representations. In *Advances in Neural Information Processing Systems*, 2022.

[55] Jihang Ye, Zhe Zhu, and Hong Cheng. What's your next move: User activity prediction in location-based social networks. In *SDM*, pages 171–179, 2013.

[56] Dong Zhang, Hanwang Zhang, Jinhui Tang, Xiansheng Hua, and Qianru Sun. Causal intervention for weakly-supervised semantic segmentation. *Advances in Neural Information Processing Systems*, 2020.

[57] Shengyu Zhang, Dong Yao, Zhou Zhao, Tat-Seng Chua, and Fei Wu. Causerec: Counterfactual user sequence synthesis for sequential recommendation. In *SIGIR*, pages 367–377, 2021.

[58] Liang Zhao. Event prediction in the big data era: A systematic survey. *arXiv:2007.09815*, 2020.

[59] Sendong Zhao, Ting Liu, Sicheng Zhao, Yiheng Chen, and Jian-Yun Nie. Event causality extraction based on connectives analysis. *Neurocomputing*, 173:1943–1950, 2016.

[60] Sendong Zhao, Quan Wang, Sean Massung, Bing Qin, Ting Liu, Bin Wang, and ChengXiang Zhai. Constructing and embedding abstract event causality networks from text snippets. In *WSDM*, pages 335–344, 2017.

[61] Guorui Zhou, Na Mou, Ying Fan, Qi Pi, Weijie Bian, Chang Zhou, Xiaoqiang Zhu, and Kun Gai. Deep interest evolution network for click-through rate prediction. In *AAAI*, volume 33, pages 5941–5948, 2019.

[62] Kaiyang Zhou, Ziwei Liu, Yu Qiao, Tao Xiang, and Chen Change Loy. Domain generalization: A survey. *arXiv:2103.02503*, 2021.

[63] Kun Zhou, Hui Wang, Wayne Xin Zhao, Yutao Zhu, Sirui Wang, Fuzheng Zhang, Zhongyuan Wang, and Ji-Rong Wen. S3-rec: Self-supervised learning for sequential recommendation with mutual information maximization. In *CIKM*, 2020.

Checklist

1. For all authors...
   (a) Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope? [Yes]
   (b) Did you describe the limitations of your work? [Yes]
   (c) Did you discuss any potential negative societal impacts of your work? [Yes]
   (d) Have you read the ethics review guidelines and ensured that your paper conforms to them? [Yes]
2. If you are including theoretical results...
   (a) Did you state the full set of assumptions of all theoretical results? [Yes]
   (b) Did you include complete proofs of all theoretical results? [Yes]
3. If you ran experiments...
   (a) Did you include the code, data, and instructions needed to reproduce the main experimental results (either in the supplemental material or as a URL)? [Yes]
   (b) Did you specify all the training details (e.g., data splits, hyperparameters, how they were chosen)? [Yes]
   (c) Did you report error bars (e.g., with respect to the random seed after running experiments multiple times)? [No]
   (d) Did you include the total amount of compute and the type of resources used (e.g., type of GPUs, internal cluster, or cloud provider)? [Yes]
4. If you are using existing assets (e.g., code, data, models) or curating/releasing new assets...
   (a) If your work uses existing assets, did you cite the creators? [Yes]
   (b) Did you mention the license of the assets? [N/A]
   (c) Did you include any new assets either in the supplemental material or as a URL? [N/A]
   (d) Did you discuss whether and how consent was obtained from people whose data you're using/curating? [N/A]
   (e) Did you discuss whether the data you are using/curating contains personally identifiable information or offensive content? [N/A]
5. If you used crowdsourcing or conducted research with human subjects...
   (a) Did you include the full text of instructions given to participants and screenshots, if applicable? [N/A]
   (b) Did you describe any potential participant risks, with links to Institutional Review Board (IRB) approvals, if applicable? [N/A]
   (c) Did you include the estimated hourly wage paid to participants and the total amount spent on participant compensation? [N/A]

A Derivation of Backdoor Adjustment

We show the derivation of backdoor adjustment for the proposed causal graph in Fig. 2 based on the rules with *do*-calculus [33]. Consider a directed acyclic graph $\mathcal{G}$ with three nodes: X, Y and Z. We denote $\mathcal{G}_{\overline{X}}$ as the intervened causal graph by cutting off all arrows coming into X, and $\mathcal{G}_{\underline{X}}$ as the graph by cutting off all arrows going out from X. For any interventional distribution compatible with $\mathcal{G}$, we have the following two rules:

Rule 1. *Action/observation exchange:*

$$P(y|do(x), do(z)) = P(y|do(x), z), \ \textit{if}\, (Y \perp Z|X)_{\mathcal{G}_{\overline{X}\underline{Z}}} \tag{17}$$

Rule 2. *Insertion/deletion of actions:*

$$P(y|do(x), do(z)) = P(y|do(x)), \ \textit{if}\, (Y \perp Z|X)_{\mathcal{G}_{\overline{XZ}}} \tag{18}$$

We could derive the interventional distribution via:

$$\begin{aligned} P_\theta(Y|do(S=\mathcal{S})) &= \sum_{i=1}^{|\mathcal{C}|} P_\theta(Y|do(S=\mathcal{S}), C=c_i) P(C=c_i|do(S=\mathcal{S})) \\ &= \sum_{i=1}^{|\mathcal{C}|} P_\theta(Y|do(S=\mathcal{S}), C=c_i) P(C=c_i) \\ &= \sum_{i=1}^{|\mathcal{C}|} P_\theta(Y|S=\mathcal{S}, C=c_i) P(C=c_i), \end{aligned} \tag{19}$$

where the first step follows the law of total probability, the second step applies Rule 2, and the last step applies Rule 1.

B Derivation of Variational Context Adjustment

We derive the variational context adjustment in Eq. (3). We re-write the logarithm of interventional distribution $P_\theta(Y|do(S=\mathcal{S}))$ as

$$\log \mathbb{E}_{c\sim P(C)} \left[P_\theta(Y|S=\mathcal{S}, C=c) \right]. \tag{20}$$

We introduce a variational distribution $Q(C|S=\mathcal{S})$ as an approximation of the true posterior $P(C|S=\mathcal{S}, Y=y)$. By using variational inference, we derive a tractable lower bound as objective:

$$\begin{aligned} &\log P_\theta(Y|do(S=\mathcal{S})) \\ =&\log \mathbb{E}_{c\sim P(C)} \left[P_\theta(Y|S=\mathcal{S}, C=c) \frac{Q(C=c|S=\mathcal{S})}{Q(C=c|S=\mathcal{S})} \right] \\ =&\log \mathbb{E}_{c\sim Q(C|S=\mathcal{S})} \left[P_\theta(Y|S=\mathcal{S}, C=c) \frac{P(C=c)}{Q(C=c|S=\mathcal{S})} \right] \\ \geq&\mathbb{E}_{c\sim Q(C|S=\mathcal{S})} \left[\log P_\theta(Y|S=\mathcal{S}, C=c) \frac{P(C=c)}{Q(C=c|S=\mathcal{S})} \right] \\ =&\mathbb{E}_{c\sim Q(C|S=\mathcal{S})} \left[\log P_\theta(Y|S=\mathcal{S}, C=c) \right] - \mathcal{D}_{KL} \left(Q(C|S=\mathcal{S}) \| P(C) \right), \end{aligned} \tag{21}$$

where the third step is given by Jensen Inequality.

C More Discussions

We gather the continued Kronecker product of the probability vector $\mathbf{q}_t^l$ in each layer to compute the final variational posterior $\mathbf{q}_t$:

$$\mathbf{q}_t = Flatten \left(\bigotimes_{l=1}^{D} \mathbf{q}_t^l \right) \in \mathbb{R}^{K^D}. \tag{22}$$

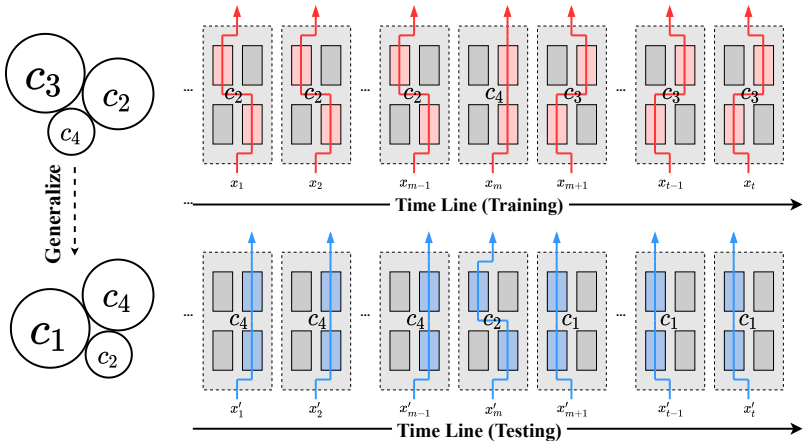


Figure 8: An intuitive explanation of how CaseQ generalize to novel context. Each small solid box denotes an inference unit. c_k denotes the context type at a certain time step. The size of the bubble represents its frequency. The colored boxes (in red for training and blue for testing) highlight the inference units selected by branching models.

Kronecker product is a generalization of the outer product from vectors to matrices. The entries of $\mathbf{q}_t$ denotes the probability for every possible combination of inference units (termed as *path*) at time step t. For example, consider $D = 3, K = 3$. We have $\mathbf{q}_t^l \in \mathbb{R}^3$, $\mathbf{q}_t^1 \otimes \mathbf{q}_t^2 \otimes \mathbf{q}_t^3 \in \mathbb{R}^3 \times \mathbb{R}^3 \times \mathbb{R}^3$, and $\mathbf{q}_t \in \mathbb{R}^{27}$. The i-th element in $\mathbf{q}_t$ denotes the probability of a certain path in the network, representing the probability of a type of context $Q(C = c_i|S = \mathcal{S})$. Suppose $i = (xyz)_3$ is a ternary number, the path corresponds to the combination of z/y/x-th inference unit at the first/second/third layer.

How CaseQ generalizes to novel context? As mentioned before, one advantage of such a design is that it could associate different types of contexts through shared inference units in some layers. By this way, we build links between majority (seen) contexts and minority (unseen) contexts. It can guide the model to generalize to future environments when the minority context may become the major one or unseen context occurs. We provide an example in Fig. 8 for further illustration. In training stage, the latent contexts gradually change over time (and so as the corresponding path in the network). During this time, c_2 and c_3 are majority contexts, c_4 is the minority context, and c_1 is a unseen one. Despite the imbalanced contexts, each inference unit as a module is relatively equally trained. Therefore, in testing stage, when c_1 and c_4 become majority contexts, our model can still generalize well by using well-trained inference units as modules.

D Comparison with Existing Causal Models

We compare CaseQ with existing causal models for sequence learning and user behavior modeling. Some existing methods [38, 16, 59, 60, 17] aim to extract causal relations based on observed or predefined patterns, yet they often require domain knowledge or side information for guidance and also do not consider temporal distribution shift. [1, 2, 6, 49] adopt counterfactual learning to overcome the effect of an ad-doc bias in recommendation task (e.g., exposure bias, popularity bias) or mitigate clickbait issue. Differently, they do not focus on modeling sequential events and are still based on maximum likelihood estimation as learning objectives. There are also few recent works that attempt to adopt causal inference to de-noise input sequences in sequential user behaviors[57]. Yet, they overlook the distribution shift and have distinct research focus compared with this paper.

E Dataset Descriptions

Movielens Movielens [15] is a widely used benchmark dataset for sequential recommendation evaluation. It contains one million ratings of 3900 movies made by 6040 users. We use each user's rating records as implicit feedbacks and construct a sequence of rated movies for each user in the order of timestamps, where each rating for a movie is treated as an event.

Datasets	#Sequences	#Event Types	#Events	Avg. length
Movielens	6039	3415	1.00M	165.50
Yelp	23056	15575	0.649M	28.14
Stack Overflow	4777	22	0.345M	72.25
ATM	1554	7	0.552M	355.35

Table 2: Statistics of four datasets.

Yelp Yelp is a dataset for business recommendation, provided by the Yelp Dataset Challenge [2]. After pre-processing and filtering, it contains $648,687$ reviews for businesses by $23,057$ users. We treat all user reviews as events and sort them chronically into sequences.

Stack Overflow This dataset is collected from a question answering website Stack Overflow [9]. It includes $480,000$ records of awarded badges of $6,000$ users during two years between 2012 and 2014. Each awarded badge is treated as an event. The task is to predict the next awarded badge of users.

ATM This dataset is collected from 1554 ATMs owned by an anonymous global bank headquartered in North America [53], which includes 7 types of events including 6 error types. It has a total number of $552,215$ events. The task is to predict future events for better maintenance support services.

The statistics of these datasets are summarized in Table 2.

F Implementation Details

We implement CaseQ with PyTorch. All parameters are initialized with Xavier initialization method. We train the model by Adam optimizer. All the models are trained from scratch without any pre-training on GTX 2080 GPU with 11G memory. For comparative methods, we refer to the hyper-parameter settings in their papers and also finetune them on different datasets. For sequences whose length is greater than 100, we only consider the most recent 100 events to speed up training. We run all experiments 5 times and take the average value.

G Evaluation Protocol and Problem Setting

Given a dataset consists of N event sequences, i.e., $\mathcal{D} = \{\mathcal{S}_i\}_{i=1}^N$, where $\mathcal{S}_i = \{x_{i,1}, x_{i,2}, \cdots, x_{i,|\mathcal{S}_i|}\}$. We use $\mathcal{S}_i[:t]$ to denote the first t events in the sequence and $\mathcal{S}_i[-t:]$ to denote the last t events. The training, validation and test sets are:

$$\begin{aligned}
\mathcal{D}_{train} &= \{(\mathcal{S}_i[:t-1], x_{i,t})\}_{i\in\{1,\cdots,N\}, t\in\{2,\cdots,|\mathcal{S}_i|-G-2\}} \\
\mathcal{D}_{valid} &= \{(\mathcal{S}_i[:t-1], x_{i,t})\}_{i\in\{1,\cdots,N\}, t=|\mathcal{S}_i|-G-1} \\
\mathcal{D}_{test} &= \{(\mathcal{S}_i[:t-1], x_{i,t})\}_{i\in\{1,\cdots,N\}, t\in\{|\mathcal{S}_i|-G,\cdots,|\mathcal{S}_i|\}},
\end{aligned} \tag{23}$$

where G is the maximal gap size. When the sequence length is less than or equal to $G+3$, we use the whole sequence for training. Given a specific gap size $g \in [0, G]$, we evaluate the model performance on a test subset:

$$\mathcal{D}_{test}(g) = \{(\mathcal{S}_i[:t-1], x_{i,t})\}_{i\in\{1,\cdots,N\}, t=|\mathcal{S}_i|-G+g} \quad . \tag{24}$$

By increasing g, we could enlarge the time gap between training and testing data, so as to simulate temporal distribution shift in real-world scenario where data collection and online serving are during different periods of time, and evaluate model's effectiveness against distribution shift. We are interested in the model performance drop as a metric to evaluate the robustness to distribution shift. Note that evaluations with different g are conducted simultaneously using the same model to control variates. When $G = 0$ and $g = 0$, this evaluation method exactly degrades to widely used leave-one-out evaluation protocol that is widely used in recommendation tasks [24, 45, 46].

[2] https://www.yelp.com/dataset/challenge