AI for Customer Journeys: A Transformer Approach

Abstract

When analyzing a sequence of customer interactions, it is important for firms to understand how these interactions align with key objectives, such as generating qualified customer leads, driving conversion events, or reducing churn. We introduce a transformer-based framework that models customer interactions in a sequence similar to how a sentence is modeled as a sequence of words by Large Language Models. We propose a heterogeneous mixture multi-head self-attention mechanism that captures individual heterogeneity in touchpoint effects. The model identifies self-attention patterns that reflect both population-level trends and the unique relationships between touch points within each customer journey. By assigning varying weights to each attention head, the model accounts for the distinctive aspects of the journey of each user. This results in more accurate predictions, enabling precise targeting and outperforming existing approaches such as hidden Markov models, point process models, and LSTMs. Our empirical application in a multichannel marketing context demonstrates how managers can leverage the model's features to identify high-potential customers for targeting. Extensive simulations further establish the model's superiority over competing approaches. Beyond multichannel marketing, our transformer-based model also has broad applicability in customer journeys across other domains.

Keywords:
Artificial Intelligence (AI), Transformers, Machine Learning, Customer Journey, Customer Heterogeneity

INTRODUCTION

With advances in Artificial Intelligence (AI), its integration into marketing has opened new possibilities for creativity, personalization, and efficiency. The recent focus has been mainly on the more flashy generative capabilities of AI for content creation, search engine optimization, idea generation, and improving customer support through chatbots (Carlson et al. 2023; Schipper et al. 2023; Huang and Rust 2024). However, AI's proficiency in handling large datasets and rapid computations – the qualities that emphasize its predictive capabilities – have remained largely untapped in marketing. Although some researchers have used deep learning models to analyze unstructured data such as text and images, the application of these techniques directly to customer panel data for actionable insights is still limited (Deveau, Griffin, and Reis 2023). To address this, we have developed a specialized AI-based transformer model dedicated to analyzing customer journeys, offering a novel contribution to the field. Our research demonstrates how AI methodologies can discern complex patterns within data, enabling firms to understand underlying trends in customer behavior and improve marketing decision making.

Understanding the customer experience and journey is central to a firm's growth and serves as an appropriate application to showcase AI's potential. With increasing channels and touchpoints such as social media interactions, email clicks, ad exposures, and search queries, customer journeys have become increasingly complex (Wedel and Kannan 2016; Lemon and Verhoef 2016). This complexity poses significant challenges to modeling, understanding, and proactively managing customer journeys. If a firm uses multiple channels to reach its customers, how does a specific channel and the timing of the interaction nudge a customer towards conversion? What role does each touchpoint play in the conversion of a customer?

Previous research has approached these issues from various perspectives, including understanding customer motivations from clickstream data (Moe 2003), exploring the search process (Dang, Ursu, and Chintagunta 2020), estimating channel attributions (Li and Kan-

nan 2014; Danaher and van Heerde 2018), and modeling the journey using Hidden Markov Models (Netzer, Lattin, and Srinivasan 2008; Abhishek, Fader, and Hosanagar 2012) and point process models (Goić, Jerath, and Kalyanam 2021). These models account for interdependent customer interactions, necessitating the consideration of prior and subsequent encounters to fully understand a single interaction. For example, a customer's usage of a search engine may signify either early-stage information gathering or later-stage purchase intent (Dang, Ursu, and Chintagunta 2020). Without information on prior and subsequent visits, it becomes challenging to determine the purpose of a single search engine visit. However, as the number of channels and touchpoints increases, understanding of the full journey becomes even more difficult as parameter space grows exponentially (Wedel and Kannan 2016).

To address these challenges, we leverage recent developments in AI and propose a transformer-based modeling framework to analyze large number of customer interactions across numerous channels. Our model understands and evaluates each interaction holistically by considering its context within the sequence of all prior and subsequent interactions in a customer journey. The transformer is a deep learning model designed to process sequential data in natural language processing (NLP) (Vaswani et al. 2017), underpinning architectures like GPT or Gemini. Although NLP and marketing problems may seem different, the transformer's ability to model each word within its context also applies to marketing settings. Specifically, it can handle a series of customer interactions as analogous to sequences of words in a sentence. The key innovation in transformers is the self-attention mechanism, which captures relationships between words and their context using attention weights. Attention weights in our model are trained to capture relationships between specific interactions and all prior ones, allowing for holistic evaluation. For example, Gmail's Smart Compose uses transformers to predict next words based on initial inputs (Chen et al. 2019). Applied to marketing, our model predicts visit and purchase probabilities in subsequent periods based on interaction history. It can also be applied to sequence-level classification tasks, such as predicting

customer churn or lifetime value.

Advances in sequence modeling from computer science offer substantial value to marketers. While traditional models often consider only short-term dependencies, studies have shown long-term dependencies significantly impact customer journeys (Mela, Gupta, and Lehmann 1997; Zantedeschi, Feit, and Bradlow 2017). Researchers have applied attention mechanisms to study the customer journey, with Zhou et al. (2019) using RNNs and a global attention mechanism to identify users' funnel stages, improving click-through and conversion rates by customizing messages. Recent marketing research recognizes the efficacy of deep learning methods in time series analysis. For instance, Valendin et al. (2022) employed LSTMs to predict customer transactions, outperforming traditional models like the Pareto/NBD model (Schmittlein, Morrison, and Colombo 1987) and the Gaussian Process model (Dew and Ansari 2018). Transformers can capture these long-term dependencies directly from data without relying on predefined functional forms.

The transformer model, with its self-attention mechanism, dynamically determines relationships between touchpoints, handling complex nonlinear interactions efficiently (Vaswani et al. 2017; Dai et al. 2019). In this context, transformers can be compared to VAR models (Dekimpe and Hanssens 1999, 2024), but while VAR models assume linear relationships and fixed lags, transformers handle complex nonlinear relationships and determines lags dynamically through self-attention mechanisms. Compared to models like HMMs, point-process models, and LSTMs, (see Table W1 in Web Appendix A), transformers offer greater flexibility and scalability, managing large numbers of channels and touchpoints effectively.

Our contributions highlight the power of transformers and its superiority over existing models in providing marketing insights not easily attainable with traditional methods. We extend the transformer model to handle customer-level heterogeneity, enhancing its applicability to provide descriptive insights into latent self-attention patterns characterizing an individual's customer journey. We demonstrate how the transformer model efficiently handles a large number of unique touchpoints and delivers results faster with superior prediction

performance when compared to that of existing methods. In our application to customer journeys in the context of multi-channel marketing, our model predicts the evolvement of purchase probability and touchpoint interaction over time for each customer based on their history. Using the model results, managers can understand and determine the impact of the timing and the channel used to target a customer. We examine the varying impact of each channel at different points in the customer journey on conversion, highlighting the impact of touchpoint timing and the importance of the sequential order of events. We conduct several analyses to illustrate how the transformer's superior predictive performance can lead to increased ROI by targeting the high potential customers. We also outline how, with the availability of appropriate marketing action data, the model's predictions can be translated into suggested actions and how these actions might meaningfully impact marketing performance.

We empirically compare our results with other models – Hidden Markov Models (HMM), point-process models, and Long Short-Term Memory (LSTM) – and demonstrate superior predictive performance and deeper managerial insights than those produced by these competitive models. One may question whether the proposed transformer's performance depends on the data. That is, compared to other benchmarks does the transformer only perform best on datasets whose data-generating processes (DGP) favors sparse and autocorrelated data? We conduct extensive simulation studies to evaluate the proposed transformer model against these competing models across datasets with different DGPs, a mixture of DGPs and sample sizes. These studies establish the boundary conditions under which the proposed transformer model performs comparably to the other approaches. However, under complex data patterns and large sample size – as is found in almost all commercial databases – the transformer significantly outperforms all competing models. Additionally, ablation experiments reveal how specific components of the transformer model provide a high degree of flexibility, effectively capturing underlying relationships and excelling in predictive tasks. Our model can be generalized to predict future events in other contexts. For instance, banks can forecast

customer churn by leveraging interaction history (Deveau, Griffin, and Reis 2023). Customer service departments can identify critical incidents shaping customer experience. Healthcare providers can anticipate patient outcomes by examining patient journeys.

The next section provides an overview of the model framework, focusing on adapting the transformer's multi-head self-attention mechanism to our marketing context.

MODEL

Figure 1 is an illustration of the model architecture. The input to the model is the customer journey data in a time-series format (see 1 at the left bottom of Figure 1). In the multi-channel marketing context, a customer interaction in the journey takes the form of a customer's visit through a channel or conversion at a visit. In each period t, the firm observes one of a customer's two states for each type of customer interaction: (1) if the customer interacts with the firm, or (2) no activity if the customer does not interact. Suppose there are S possible types of customer interactions that the firm can observe. We encode the customer journey sequence using an approach similar to the multi-hot encoding, capturing interactions per individual customer per time period. $\{X_{nst}\}$ $(s = 1, 2, ..., S; t = 1, 2, ..., T)$ is the matrix that contains the user n's interaction history with the firm from $t = 1$ to T. $X_{nst} = 1$ when interaction s occurs in t and $X_{nst} = 0$ when s is not observed. The number of types of interactions, S, should depend on the granularity of the data. For example, if the available data only indicates whether a display channel was accessed, it can be coded as a binary variable (0/1). On the other hand, if information about the specific ad copy shown on the display is available, the combination of display, with different ad copy can be represented by multiple binary variables, each indicating which ad copy was displayed. Because our model takes each individual's history as input, the subscript n is omitted in the following description. Let $\mathbf{X}_t = [X_{1t}, X_{2t}, ..., X_{St}]$, $\mathbf{X}_t$ can be viewed as an element (token) in the input sequence, similar to a token in a vocabulary used in NLP. The model takes an individual customer's history $\mathbf{X}_0, \mathbf{X}_1, ..., \mathbf{X}_t$ as a unit of input for processing, and outputs

predictions for $\mathbf{X}_{t+1}$, and repeats this for every $t = 1, ..., T-1$. During model training, the prediction is compared with the actual target label at $t+1$, and the model works to minimize the loss between these predicted and actual values. Researchers can set the target to various outcomes of interest such as visit through a channel or purchase incidence. Note that although the model generates predictions for the next period, one can predict multiple periods forward by predicting recursively based on previous predictions, as is done by other generative AI models.

In the case when continuous variables need to be incorporated[1], they are directly concatenated with the $\mathbf{X}_t$ vector to form a token. For example, suppose there are L continuous variables $Y_{1t}, Y_{2t}, ..., Y_{Lt}$, such as revenue or demographic information, etc. In this case, a token representing a customer's journey would be constructed as $\mathbf{X}_t = [X_{1t}, ..., X_{St}, Y_{1t}, ..., Y_{Lt}]$.

Shared Embeddings and Separate Encoders

In this task, our prediction target $\mathbf{X}_{t+1}$ is an S-dimensional vector (or $(S+L)$-dimensional vector in the case of mixed variable types). This makes the prediction a multi-objective optimization problem, which is also called multi-task learning in machine learning literature (Caruana 1997). By exploiting the commonalities among different tasks, the model can learn patterns more efficiently. In the customer journey scenario, learning to predict channel visits should help the model better predict conversion, and vice versa. However, optimizing multiple objectives at the same time unavoidably forces the model to make trade-offs between the performances on different tasks, especially when there are many model components shared across tasks (Crawshaw 2020). To strike a balance, when predicting the outcome for each type s, we first use a shared embedding layer to convert the touchpoint sequence $\mathbf{X}_t$ to embeddings (2 in Figure 1), and then assign an independent set of encoders for each prediction target s (3, 4, and 5 in Figure 1). This means that inside the model, the same

[1]Although we do not incorporate continuous variables in this paper, we have conducted analysis on a patient journey dataset in the healthcare setting that contains continuous variables. The results can be provided upon request.

user activities will be coded with the same embedding vector in the first step to facilitate learning, then the model explores substantive differences in predicting outcomes for each type of user activity. The shared embedding can be viewed as analogous to a common latent state variable in statistical models such as Hidden Markov Models (HMM).[2]

The embedding vectors represent different states in a multidimensional vector space where the relative positions of two vectors represents the similarity of the corresponding states. For NLP transformers, word embeddings pretrained from other language models are often used for transfer learning. Because our customer journey contexts are unique in the empirical setting and there is no existing model to learn from, the parameters of the embedding transformation are optimized based on the data during the training process. We transform the $\mathbf{X}_t$ to an embedding vector $x_t (x_t \in \mathbb{R}^{d_{model}})$ using a linear transformation. x_t has the dimension of d_{model}, which is a hyperparameter specified by the modeler. This is equivalent to assigning a vector to represent each type of interaction s in a d_{model}-dimensional space, while using the sum of the vectors to represent the period when there are multiple interactions happening within the same period, i.e., $X_{st} = 1$ for multiple s. When there are continuous variables $Y_{lt} (l = 1, ..., L)$, this transformation essentially uses a vector in the same d_{model}-dimensional space to represents one unit of each continuous variable Y_l.

So far the embedding vector $\mathbf{X}_t$ has not confronted the order information of the tokens. It is only a "bag of words" from the view of the model encoder. Transformers use the positional encoder to add the order information. The positional encoder is essentially a set of vectors added to the embedding vector, with a different vector added to every different t. Thus, $\mathbf{X}_t$ with the positional encoder added can have a time-varying impact on predictions even when the type of user activities is the same. The positional encoding of t-th position in a sequence is denoted as PE_t. It has the same dimension d_{model} as the embedding vector x_t. In NLP tasks, the positional encoding is added to the word embedding to account for the case in which the same word may have varying representations when used in a different position

[2]We compare the performance of the proposed transformer with that of HMM in the Model Comparison section. We thank the Associate Editor for suggesting this analogy.

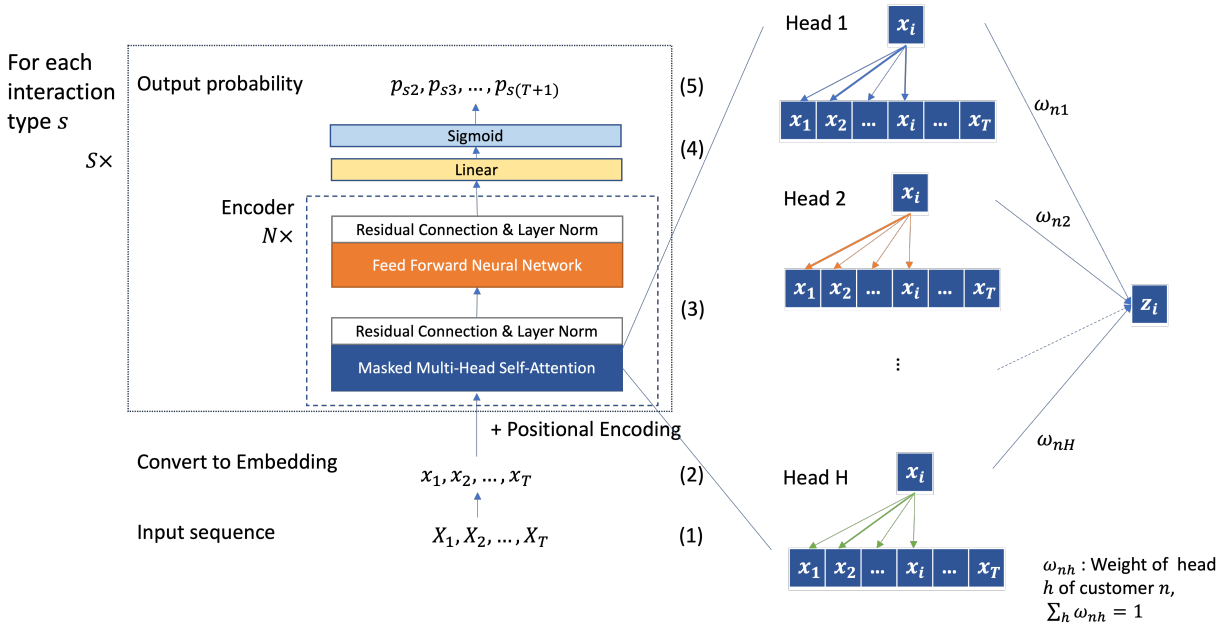


Figure 1: The architecture of the model

within the sentence, which yields the updated embedding $\tilde{x}_t = x_t + PE_t$. In our setting, we use positional encoding to account for the differential effects of the same type of interactions that happen at different times, aka "time effect." For example, a display click-through at the beginning of the customer journey should be different from a display ad click-through near the end of the customer journey.

Following Vaswani et al. (2017), we use sine and cosine functions for positional encoding, which takes the form $PE_{t,2\tau} = \sin\left(t/10000^{2\tau/d_{model}}\right)$ and $PE_{t,2\tau+1} = \cos\left(t/10000^{2\tau/d_{model}}\right)$ $(\tau = 0, 1, 2...; 2\tau \leq d_{model})$. 2τ, $2\tau + 1$ are the even and odd index of the elements in the positional encoding vector PE_t (assuming indexed from 0). The sine and cosine functions are one of the simplest ways to project the ordinal position numbers to a periodic sequence. Instead of one periodic function, the transformer uses a set of periodic functions with varying wavelengths, ranging from $2\pi(\tau = 0)$ to $10000 * 2\pi(2\tau = d_{model})$, to encode the positional effects. Each $2\tau^{th}$ and $(2\tau+1)^{th}$ element of the positional encoding vector PE_t captures the time effect at a different length scale. A smaller τ captures the time effect at a smaller scale and a larger τ captures time effect at a larger scale. This form of positional encoding assumes

periodic positional effects. Such configuration can capture the day-of-week or month-of-year effect that commonly affect customer's shopping behavior. From another perspective, the set of sine and cosine functions resemble the components in Fourier series, which can converge to arbitrary periodic functions with defined Fourier coefficients, thus capturing potentially non-linear time effects that impacts customer's shopping behavior.

With positional encoding added, embedding vectors enter a stack of encoder layers (see 3 in Figure 1), the heart of the transformer architecture. Starting from the encoder, the model diverges into separate paths for each prediction task s. We assign an independent set of encoder layers for each prediction target of interaction type s (3-5 in Figure 1). Each layer comprises two sub-layers: a self-attention layer and a feed-forward neural network (FFNN). In NLP tasks, the self-attention layer transforms input word embeddings, ensuring that the relative position of the output embedding vectors not only reflects its original semantic closeness in the input embedding, but also accounts for their dependence in the context. The "attention" refers to the distributed weight to words in a sentence (reflecting the relevance of the words) while focusing on one word at a time (e.g., the word "it" in Figure 2). In the context of customer journey data, "attention" refers to the weight or relevance of the past customer interactions on the specific interaction in focus. That is, we apply self-attention to uncover the dependence of the current customer interactions on the previous interactions in a journey, ignoring (masking) the subsequent interactions. Thus, we consider only past interactions during encoding a specific interaction (Figure 2b). In other words, only the past can predict the present; the future tells us nothing about now.

Self-Attention

The self-attention layers are the key innovation of the transformer. At the core of an attention-based approach is the ability to compare an item of interest to a collection of other items in a way that reveals their relevance in the current context. In the case of self-attention, the set of comparisons are to other elements within a given sequence. The

(a) Attention in sentence **(b)** Attention in customer journey

Figure 2: Illustrations of Multi-head Self-attention in Sentence and Customer Journey

simplest form of comparison between elements is a dot product. For two vectors x_i and x_j $(i, j \in \{1, ..., T\})$ in the input sequence, their dot product $x_i \cdot x_j$ reflects their relationship – a positive larger value indicates higher proximity in the embedding space. Because the value of the dot products can range from $-\infty$ to ∞, it is usually more desirable to normalize it over all items in the context, which yields a weight distribution of $\alpha_{ij} = \frac{\exp(x_i \cdot x_j)}{\sum_{k=1}^{i} \exp(x_i \cdot x_k)}, \forall j \leq i$. [3] Given the weight α_{ij}, a representation vector that incorporates the context information can be calculated by taking the weighted sum of all inputs seen so far $(j \leq i), z_i = \sum_{j \leq i} \alpha_{ij} x_j$. Now instead of a single x_i that only contains information about an individual item, the weighted sum z_i incorporates all information from the input, with different weights assigned to each item based on similarity.

In actual modeling, transformers go one step further by allowing a more flexible way of generating the weight α_{ij}. Specifically, each input embedding can play three different roles in the attention process described above. First, it can be the current focus of attention being compared to all of the other preceding inputs, i.e., the x_i. This role is referred to as a query.

[3] This equation for α_{ij} presented here is for demonstration purposes only. The actual form of α_{ij} used in the model is provided in Equation 2.

Second, it can serve as a preceding input being compared to the current focus of attention, which is referred to as a key. In the above example, x_j in α_{ij} is a key. Lastly, it can serve as a value being weighted and summed up, i.e., the x_j in the formula of z_i. The transformer captures these three different roles, and introduces three weight matrices W^Q, W^K and W^V for each role separately. These weights will be used to project each input vector x_i into a representation of its role as a query, key, or value.

$$q_i = W^Q x_i; k_i = W^K x_i; v_i = W^V x_i. \tag{1}$$

With the projection vectors q_i, k_i and v_i, the transformer uses the dot product between the query q_i and the key k_j, rather than the original x_i and x_j, to generate a weight distribution over other items in the context. The attention weight used in actual modeling is given by

$$\alpha_{ij} = softmax\left(\frac{q_i \cdot k_j}{\sqrt{d_k}}\right), \tag{2}$$

where d_k is a scalar, which is the dimensionality of the query and key vectors used to scale the dot products to a more suitable range for the subsequent processes. The transformer decoder predicts the next $i+1$ item based on information in $z_i = \sum_{j \leq i} \alpha_{ij} v_j$, which preserves information from all precedent inputs. From a reversed model training perspective, the α_{ij}, or the W^Q, W^K and W^V parameters, are trained in a way that reflects how much dependence to put on each precedent items when predicting the item at $i + 1$.

In the context of a multi-channel customer journey, consider the following five interactions in this specific sequence over time: e-mail, paid search, email, display and direct & booking as shown in Figure 2b. A query involving direct & booking is the relevance to itself when compared to all other interactions, while the key could examine how e-mail in the first position is relevant to direct & booking or e-mail in the third position is relevant to direct & booking or how paid search is relevant to direct & booking and so on. The query and the key are multiplied together to produce the attention scores. The value will be the representation

of each past interaction that is being weighted by its respective attention score to incorporate its relevance on direct & booking. See Figure 2b.

Multi-head Self-Attention with Customer Heterogeneity

To capture different patterns of word dependence, transformers use multi-head self-attention layers. Each head, denoted by h, trains an independent set of attention projection matrix W_h^Q, W_h^K and W_h^V, which can learn different aspects of the relationships that exist among inputs at the same level of abstraction. In Figure 2a, when processing the focal word "it", one head (blue) puts more weight on "cat", while another head (red) puts more weight on "hungry." To combine the information from multiple heads, Vaswani et al. (2017) concatenates z_{ih} from all heads $h = 1, ..., H$ and uses a matrix to project the concatenated vector $[z_{i1}, ..., z_{iH}]$ to form a new embedding.

In the context of the customer journey, the multiple heads capture different types of relevance relationships among the interactions using the self-attention patterns. For example, the first head's self-attention pattern could weigh the relevance of the e-mail interactions on direct & booking much more than other touchpoint interactions. The second head's self-attention pattern could weigh the relevance of firm-initiated touchpoints such as paid search and display on direct & booking more than other interactions. The other heads could capture other different self-attention patterns existing in the relevance relationships among the interactions. This feature of the transformer makes it much more flexible to model the relationships among interactions as compared to models such as HMM, point process or LSTM. In addition, we extend the transformer model to incorporate individual-level heterogeneity among consumers.

While modern marketers want to predict individual customer behaviors, most machine learning algorithms use the same set of parameters to model all inputs, ignoring individual differences. The heterogeneity across inputs is usually not the primary goal of ML models. Even though the way one person phrases a sentence will be very different from another per-

son in terms of word selection, tone, etc., this heterogeneity is generally ignored by most NLP models. For customer journey, we extend the transformer to incorporate individual heterogeneity, capturing the individual level variations in the relationships between events. For example, firm-initiated channels such as paid search and display ads may have a stronger effect on purchase for some customers than others. Uncovering and identifying such heterogeneity can help with user profiling and targeting. To incorporate individual heterogeneity, we propose and estimate a mixture-head attention mechanism, which is a variant of the transformer's multi-head self-attention. After getting the vector z_{ih} from head $h = 1, ..., H$, an individual n's vector embedding of period $t = i$ is a weighted sum of z_{ih}. And the weights of all heads sum up to unity (one). The new embedding takes the form

$$\bar{z}_{in} = \sum_{h=1}^{H} \omega_{nh} z_{inh}, \tag{3}$$

where ω_{nh} is the individual n's weight for head h and $\sum_{h=1}^{H} \omega_{nh} = 1$. The weights are estimated together with other parameters in the model training process. This renders it very similar to a finite mixture model for preference estimation.

The output of the attention layer is added to the original input embedding in a step called the residual connection. Afterwards, the summed-up vector is normalized, also known as layer norm process. These two steps are performed after each sub-layer. After the attention sub-layer and the layer norm operation, the output embedding goes through a feed-forward neural network (FFNN) sub-layer. Finally, the embedding is passed through a linear layer and was projected to proper size for output. For binary outcome variables, the model incorporates a sigmoid layer as the final activation layer (4 in Figure 1). In our application, for each position t in the sequence and each interaction type s, a linear layer projects the embedding to a single dimension, followed by a sigmoid layer that outputs the probability $p_{s,t+1}$ for the binary outcome at the next position $t + 1$. The target of prediction is decided by the modeler. It can be the customer's purchase decision in the following period or other

customer behavior of interest. More details of the FFNN and sigmoid layers can be found in Web Appendix A.

Depending on the type of tasks, different loss functions can be chosen to train the model. Because all variables to be predicted are binary in our application, we use the cross entropy loss as the loss function between the prediction and the target, and apply weights to balance the positive and negative class in the classification tasks (See Web Appendix A for a detailed discussion on loss function and class weighting). We minimize the mean loss across all dependent variables during the model training. In the case when continuous variables are present, loss functions such as mean squared error loss can be used.

Multi-step Prediction

By design, the transformer model predicts $\mathbf{X}_{t+1}$ based on the input sequence $\mathbf{X}_0, \mathbf{X}_1, ..., \mathbf{X}_t$. In practice, however, firms often plan marketing decisions over a longer horizon beyond a single time step. Therefore, a model's ability to generate accurate long-term forecasts across multiple future periods is critical. To evaluate long-term predictive performance, we hold out the final 20% of the time periods in our dataset. Multi-step predictions are generated recursively, following standard practice in time-series forecasting. Specifically, the model first outputs a probability estimate $p_{s,t+1}$, which is used to generate a prediction of user activity $\hat{X}_{s,t+1}$ via Bernoulli sampling. This predicted value is appended to the input sequence to form $\mathbf{X}_0, \mathbf{X}_1, ..., \mathbf{X}_t, \hat{\mathbf{X}}_{t+1}$, which is then used to predict $p_{s,t+2}$. The process continues iteratively until the end of the forecast horizon.

Predicting for multiple periods when the model is trained for one-step-ahead prediction has been shown susceptible to error accumulation problem, i.e., errors from the past are propagated into future predictions (Venkatraman, Hebert, and Bagnell 2015; Cheng et al. 2006). To mitigate this issue, we repeat the recursive prediction procedure multiple times and average the resulting probability estimates across runs. This ensemble-style approach helps stabilize the forecasts and improves reliability, especially over longer horizons.

APPLICATION

Data

The data for this study is from the hospitality sector. The focal firm uses multiple online marketing channels, such as paid search, display ads, and emails. Using first-party cookie data (collected using Adobe SiteCatalyst), the firm obtained a comprehensive history of customers' interactions with the firm's marketing channels. The data set includes 546,745 visits to the firm's websites clicking through several different channels, made by 92,575 users[4] belonging to the firm's loyalty program, spanning a time period from September 19, 2011, to December 14, 2011. On average, each user visited the firm's website about four times, made one booking, and generated a total revenue of $310 (Table 1). For each visit to the merchant's website, we observe the time and the source of the visit, whether a transaction was completed during the visit, and the revenue generated if the transaction was completed. After merging some minor sources, we examined visits from thirteen categories of campaign sources. Table 2 summarizes the sources of the visits and conversion rate for each source. About 45% of the visits represent direct traffic, the next largest being natural search (same as organic search). In terms of conversion rate, RESLINK[5] and B2B have the highest conversion rate because the visitors are business travelers with predetermined travel plans with conference hotels. Table 3 summarizes the statistics of the 102,375 transactions in the data set. On average, about one room is booked for two nights per transaction and the average revenue per booking is $282. About 38% of the bookings include a weekend stay, an indicator of leisure travel.

Table 1: Descriptive Statistics of Users

Per user	N	Mean	Min	Median	Max
No. of visits	92,575	4.35	1	2	100
No. of bookings	92,575	1.106	0	1	247
Total revenues	92,575	$310	0	$20	$55,674

[4]We remove the 58 users who have more than 100 visits during the observation period.

[5]RESLINK is short for reservation link, which are usually sent by event hosts to attendees.

Table 2: Visit Source Statistics

Campaign Source	N	%	# Bookings	Conversion Rate
DIRECT	246,106	45.0%	50,984	20.7%
NATURAL SEARCH	132,547	24.2%	25,390	19.2%
UNPAID REFERRER	66,474	12.2%	7,840	11.8%
PAID SEARCH	31,262	5.7%	5,559	17.8%
EMAIL	25,169	4.6%	3,721	14.8%
ECONFO AND PRE-ARRIVAL EMAIL	18,888	3.5%	3,138	16.6%
RESLINK	7,598	1.4%	2,441	32.1%
AFFILIATE	6,838	1.3%	1,734	25.4%
DISPLAY	6,557	1.2%	762	11.6%
REFERRAL ENGINE	3,350	0.6%	600	17.9%
SOCIAL MEDIA	924	0.2%	62	6.7%
EMERGING TECHNOLOGIES	658	0.1%	45	6.8%
B2B	374	0.1%	99	26.5%
Total	546,745	100%	102,375	18.7%

Note: a) RESLINK is short for reservation links, which are usually sent by event hosts to attendees. b) NATURAL SEARCH is often referred to as organic search. c) EMERGING TECHNOLOGIES mainly consists of the firm's App users.

Table 3: Descriptive Statistics of Transactions

	N	Mean	Min	Median	Max
Booked rooms	102,375	1.03	1	1	6
Booked nights	102,375	2.10	1	1	212
Revenue	102,375	$281	0	$172	$22,781
Include weekend stay	102,375	0.38	0	0	1

Model Training and Customer Journey Prediction

We divide the three-month window into 12-hour intervals, resulting in 173 time periods. We choose the 12-hour window to ensure that a time-window does not have too many touchpoints within it, which may hinder the modeling of sequence of touchpoints. The 12-hour windows allows us to have trade-off such issues against sparseness. For each period t, using the customer's visit and purchase history up to t, we predict their channel visit and purchase probabilities for $t+1$. The model input is a sparse time series that includes

both active periods with user activity and inactive periods without it. Since customers can visit multiple channels within a single period, channel visits are not mutually exclusive, making the common multinomial assumption inapplicable. Instead, our model predicts each customer's likelihood of visiting each channel during each time period[6].

We randomly sample 50% (46,288) of the customers as the holdout sample, with the remaining 50% used for training and validation via five-fold cross-validation. The model is trained using stochastic gradient descent, a widely used optimization method for deep neural networks (Farrell, Liang, and Misra 2021). Specifically, we adopt a hybrid approach, combining two variants of stochastic gradient descent: the Adam optimizer (Kingma and Ba 2014) for the heterogeneous head weights and mini-batch gradient descent for the remaining parameters. Detailed model training procedures are provided in Web Appendix A.

We split the training, validation, and holdout datasets at the time level: the first 140 time periods are designated as the calibration period for model training, while the final 33 periods are reserved to evaluate long-term prediction performance. This evaluation emphasizes the model's ability to forecast multiple future periods sequentially, rather than just the next immediate period. For out-of-sample customer predictions, where head weights are unknown, we use the average head weight from the training population $\bar{\omega}_h = \frac{1}{N}\sum_n \omega_{nh}$ as the weight for each head in the holdout sample. Table 4 reports the in-sample and out-of-sample AUC for the calibration and holdout periods of the proposed transformer model. For purchase prediction, Figure 3a shows the ROC curve for the first 140 calibration periods in both training and holdout samples. The in-sample AUC is 0.9435, and the out-of-sample AUC is 0.9205. Figure 3b illustrates the prediction performance over the 33 holdout periods following the calibration period, with an in-sample AUC of 0.8862 and an out-of-sample AUC of 0.8585. We also present the balanced accuracy and F1 results of model performance in Web Appendix B.

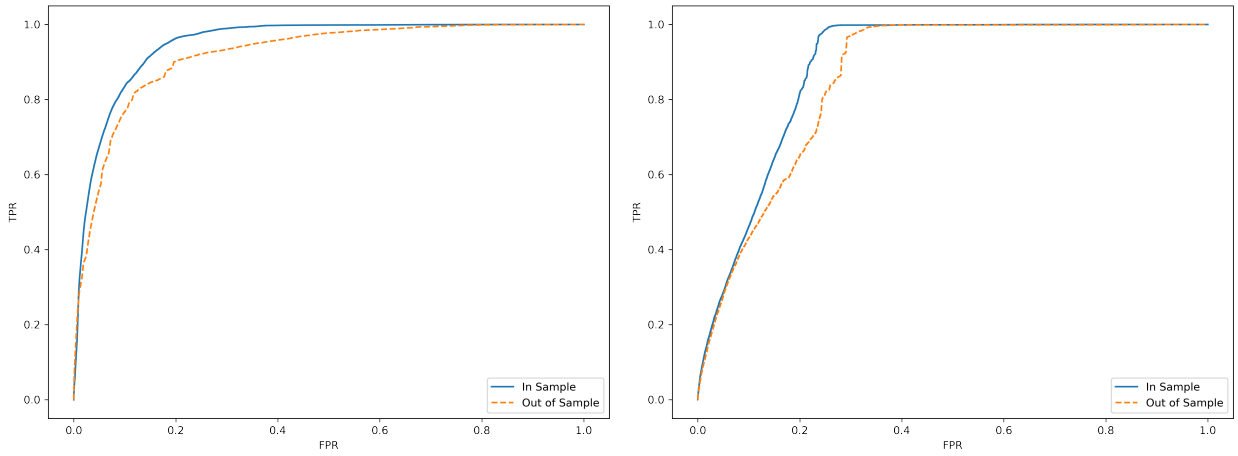
We randomly select two users from the data to demonstrate individual-level insights and

[6]Robustness checks with 6-hour and 24-hour intervals are presented in Web Appendix G.

Table 4: AUC of the Proposed Transformer Model

Dependent Variables	Calibration Period		Hold-out Period	
	In-Sample AUC	Out-of-sample AUC	In-Sample AUC	Out-of-sample AUC
Purchase				
Booking	0.9435	0.9205	0.8862	0.8585
Weekend Stay Booking	0.9395	0.9119	0.8685	0.8067
Channel Visit				
AFFILIATE	0.9937	0.9165	0.9172	0.8228
B2B	0.9994	0.9541	0.8386	0.7502
DIRECT	0.9225	0.8939	0.9018	0.8254
DISPLAY	0.9805	0.9042	0.8664	0.6354
ECONFO AND PRE-ARRIVAL EMAIL	0.9720	0.9176	0.8810	0.8555
EMAIL	0.9740	0.9197	0.8583	0.6834
EMERGING TECHNOLOGIES	0.9939	0.8879	0.3652	0.3579
NATURAL SEARCH	0.9402	0.8944	0.9010	0.7903
PAID SEARCH	0.9576	0.8972	0.8949	0.8332
REFERRAL ENGINE	0.9872	0.9198	0.8868	0.7261
RESLINK	0.9871	0.9197	0.7993	0.5426
SOCIAL MEDIA	0.9973	0.9180	0.8391	0.7978
UNPAID REFERRER	0.9692	0.9223	0.9072	0.8237

Note. a) The input variables include history of all dependent variables. b) The calibration period refers to the initial 140 periods used to train the model, while the hold-out period consists of the final 33 periods reserved for evaluation and have not been exposed to the model during training. c) In-sample AUC is the performance on the customers in the five-fold training samples, and out-of-of sample AUC is the performance on the 50% hold-out customers.

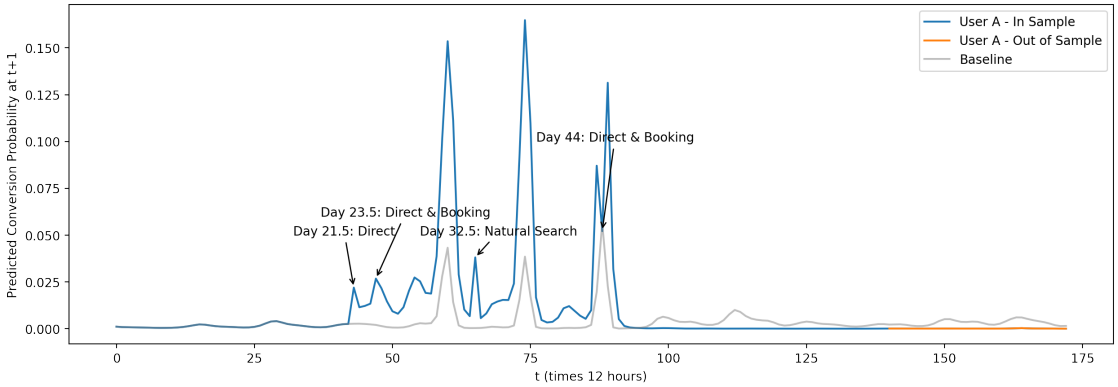


(a) Calibration Period ($0 \leq t < 140$) **(b)** Hold-out Period ($t \geq 140$)

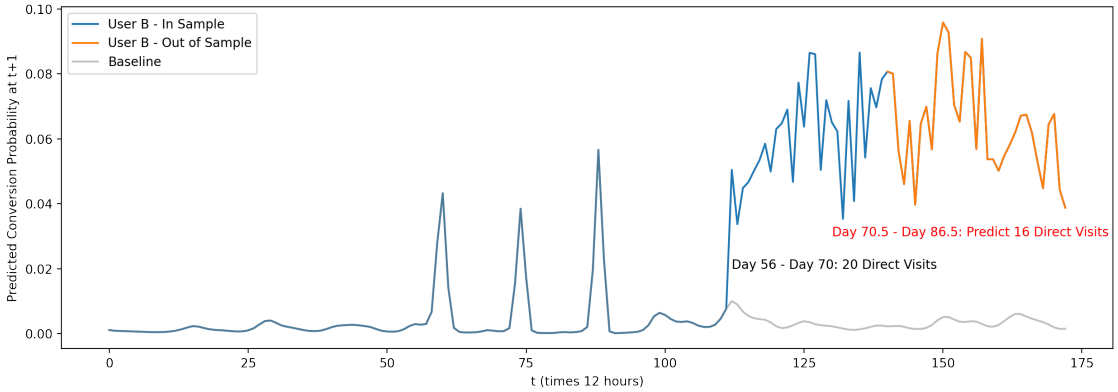
Figure 3: In-Sample and Out-of-Sample ROC Curve of Conversion Prediction

comparisons that the transformer model delivers. Figures 4a and 4b illustrate how their conversion probabilities (blue line) evolve over time compared to the baseline purchase probability (grey line), which assumes no observed visits ($\mathbf{X}_t = 0$ for all t) and reflects the overall

booking trend in the sample[7]. User A, who visits via direct channels and natural search mid-period, has a peak conversion probability of 0.15. In contrast, User B, with more visits but only via direct channels late in the period, peaks at 0.08. After completing transactions, User A's probability falls below the baseline, while User B's prediction aligns with the baseline until her first visit on day 56. Despite making 20 direct visits between days 56 and 70 without booking, her probability fluctuates around 0.06 before declining. These examples highlight how individual visit patterns and population trends jointly influence conversion predictions. Using the same two customers, we also demonstrate how their probability of visiting through different channels evolves over time in the Web Appendix B.



(a) User A - Booking Probability



(b) User B - Booking Probability

Figure 4: Predicted Booking Probability of the Subsequent Period

[7]The three peaks in all predictions are corresponding to the three booking peaks that occur from mid-October to early November observed in the data. We show the model-free evidence in the Web Appendix B

Predictive and Descriptive Marketing Insights

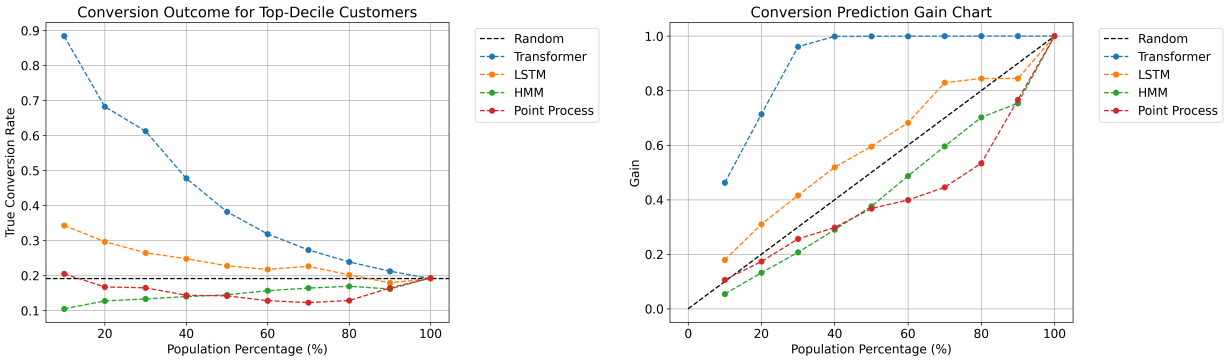
The possibility of predicting the evolvement of purchase probabilities at the individual customer level can help marketers in several ways to increase the return on investment (ROI) of marketing interventions. First, it can identify customers with high potential to convert to generate profiles for potential use in lookalike modeling. Second, we can estimate the time-varying impact weights of a specific intervention (say, email) on conversion for every individual and at the aggregate levels. We provide a comparison of such time-varying impacts across channels which reveal interesting insights into relative importance of the channels. With the availability of appropriate historical data on marketer actions, these analyses can help marketers choose among different targeting strategies. They can focus on specific intervention tools and appropriate timing of these interventions, thereby improving the effectiveness of targeting strategies as we discuss as extensions to the above analyses.

Identifying high-potential customers.

Given the high AUC values achieved by our model (see Table 4), it demonstrates strong potential for identifying customers with a high likelihood of conversion and generating behavioral profiles for lookalike modeling. To illustrate this, we create cumulative True Conversion Rate (TCR) and gain charts comparing our model's conversion predictions to those of competing models for users in the holdout sample.[8]

In Figure 5a – the true conversion rate (TCR) chart – we plot the true conversion rates (Y-axis) for customers in the top 10% of the holdout sample based on predicted probabilities, then the top 20%, and so on until the entire sample is included. A random selection of 10%, 20%, etc., from the sample would yield a constant true conversion rate of approximately 19%, representative of the entire sample (indicated by the black dashed line in the graph against which a lift is determined; but, in this graph, we just plot the true conversion rates in the Y-axis). In the top 10% of the sample identified by our transformer model (blue

[8]These are customers in the holdout sample during the holdout periods.



(a) True Conversion Rate Chart **(b)** Gain Chart

Figure 5: True Conversion Rate and Gain in Top Decile Customers

line), the true conversion rate is 88%. For comparison, we also evaluate the performance of competing models (HMM, Point Process, and LSTM).[9] The corresponding figures for competing models are considerably lower, with LSTM achieving the highest rate of about 34%. Figure 5a clearly demonstrates our model's superior performance at every targeted percentage level of the holdout sample, as shown by the blue line compared to the other lines.

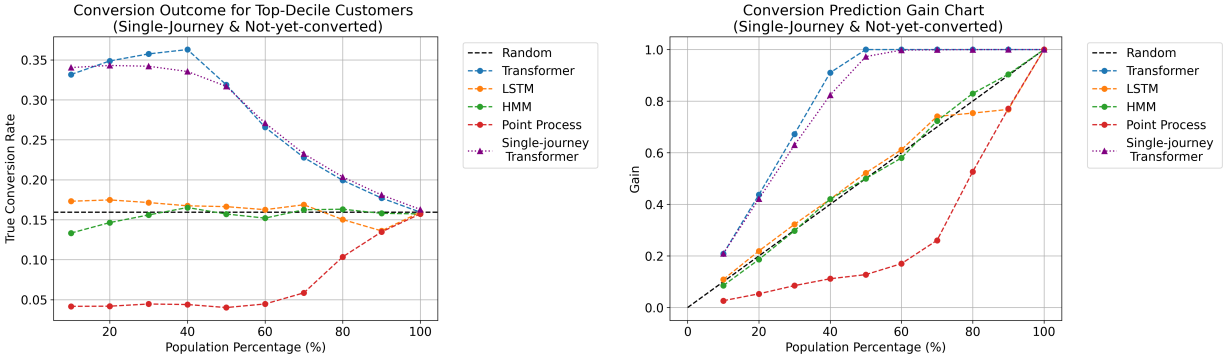
In Figure 5b, the gain chart shows the cumulative percentage of actual conversions (Y-axis) captured within the top 10%, 20%, etc., of the holdout sample ordered by predicted conversion probabilities from the different models. Our transformer model identifies 100% of actual conversions within the top 40% of the holdout sample. The LSTM, the next-best model, identifies only 83% of actual conversions even when extending to 70% of the holdout sample ordered by predicted probabilities. Both the TCR and gain charts clearly indicate that our model significantly outperforms competing approaches, offering superior accuracy and more precise profiling for targeted marketing, ultimately enabling higher ROI.

[9]Specifications and details of these models are discussed later in the paper under the Model Comparison section. These results are presented here for completeness and comparison. Note that the holdout sample size for our model and LSTM (n = 46,288) is the same, whereas the sample size for the HMM and Point Process models is smaller (n = 2,000) to expedite estimation, as these models are significantly slower due to their lack of parallel processing.

Single-Journey versus Multiple-Journey Customers.

An argument could be made that training models with data from users having multiple customer journeys, rather than from single journeys, might inflate model performance due to the additional information available about repeat behaviors (e.g., repeat visits or purchases). It could also be argued that predicting conversion rates for multiple-journey customers might be easier, whereas the critical utility of such models is predicting conversions in customer journeys without historical data beyond a single journey. Our results lend some credibility to this argument. Specifically, 84% of our data represents single-journey customers, with the remainder being multiple-journey customers. However, within the top decile of predicted conversions identified by our transformer model, 39% of customers have multiple journeys, leaving 61% as single-journey customers.

To test the capability of our transformer model in predicting conversions within a single customer journey context, we train the model exclusively on single-journey customers (n = 77,907) and used it to predict conversion rates for those single-journey customers who did not convert during the calibration period but may have converted during the holdout period. Approximately 16% of these customers converted in the holdout period.



(a) True Conversion Rate Chart **(b)** Gain Chart

Figure 6: True Conversion Rate and Gain in Top Decile Single Journey and Not-yet-Converted Customers

Figures 6a (true conversion rate chart) and 6b (the gain chart) compare the performances of our transformer model trained on single-journey data (the violet line) versus

multiple-journey data (blue line), alongside competing models trained on multiple-journey data (other lines). The cumulative lift chart illustrates that at the top 10% of the holdout sample ordered by predicted probabilities, the transformer model trained on single-journey data slightly outperforms the multiple-journey trained model (34% versus 33%, respectively). At the top 40%, however, the multiple-journey trained model performs slightly better. Overall, as indicated by the gain chart (Figure 6b), the transformer model shows comparable performance in identifying actual conversions from single-journey data, regardless of whether it was trained on single- or multiple-journey datasets. In contrast, competing models perform poorly in predicting conversions for single-journey customers who have not converted during the calibration period. The key takeaway from this analysis is that training the transformer model on multiple-journey data does not diminish its effectiveness in identifying conversions within single-journey datasets, even with significantly fewer touchpoints. This highlights the transformer's versatility and consistent predictive performance across different training data sets.

Prediction Performance Across Customer Types.

In the earlier subsection, we noted that within the top 10% of customers ordered by predicted probabilities derived from the entire dataset (including both multiple-journey and single-journey customers), 39% are multiple-journey customers, and 61% are single-journey customers—even though single-journey customers comprise 84% of the dataset. This clearly indicates that the model performs better at identifying multiple-journey customers, which aligns with expectations: more touchpoints and historical purchase information naturally lead to more accurate predictions.

We further analyze the top 10% of non-converted customers at the end of the calibration period, ranked by predicted conversion probabilities in the holdout period using the transformer model trained exclusively on single-journey data. Specifically, we compare the frequency distribution of touchpoints within this top decile to that of the entire holdout sam-

ple and calculated the corresponding lift values—defined as the true conversion rate within the top 10% divided by the average conversion rate among customers in the holdout sample with the same number of touchpoints. This metric indicates the model's effectiveness relative to a random selection baseline.

As summarized in the table below, we observe that conversion rate decrease slightly as we have more touchpoints in the customer journey. This could be because in our loyalty program data, increased engagement and interactions, more often than not, is indicative of no purchase. That is, those who purchase do it quickly with fewer touchpoints. The lift increases with more touchoints, which suggests more accurate prediction with more customer data.

Table 5: Touchpoint Frequency Table for Top 10% High Potential Customers

No. of	Top 10% Customers in Hold-out Sample			All Customers in Hold-out Sample			Lift
Touchpoints	Counts	Proportion	True Conversion Rate (TCR)	Counts	Proportion	True Conversion Rate (TCR)	
0	614	0.229	0.340	8802	0.328	0.163	2.090
1	780	0.291	0.331	9963	0.371	0.058	5.695
2	474	0.177	0.345	3258	0.121	0.073	4.754
3	238	0.089	0.316	1544	0.058	0.067	4.747
4	154	0.057	0.292	897	0.033	0.062	4.696
5	107	0.040	0.280	590	0.022	0.059	4.737
6	85	0.032	0.270	420	0.016	0.056	4.811
7	46	0.017	0.252	265	0.010	0.048	5.218
8	45	0.017	0.239	177	0.007	0.045	5.271
9	25	0.009	0.216	164	0.006	0.037	5.906
10	26	0.010	0.202	109	0.004	0.032	6.212
> 10	88	0.033	0.182	630	0.023	0.027	6.738

Note. Top 10% customers are identified by sorting single-journey, non-converted customers in the hold-out sample according to their purchase probability as predicted by the model. The True Conversion Rate (TCR) is given by the proportion of customers who convert within each segment. Lift is then calculated as the ratio of the TCR for the top 10% customers to the TCR for the full sample, reflecting the model's advantage over a random predicting model.

Time-varying impact of touchpoints.

With our model, we can estimate the time-varying impact of each touchpoint on conversions, both at the individual and aggregate level, assigning a time-varying importance score to each channel at each touchpoint. Choosing to estimate these scores with the widely used Shapley value can be computationally costly or virtually impossible to calculate the precise

values for each variable when their number is large (Castro, Gómez, and Tejada 2009). In this research, we use the Integrated Gradients Attribution method proposed by Sundararajan, Taly, and Yan (2017) to calculate the importance score for each customer interaction event in the journey (please see Web Appendix B for more details).

We select a hold-out sample consisting of 10% randomly sampled customers and calculate the importance score for the conversion probability prediction of each customer in each period. For each prediction of conversion probability, the importance score is calculated for each touchpoint interaction the customer has in their history of visits, indicating how the customer's probability of conversion will change with the touchpoint compared to without the touchpoint.

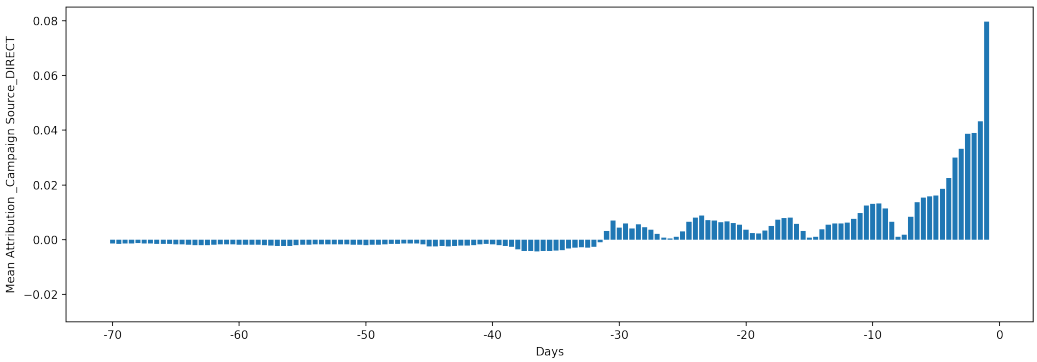


Figure 7: Time-Varying Impact of Direct Channel Visits

Do visits through the same touchpoint at different times impact conversion differently? To analyze this, we designate the purchase day as day 0 and compute the mean importance scores for touchpoint visits occurring at various time intervals prior to the purchase. Figures 7 and 8 show the mean attribution results for direct (customer-initiated) visits and email (firm-initiated) visits. The vertical axis represents the mean importance scores per visit, while the horizontal axis indicates the time difference between the visit and the purchase, with more recent touchpoints appearing on the right. As expected, the most recent visits (within a 12-hour window) have the highest impact on conversion probability, especially for direct visits (0.08). Interestingly, most touchpoints positively impact conversion predictions up to

a time threshold, typically around 30 days before purchase. Beyond this threshold, visits tend to exhibit slightly negative attribution scores, suggesting reduced purchase likelihood (as compared to the baseline probabilities) associated with earlier interactions.

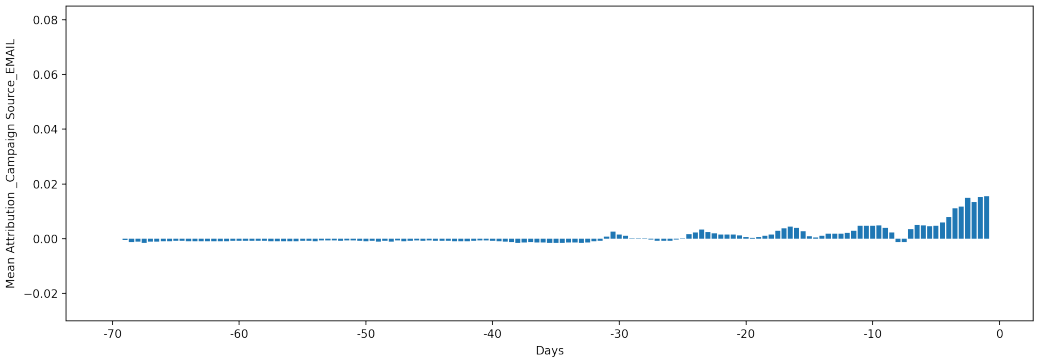


Figure 8: Time-Varying Impact of Email Channel Visits

Understanding the time-varying impact of channel touchpoints on purchases provides valuable insights into the effectiveness and limitations of firm-initiated interventions, such as paid search or email. These interventions generally exhibit a shorter span of impact compared to customer-initiated visits, like direct visits. Table 6 provides a comparison of the aggregate impacts of such touchpoints occurring in 7-day window prior to the conversion event and in 14-day window prior to the conversion event for all the channels as well as prior purchase. Direct visits have the most impact on a conversion event in the 7- and 14-day prior windows, followed by natural (organic) search, unpaid referrer, and affiliate. Paid search and e-mail have impacts that are lower than these other channels, highlighting their relative impacts under the marketing mix policy that generated this data. For natural search the 7-day prior window captures most of the impact and the 14-day window does not add anything significant in terms of impact. For B2B, the impact of prior visits is even shorter (0.0289 for 7-day window versus 0.0282 for the 14-day window). The impact of prior purchases is strong on conversion highlighting the impact of behavioral loyalty.

The estimates in Table 6 are important for assessing the differential effects of customer-initiated versus firm-initiated interactions within a given media mix allocation. Additionally,

Table 6: 7- and 14-Day Aggregate Impact after Touchpoint

	7-Day Aggregate Impact	14-Day Aggregate Impact
Booking	0.1186	0.2402
Channel Visit		
AFFILIATE	0.1328	0.1652
B2B	0.0289	0.0282
DIRECT	0.2865	0.3762
DISPLAY	-0.0112	-0.0132
ECONFO AND PRE-ARRIVAL EMAIL	0.0974	0.1293
EMAIL	0.0897	0.1171
EMERGING TECHNOLOGIES	-0.0278	-0.0319
NATURAL SEARCH	0.1609	0.1976
PAID SEARCH	0.0898	0.1135
REFERRAL ENGINE	0.0015	0.0026
RESLINK	0.0849	0.0988
SOCIAL MEDIA	-0.0100	-0.0094
UNPAID REFERRER	0.1398	0.1927

they can inform the optimal timing of interventions if data on historical marketing mix were available, as we discuss in the following subsection.

Extensions.

A user's customer journey consists of both customer-initiated and firm-initiated touchpoints. This raises a critical question: when is the optimal time for a firm to target a customer based on an observed customer-initiated touchpoint that might signal a potential sale? Targeting too early, before the customer is ready, or too late, after the purchase decision has already been made, can lead to ineffective ad targeting. The optimal timing of targeting has not been extensively explored in the existing literature, partly due to the sparsity of customer visit or transaction data over time. However, with the proposed transformer model, we can now dynamically tailor targeting strategies for each individual or cohort of individuals leveraging their observed history of visits and purchases to optimize touchpoint timing and maximize overall impact, with a significant caveat. Specifically, the pattern of ob-

served data is endogenous in that the marketing-mix variables are often chosen by managers with at least partial knowledge or expectation of the response parameters we estimate using our transformer model. If we have the history of marketing-mix decisions that lead to the observed data, we can estimate a supply-side model and relate it to our transformer model, similar to extant approaches in new empirical IO models (also see Manchanda, Rossi, and Chintagunta (2004)). To illustrate the power of our method in such targeting decisions, we provide an example of (a) e-mail targeting to explore different policies for e-mail targeting, (b) targeting timing with e-mail for individuals as well as (c) cohorts in Web Appendix C.

Marketing Implications of Multi-Head Self-Attention.

In an LLM implementation using transformers, each head captures a specific latent self-attention pattern or relationship between different words in a sequence, as illustrated in Figure 2a. In our application, the heads capture latent self-attention patterns that characterize relationships between touchpoints in a customer journey, similar to how latent classes in a finite mixture model capture heterogeneity in customer preferences. The number of heads, H, retained in the model is taken as a hyperparameter that is trained together with other parameters (see Web Appendix A for a discussion on hyperparameter tuning). In our application, we selected $H = 4$. We also validate our number of head selection by training a transformer model for $H = 1, 2, 3, 4$, respectively. The four models are then evaluated on the training and hold-out sample. Table 7 shows that the increase in AUC becomes marginal when moving from three to four heads.

Table 7: Transformer Performance under Different Number of Heads

Number of Heads	Mean AUC		Mean Balanced Accuracy	
	Training Sample	Hold-out Sample	Training Sample	Hold-out Sample
H=1	0.9019	0.8912	0.7976	0.7799
H=2	0.9665	0.9102	0.9006	0.7807
H=3	0.9717	0.9111	0.8886	0.7819
H=4	0.9714	0.9138	0.8817	0.7879

Figure 9 visualizes the attention weights generated by the four heads using their latent

self-attention patterns in the first layer in the modeling process of User A's journey [10] . The subsequent three layers exhibit similar patterns to the first layer. The sequence on the left represents the role of query, corresponding to the focal event being encoded. On the right, the sequence illustrates the role of key, representing the prior events in relation to the focal event under examination. The grey block on DIRECT indicates it as one key, while the grey block further down on the right indicates the non-event (no visit) as another key. The colored blocks in the visualization represent attention weights generated by the four attention heads, each with their own self-attention pattern. Lighter colors indicate lower attention weights with blue, orange, green and red distinguishing each head.

Starting with 'DIRECT' as the key, the blue and green heads indicate that the attention weight of the first direct visit is higher in the periods immediately following the visit (darker color). While the green head captures the immediate impact, the blue head reflects a slightly lagged effect, as it is lighter in the period immediately following the visit. Both heads show much lower weights after the second 'Direct & Booking' visit, illustrating that the impact of a prior interaction diminishes following the completion of a transaction, which often marks the beginning of a new customer journey. However, the impact persists slightly for several periods and even increases at certain points. These self-attention patterns may capture different purchase motivations or booking occasions, such as a customer quickly booking a hotel room for a business trip (immediate impact of a visit).

The orange head shows a very delayed impact of the direct visit, with darker colors appearing after a substantial lag, potentially capturing a customer booking a room for leisure travel after a longer decision-making process. When a non-event (the grey block representing 'no visit') is used as the key, none of the blue, orange, or green heads show engagement. However, the red head demonstrates a subtle initial impact that intensifies in later periods, indicating higher attention weights. This self-attention pattern might capture overall population trends in the customer journey data and associate them with the non-event

[10] For an interactive version of the figure, see https://zplu.github.io/blog/viewa_updated.html.

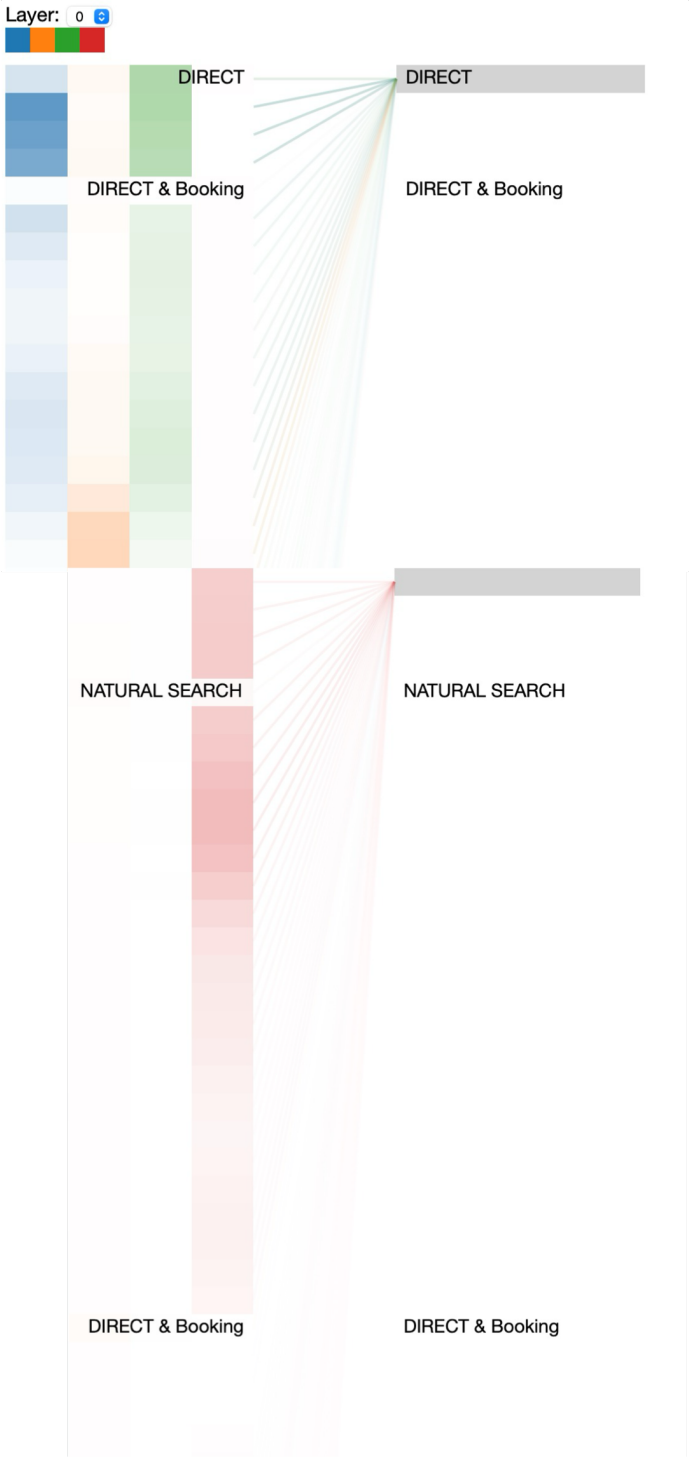


Figure 9: Attention Weights of the Four Heads in Modeling of the User A

In this context, the heads H with different self-attention patterns can be viewed as latent classes, with each head capturing a specific pattern (immediate effect, a lag effect, overall trend) characterizing motivations, booking occasions or population trends. The variation in head weights captures the customer-level heterogeneity in the mix of motivations/usage occasions that characterize their journeys, with individual head weights determining the mix appropriate for each user. The number of heads H and the corresponding user head weights can be viewed as latent classes and as the probabilities of belonging to each latent class. This can be useful for managers for descriptive purposes: (a) to segment the customer base, (b) generate profiles based on their relationship patterns between touchpoints, and (c) gain deeper understanding of their motivations in purchase journeys.

It is important to note that the self-attention weights we estimate in our transformer model are point estimates. Estimating the uncertainty in these weights could aid descriptive interpretation that we discuss above. While we do not perform such uncertainty estimation in the paper, we can use the Monte Carlo Dropout method suggested by Gal and Ghahramani (2016). This method involves setting (drop out) some neurons to zero in each layer during training. Specifically, we enable dropout at inference time and run the same input multiple times through the model. Since, each time, dropout causes slightly different outputs because of different neurons being active, we can collect these outputs and compute averages and standard deviations. This procedure approximates a Bayesian inference model (Gal and Ghahramani 2016). We could have the potential for redundant heads in estimating the different self-attention weights (Michel, Levy, and Neubig 2019; Gordon, Duh, and Andrews 2020). In our empirical application, this does not seem to be a problem as we have just four heads and the pattern of relationships in the individual heads are distinctly different as seen in Figure 9 visualizing the heads. Symmetry in hyperparameter tuning rarely leads to such problems in transformers as permutation symmetry breaks down due to optimization dynamics.

MODEL COMPARISONS

Among customer journey models, the Hidden Markov Model (HMM) (Netzer, Lattin, and Srinivasan 2008; Abhishek, Fader, and Hosanagar 2012; Li and Ma 2020) is a key benchmark, using hidden states to represent customers' underlying journey stages inferred from behaviors like visits or purchases. Recently, the Poisson point process model (Goić, Jerath, and Kalyanam 2021) was introduced to model the likelihood of events over time, a challenge for other models. LSTM, a recursive neural network model, has also been applied for sequence modeling to predict customer transactions (Valendin et al. 2022). For comparison, we use HMM, the Poisson point process, and LSTM as benchmarks to evaluate our model's performance.

Benchmark I: Hidden Markov Model

We build our HMM benchmark based on Li and Ma (2020). In addition to the 13 channels listed in Table 2, we introduce an outside option to represent periods when no visits to the firm's website are observed from a customer in a time period. This addition enables us to build the model on a panel-structured data instead of the touchpoint sequence data in Li and Ma (2020). Below, we briefly introduce the structure of the HMM used for comparison.

The model assumes S hidden states, representing different levels of a consumer's latent purchase intent. Let s_{it} denote the latent state of consumer i at time t, where $s_{it} \in \{1, \ldots, S\}$. The states are ordered by increasing intrinsic purchase propensity, which helps to deal with label switching. The initial probability that a consumer starts her journey from a state is given by $P^0 = (\rho_{01}, ..., \rho_{0S})$. State transitions depend on the channel and are governed by a channel-specific $S \times S$ transition matrix $P_c = \{\rho_{css'}\}_{s,s' \in \{1,...,S\}}$, where $c = 1, \ldots, C$ indexes the 13 channels and the outside option.

At each period, the consumer first decides whether to visit through a channel or not (i.e., choose the outside option). The probability of visiting through channel c is determined by

the consumer's current state s_{it} via an emission coefficient $\lambda_{cs_{it}}$, using a binary logit model:

$$p^v_{cs_{it}} = \frac{\exp(\lambda_{cs_{it}})}{1+\exp(\lambda_{cs_{it}})}, c = 1, ..., C-1. \tag{4}$$

The probability of choosing the outside option C is modeled as $p^v_{Cs_{it}} = \prod_{c=1}^{C-1}(1-p^v_{cs_{it}})$. The superscript v denotes "visit". The consumer's state evolves over time based on her choice of channel to visit (or not) and the corresponding transition matrix P_c.

Conditional on making a visit, the consumer then decides whether to make a purchase. The purchase probability is also state-dependent and follows a binary logit model based on a coefficient $\alpha_{s_{it}}$:

$$p^p_{s_{it}} = \frac{\exp(\alpha_{s_{it}})}{1+\exp(\alpha_{s_{it}})}, \tag{5}$$

where the superscript p denotes "purchase".

Benchmark II: Poisson Point Process Model

Our Poisson point process model is built based upon Goić, Jerath, and Kalyanam (2021). The Poisson point process models the arrival rates for each channel at each period. The number of visits in each time period is assumed to follow Poisson distribution. The propensity of consumer i to visit channel c at period t is given by

$$\mu_{ic'ct} = \mu_0 \exp(\alpha_{c'} + \beta_c + \theta\delta_{c'c} + \sum_c \rho_c \ln(1+N_{ict})). \tag{6}$$

The model is specified in a first-order Markov manner. c' is the previous channel visited by the consumer. N_{ict} is the number of times consumer i has visited the website through channel c up to time t. The $\alpha_{c'}$ captures the attractiveness of the last visited channel c' and the current channel β_c, respectively. $\delta_{c'c}$ is a dummy variable taking the value 1 if $c' = c$, so θ measures the inertia of visiting the same channel. The term $\ln(1+N_{ict})$ represents a cumulative inventory of visits for each channel, which is widely adopted by extant literature.

Conditional on having a visit to the website, the probability a consumer makes a purchase is described by a logit form

$$p_{it} = \frac{\exp(\phi_0 + \sum_c \phi_c \ln(1 + N_{ict}))}{1 + \exp(\phi_0 + \sum_c \phi_c \ln(1 + N_{ict}))}. \tag{7}$$

The probability of purchase depends on the weighted inventory of visits at each channel. The model assumes that consumers accumulate information at each channel as more visits are made through the channel, which impact their purchase probability. ϕ_0 determines the baseline purchase probability and ϕ_c determines the contribution of each channel.

Benchmark III: LSTM

We build the LSTM model similar to the transformer, with the same input and output variables. The model is built in a similar way that takes a customer's previous interactions as input and output the probability of visit or purchase in the next period. It also has a shared embedding layer and a separate separate LSTM layer for each prediction target variable of customer purchase or visit.

Performance Comparisons

We estimate HMM and Poisson point process using the Markov chain Monte Carlo (MCMC). We randomly sample 2,000 users from the same training data set we used for transformer training to train both models. For HMM, after comparing different numbers of states, we choose three states to estimate the model. We use the Adam optimizer to train the LSTMs. For all three benchmark models, the 173 time periods are split into 140 and 33 time periods, with the first 140 time periods calibration periods for training and the last 33 periods as hold-out periods.

Table 8 compare the model performances[11] of conversion prediction on the first 140

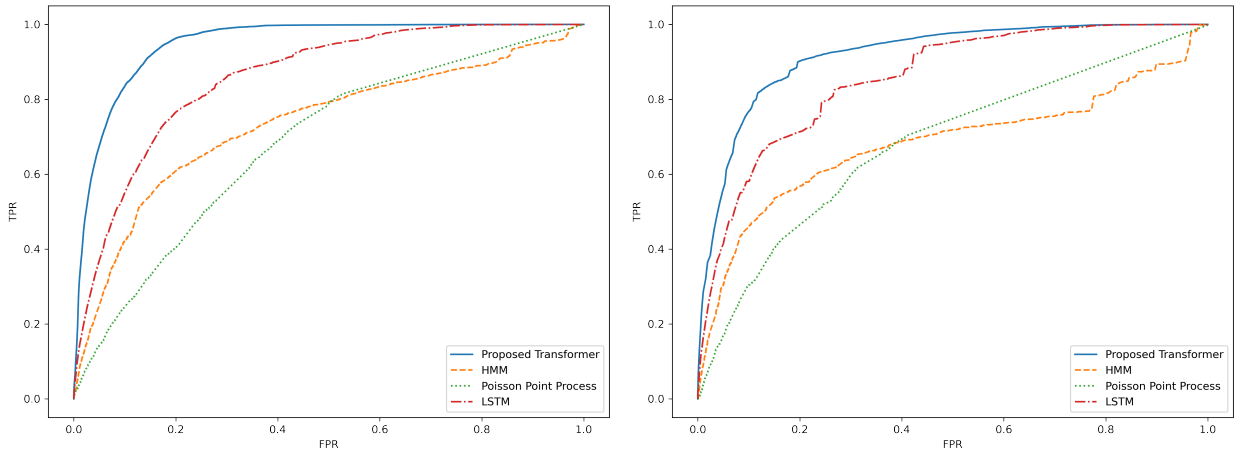
[11]We also present the balanced accuracy performance, as detailed in Web Appendix C, and observe results consistent with the AUC.

time periods on the training sample and hold-out sample respectively. Figure 10 shows the ROC curve of the proposed model versus three benchmark models. The training and hold-out samples are split by individual. The in-sample performance indicates the model goodness of fit in the model fitting process, while the out-of-sample performance indicates the model accuracy in the hold-out sample. When estimating the hold-out sample results, for each time period $t \leq 140$, all models predict a customer's visit channel and conversion in a subsequent period given the customer's history of all previous time periods. Compared to the benchmarks, the proposed transformer model has significant better out-of-sample performance, and it also has consistent in-sample and out-of-sample performances, indicating it is not overfitting the data.

Table 8: Model Comparison in the Calibration Period ($0 \leq t < 140$)

Dependent Variable	In-Sample AUC				Out-of-Sample AUC			
	Proposed Transformer	LSTM	HMM	Point Process	Proposed Transformer	LSTM	HMM	Point Process
Booking	0.9435	0.8466	0.7346	0.6826	0.9205	0.8456	0.6822	0.6817
Channel Visit								
AFFILIATE	0.9937	0.8544	0.8025	0.8394	0.9165	0.8289	0.7204	0.7761
B2B	0.9994	0.7342	0.8295	0.8524	0.9541	0.7498	0.8213	0.8833
DIRECT	0.9225	0.8043	0.7749	0.7590	0.8939	0.7938	0.7378	0.7634
DISPLAY	0.9805	0.8092	0.6846	0.7086	0.9042	0.7896	0.7074	0.6482
ECONFO AND PRE-ARRIVAL EMAIL	0.9720	0.8560	0.8170	0.7053	0.9176	0.8465	0.7874	0.6805
EMAIL	0.9740	0.8114	0.7598	0.7049	0.9197	0.8084	0.7124	0.7130
EMERGING TECHNOLOGIES	0.9939	0.7833	0.7947	0.5715	0.8879	0.7556	0.777	0.8140
NATURAL SEARCH	0.9402	0.8201	0.7439	0.7493	0.8944	0.8043	0.6814	0.7012
PAID SEARCH	0.9576	0.7894	0.6856	0.6723	0.8972	0.7747	0.6183	0.6652
REFERRAL ENGINE	0.9872	0.8040	0.7089	0.7212	0.9198	0.8012	0.7153	0.6890
RESLINK	0.9871	0.8232	0.5776	0.7907	0.9197	0.7938	0.5751	0.6848
SOCIAL MEDIA	0.9973	0.8879	0.7982	0.8240	0.9180	0.8326	0.7928	0.7264
UNPAID REFERRER	0.9692	0.8718	0.8556	0.8152	0.9223	0.8553	0.8053	0.7752

When making predictions of a customer journey, oftentimes the firm needs to predict more than one period ahead. Using the last 33 time periods as hold-out periods, the transformer model performances in the hold-out periods demonstrate its long-term predictive ability (Table 9). Figure 11 shows the ROC curve for this comparison. For the HMM and Poisson point process models, the long-term prediction performance declines greatly as the prediction



(a) In-Sample **(b)** Out-of-Sample

Figure 10: ROC curve of proposed model versus three benchmark models on the first 140 time periods.

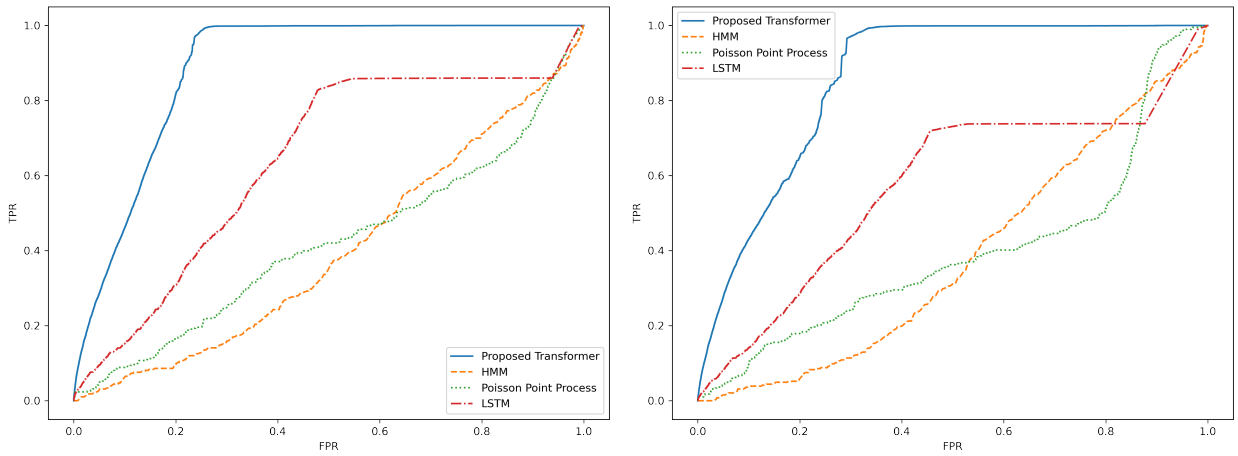
period becomes longer, which can be attributed to the error propagation for first-order Markov models. Because both the HMM and Poisson point process model rely solely on information from the previous period to predict the current period, when predicting more than one period ahead, the error from one period propagates and will carry over and affect the accuracy of the next period prediction. When the sequence to be predicted is long, the accumulated error can be very large. The transformer mitigates the issue of long, sparse data by spanning its self-attention across a long sequence. Thus the prediction performance of the transformer model is much better than the three benchmark models in long-term prediction.

SIMULATION

A question naturally arises: Why does the transformer outperform alternative models, even when compared with the LSTM, which shares similar neural network structures. This section aims to address this through simulation. Our simulation studies are divided into two parts. First, we conduct extensive simulations across various data-generating processes (DGPs) to evaluate the performance of transformer models compared to the same benchmark models. The results demonstrate that transformers consistently perform well, even when a

Table 9: Model Comparison in the Hold-out Period ($t \geq 140$)

Dependent Variable	In-Sample AUC				Out-of-Sample AUC			
	Proposed Transformer	LSTM	HMM	Point Process	Proposed Transformer	LSTM	HMM	Point Process
Booking	0.8862	0.6380	0.398	0.4142	0.8585	0.5737	0.3839	0.3947
Channel Visit								
AFFILIATE	0.9172	0.6487	0.9096	0.9573	0.8228	0.6359	0.7099	0.7503
B2B	0.8386	0.3592	0.3962	0.8673	0.7502	0.3995	-	-
DIRECT	0.9018	0.6008	0.5889	0.5667	0.8254	0.5857	0.6169	0.5730
DISPLAY	0.8664	0.6171	0.6055	0.5265	0.6354	0.5661	0.6364	0.5679
ECONFO AND PRE-ARRIVAL EMAIL	0.8810	0.7114	0.4891	0.6646	0.8555	0.6740	0.6189	0.5522
EMAIL	0.8583	0.6719	0.6325	0.5653	0.6834	0.6098	0.5957	0.5448
EMERGING TECHNOLOGIES	0.3652	0.6411	0.7284	0.4689	0.3579	0.5990	0.9324	0.4663
NATURAL SEARCH	0.9010	0.6598	0.5990	0.5668	0.7903	0.5737	0.6016	0.5636
PAID SEARCH	0.8949	0.6309	0.5584	0.6094	0.8332	0.6018	0.658	0.7258
REFERRAL ENGINE	0.8868	0.6447	0.6679	0.5704	0.7261	0.5434	0.6557	0.5944
RESLINK	0.7993	0.6827	0.5771	0.5933	0.5426	0.5811	0.7217	0.7351
SOCIAL MEDIA	0.8391	0.6516	0.2457	0.4568	0.7978	0.7498	0.9714	0.4576
UNPAID REFERRER	0.9072	0.7417	0.6311	0.6781	0.8237	0.6858	0.6427	0.6667



(a) In-Sample **(b)** Out-of-Sample

Figure 11: ROC curve of proposed model versus three benchmark models on the last 33 hold-out time periods.

competing model aligns with the underlying DGP. Moreover, transformers excel in handling datasets with mixed DGPs or complex non-linear relationships, emphasizing their versatility. These simulations highlight the superiority of transformer-based models across diverse scenarios while identifying the boundary conditions of their performance relative to competing models. Second, we perform ablation experiments to analyze how different transformer

components contribute to prediction accuracy[12]. The results reveal that disabling positional encoding significantly degrades performance across all DGPs, while reducing self-attention (via masking touchpoints) similarly impacts performance in high-order DGPs, emphasizing the critical role of self-attention in capturing higher-order processes. Additionally, reducing the number of heads significantly affects performance in mixed DGPs, highlighting the importance of multi-head attention for handling complex relationships. We elaborate on these findings in the sections below.

Model Comparison under Different DGPs

In the Model Comparison section, we evaluated the proposed transformer against HMM, Point Process, and LSTM models using our application data. To assess performance under diverse conditions, we conduct systematic simulations across various data-generating processes (DGPs). These simulations are designed to approximate the broad range of DGPs encountered in real-world scenarios, offering a comprehensive comparison of the transformer's capabilities relative to the benchmark models.

HMM and Point Process DGPs.

We conduct 50 simulation scenarios with varying parameters under the HMM and Point Process DGPs. These DGPs are specified as in the benchmark models discussed in the Model Comparison section, with the number of channels reduced to three. For each model, we draw 50 sets of parameters from their prior distributions and generate 50 datasets, simulating visit and purchase behavior for 1,000 customers over 100 time periods. Table 10 summarizes the simulated datasets. For each dataset, we estimate the Transformer, LSTM, HMM, and Poisson Point Process models. Since the true data-generating probabilities are known, we evaluate predictive performance by comparing each model's probability outputs to the true probabilities using average cross-entropy. Lower cross-entropy scores indicate

[12]Please see Web Appendix E for results of ablation experiments

better alignment with the true probabilities. We also assess classification performance using AUC, balanced accuracy, and F1 scores, where higher values indicate better performance. To simplify comparisons, we calculate the absolute deviation of each model's in-sample predictions from the best-performing model (which has zero deviation) for each metric. Figure 12 reports the mean absolute deviation across the 50 datasets, where smaller deviations indicate better performance. The Transformer demonstrates superior performance in cross-entropy, closely aligning with the true DGP probabilities. For classification metrics, the model aligned with the DGP performs best as expected. Yet notably, the Transformer consistently achieves strong classification performance across all scenarios, comparable to that of the DGP model.

Table 10: Summary Statistics of the Simulated Datasets

	Number of Datasets	Mean	SD	Min	Median	Max
DGP - HMM						
Frequency of Channel 1	50	0.489	0.180	0.171	0.442	0.846
Frequency of Channel 2	50	0.503	0.161	0.179	0.529	0.852
Frequency of Channel 3	50	0.523	0.158	0.193	0.535	0.755
Frequency of Purchase	50	0.429	0.112	0.207	0.430	0.644
DGP - Point Process						
Frequency of Channel 1	50	0.200	0.135	0.0004	0.207	0.368
Frequency of Channel 2	50	0.198	0.136	0.0001	0.210	0.373
Frequency of Channel 3	50	0.195	0.134	0.0004	0.198	0.366
Frequency of Purchase	50	0.230	0.249	0.0004	0.114	0.722

Note. Frequency of a channel is calculated by the number of periods with a visit through the channel divided by the total number of periods. Frequency of purchase is calculated by the number of periods with a purchase divided by the total number of periods.

Autoregressive DGPs with calendar effects.

Next, we conduct simulations comparing performances under DGPs of different autoregressive process. The AR DGPs are designed to create dependencies spanning various number of time steps (AR1, AR3, AR5) in the touchpoint sequence. Each variable is modeled as a function of the lags of all other variables, analogous to the structure of a Vector Autoregression (VAR) model. We further simulate varying degrees of calendar effects (weak/strong),

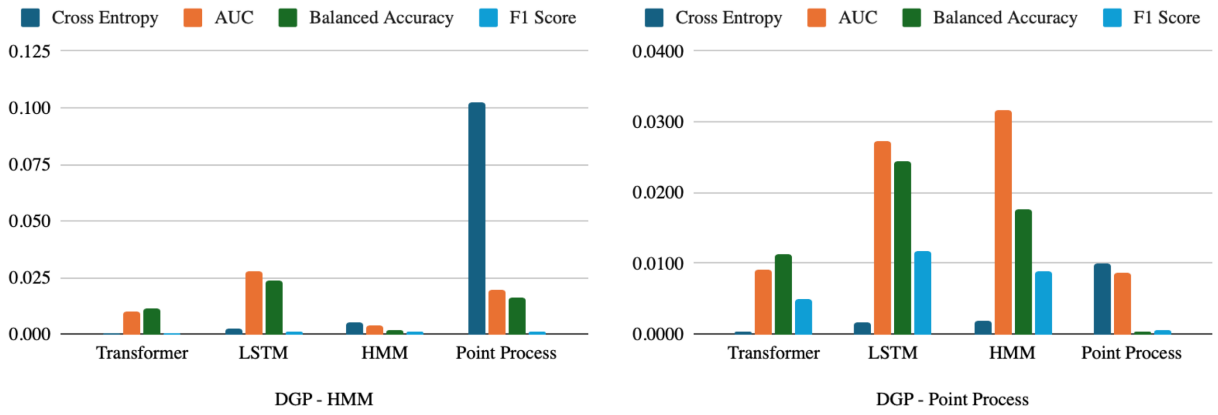


Figure 12: Mean Absolute Deviation from the Best Performing Model across the 50 Simulated Datasets

including day-of-week and month-of-year effects, on top of the AR process. The coefficients for the strong and weak calendar effects are drawn from different uniform distributions. The details of model specifications can be found in Web Appendix F. We simulate panel data of 10,000 customers across 100 time periods for each AR model, with each dataset having three channel visit variables and one purchase indicator. The AR datasets have a larger customer base because we find that sample size plays an important role in identifying the inter-temporal dependency, on which we have run a separate experiment specified below.

The performance comparison is shown in Figure 14. Under AR DGPs without calendar effects, transformers perform better than HMM or Point Process models, while performing as well as LSTMs under AR1 and second to LSTMs under other AR conditions. This result highlights LSTMs' excellent performance in handling high-order linear dependencies in the sequence data (Siami-Namini, Tavakoli, and Siami Namin 2018). In the presence of calendar effects, transformers outperform LSTMs under AR1 and narrow the gap with LSTMs under other AR conditions, indicating transformers are better at identifying time effects. This could result from the different mechanisms two models use to identify time effects: LSTMs process the data step-by-step, each step processing one time step of the input and passing its hidden state to the next step. The sequence order is captured implicitly in this structure. Transformers, on the other hand, process the entire sequence simultaneously. They use

explicit positional encoding to inject information about the order of the sequence.

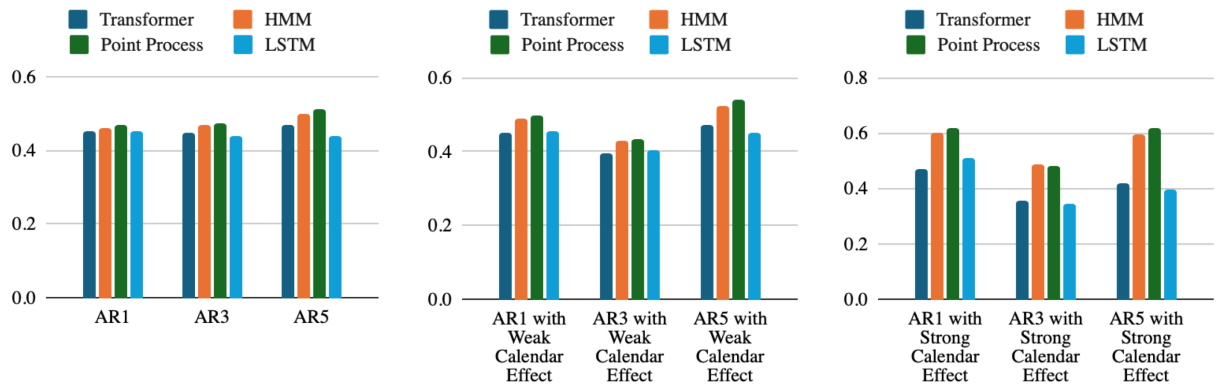


Figure 13: Mean Cross Entropy with True DGP Probability for AR Datasets

Varying sample size under AR DGPs.

We find that the sample size plays a critical role for transformers to identify the dependencies in the AR simulations. To validate this, we simulate four datasets, each containing 10,000, 20,000, 50,000, and 100,000 customers, respectively, all simulated under the same AR5 DGP. All datasets have the same time window of 100 periods. We compare the performance of the proposed transformer and LSTM for each sample size. The results are shown in Figure 14. As the sample size increases, the performance gap between the transformer and the LSTM decreases. The difference in AUC between the two models is less than 1% for sample size of 100,000. This result highlights that, compared with LSTMs, transformers need a larger sample size to effectively capture the linear dependencies in the sequence.

Mixture DGP.

To approximate the complexity of real-world data, where the true generating process is often unknown and multiple mechanisms may influence customer journeys, we construct a mixture DGP combining multiple types of generating processes: (a) an HMM DGP, (b) a Point Process DGP, and (c) an AR5 process with weak calendar effect. All three DGPs are from the DGPs described above, with three channel visit variables and a purchase indicator.

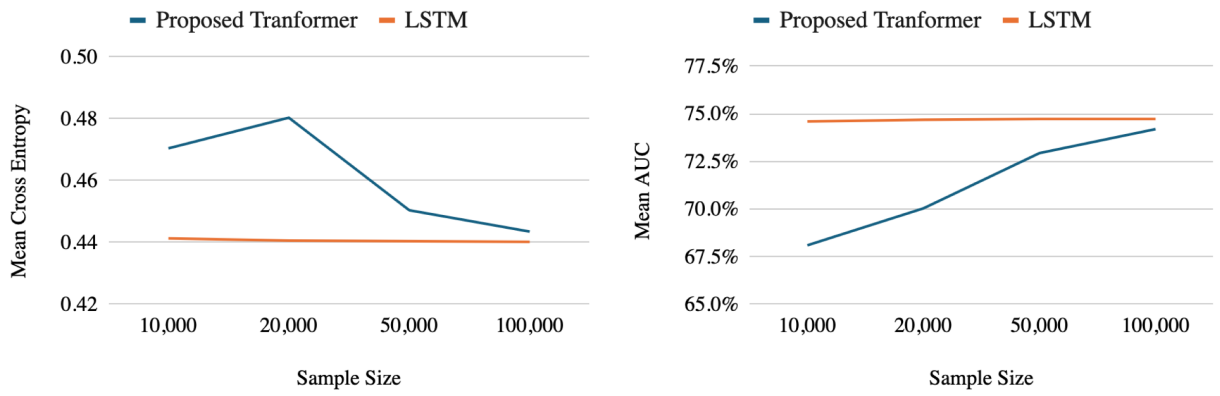


Figure 14: Transformer and LSTM Performance under Different Sample Size under AR5 DGP

The probability distribution over the three DGPs is drawn from a flat Dirichlet distribution and remains constant for all customer in all periods. Figure 15 shows the model comparison under the mixture DGP. The transformer outperforms all other models in predicting all four variables in both cross entropy and AUC, highlighting that the transformer handles complex data patterns much better than competing models.

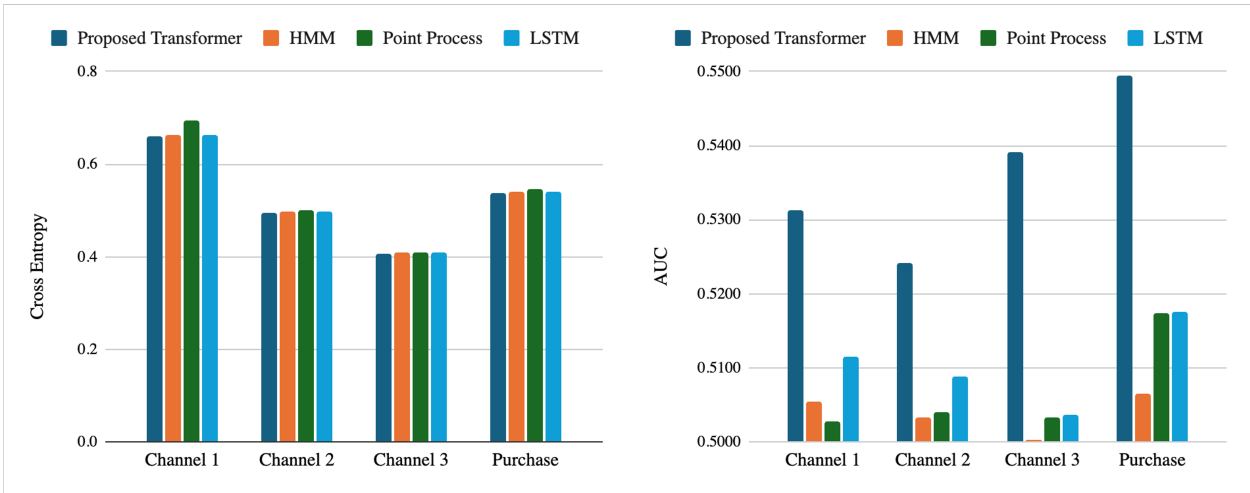


Figure 15: Model Comparisons under Mixture DGP

Simulating the time series patterns in the application data.

Lastly, we experiment with data where the time varying patterns are similar to the data we use in our applications. We focus on three channels (Direct, Natural Search, and Email)

in our applications along with the booking variable. Since the visit and purchase indicators are binary variables, these variables are modeled using a logistic regression that includes only a time fixed effect, represented as $y_{ct} = \text{logit}(\lambda_{ct})$, where λ_{ct} denotes the time fixed effect to be estimated for variable c. By examining the ACF and PACF plots of λ_{ct}, and also checking the AIC and BIC of ARMA models of different orders, we determine that an ARMA(2,2) process most accurately describes the patterns observed in the data. We fit an ARMA(2,2) model to each time series λ_{ct}, and simulate data from the coefficients estimated. More details on time series modeling and simulation can be found in Web Appendix F. The simulated data contains 10,000 customers across 100 periods under the same ARMA(2,2) process. This simulation simplifies the original data patterns by focusing solely on the autocorrelation structure, while disregarding any inter-channel correlations.

We compare the performance of the proposed transformer and other three models on the simulated ARMA(2,2) data. As seen in Figure 16, the transformer outperforms the all other models by a large margin. Although this is a simplified simulation of the pattern in the application data, it echoes the good performance of our transformer approach in the Application section.

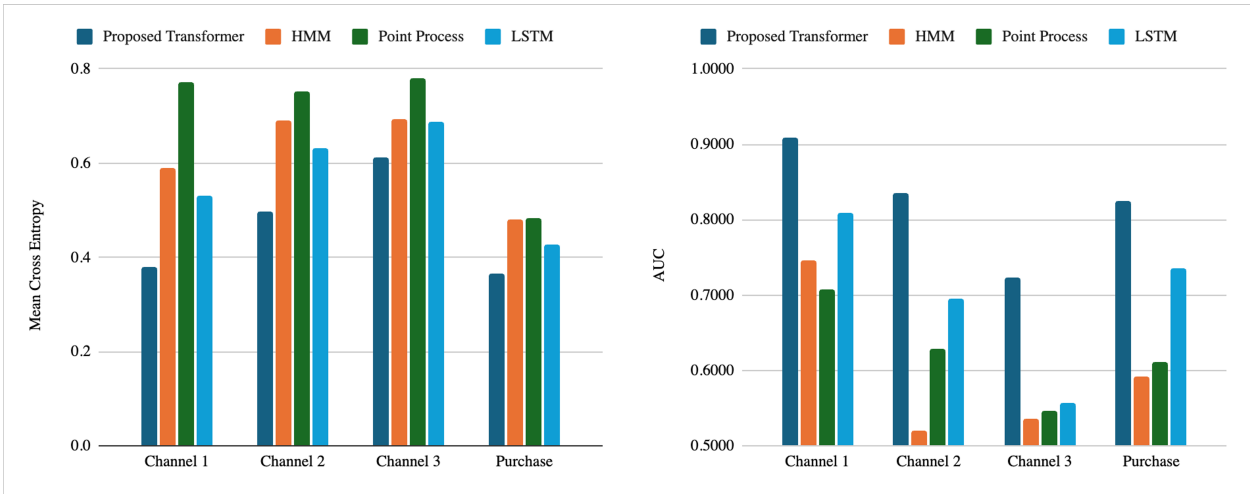


Figure 16: Model Comparisons under ARMA(2,2) DGP

Our experiments across various DGPs demonstrate that the transformer consistently outperforms all competing models under all conditions, while competing models excel only in

specific scenarios (e.g., HMM under HMM DGP, Point Process under Point Process DGP). LSTMs handle linear high-order dependencies better when sample sizes are small, highlighting the transformer's need for larger datasets to capture such relationships and establishing its performance boundaries. However, for complex real-world data patterns, the transformer surpasses all competing models.

DISCUSSION AND CONCLUSION

In this paper, we apply AI for modeling customer journeys using a transformer-based approach. Just as the transformer technology generates the next word or vocabulary in the LLM context based on learning a large corpus of text, we use the transformer technology to predict the next touchpoint in the customer journey using the data to learn the self-attention patterns. We show through simulations and empirical analyses that our transformer-based model is superior to competing models such as LSTM, HMM and Point process models in terms of predictions as well as providing unique insights into the heterogeneity characterizing customer journeys through the multi-head self-attention patterns. The empirical application highlights how managers can use the features of the model to identify high-potential customers and could plan the timing of interventions both at the individual and cohort levels if appropriate data were available.

From a modeling perspective, our approach captures the relationships between past and current visits using multiple attention heads that identify latent self-attention patterns. We describe how the model captures self-attention patterns reflecting population-level trends as well as the heterogeneity in relationships between touchpoints within individual customer journeys. The model assigns varying weights to each head, reflecting the unique aspects of each user's journey. By accounting for customer-specific heterogeneity, the model achieves more accurate predictions, enabling more precise targeting and thus outperforming existing approaches. Currently, no effective tools exist for leveraging data to develop precise targeting strategies at the touchpoint level. Our model addresses this gap by providing actionable

insights for such tactics. The examples highlighted here emphasize the model's value to managers. By using our approach, managers can make informed decisions about targeting and timing across various instruments, such as search, display, and email, enhancing the effectiveness of their marketing strategies with the availability of complementary data on marketing mix decisions.

Although we do not illustrate it in the present application, our model can handle a large number of distinct event types along a customer journey with ease (over 3900 events as we have done in an healthcare setting). For example, in the context of customer relationships with service firms, our methodology can handle different types of customer interactions within the firm, outside the firm, across channels and with a customer service team, etc., and identify critical incidences along the customer journey with the firm that impact churn or retention outcomes significantly.

Our paper complements recent effort in marketing in using LLM technologies for marketing research applications (e.g., Arora, Chakraborty, and Nishimura 2024; Angelopoulos, Lee, and Misra 2024; Gabel and Ringel 2024; Brand, Israeli, and Ngwe 2023). Just as GPT implementations use the left-to-right transformers to generate the next word in a sentence, we can use the transformers to generate customer journeys of hypothetical customers which can be used to test and simulate various interventions plans. They can also be used to plan field experiments based on these scenarios. Another application of our model is identifying customers with similar propensities to convert at a specific point in time based on their customer journeys up to that moment, enabling their use in test and control groups for A/B testing in e-mails, display ads and other marketing interventions.

At the core, the methodology we propose is predictive and descriptive in nature. Many of the touchpoints seen in a customer journey are initiated by customers and firms reacting to them with their own interventions and are, as such, endogenous in nature. Given such limitations, our methodology tries to predict the next touchpoints and actions conditional on the touchpoints that have occurred thus far. In this context, our methodology shares the

same spirit as that of VAR modeling (Dekimpe and Hanssens 1999) which relates outcome variables to lagged variables capturing the endogenous relationships, without trying to correct for endogeneity or debias the estimates. If there is sufficient variation in firm-initiated actions and such data were available the firm can use these predictions to help make decisions on who to target and when to target. As we highlighted before, with data from an ensemble of experiments, we could learn optimal policy with a sequence of interventions along the customer journey based (e.g., Song and Sun 2024). When comparing the performance of our proposed transformer model with HMM and Point Process models, we are essentially contrasting a non-parametric estimation method with a Bayesian estimation method. While this comparison may not be entirely fair from a methodological standpoint, our focus on the models' predictive abilities justifies an outcome-driven perspective. Despite the above limitations, the modeling framework that we propose illustrates the power of AI for marketing applications (Deveau, Griffin, and Reis 2023). Ours is one of the first applications to illustrate how such AI models can be used to extract relevant marketing insights from quantitative data. We trust this work will inspire many such applications going forward.

REFERENCES

Abhishek, Vibhanshu, Peter Fader, and Kartik Hosanagar (2012), "The Long Road to Online Conversion: A Model of Multi-Channel Attribution," *SSRN Electronic Journal* http://www.ssrn.com/abstract=2158421.

Angelopoulos, Panagiotis, Kevin Lee, and Sanjog Misra (2024), "Value Aligned Large Language Models," *SSRN Electronic Journal* https://www.ssrn.com/abstract=4781850.

Arora, Neeraj, Ishita Chakraborty, and Yohei Nishimura (2024), "EXPRESS: AI-Human Hybrids for Marketing Research: Leveraging LLMs as Collaborators," *Journal of Marketing* https://journals.sagepub.com/doi/10.1177/00222429241276529.

Brand, James, Ayelet Israeli, and Donald Ngwe (2023), "Using GPT for Market Research," *SSRN Electronic Journal* https://www.ssrn.com/abstract=4395751.

Carlson, Keith, Praveen K. Kopalle, Allen Riddell, Daniel Rockmore, and Prasad Vana (2023), "Complementing human effort in online reviews: A deep learning approach to automatic content generation and review synthesis," *International Journal of Research in Marketing*, 40 (1), 54–74 https://linkinghub.elsevier.com/retrieve/pii/S016781162200009X.

Caruana, Rich (1997), "Multitask Learning," *Machine Learning*, 28 (1), 41–75 http://link.springer.com/10.1023/A:1007379606734.

Castro, Javier, Daniel Gómez, and Juan Tejada (2009), "Polynomial calculation of the Shapley

value based on sampling," *Computers & Operations Research*, 36 (5), 1726–1730 https://linkinghub.elsevier.com/retrieve/pii/S0305054808000804.

Chen, Mia Xu, Benjamin N. Lee, Gagan Bansal, Yuan Cao, Shuyuan Zhang, Justin Lu, Jackie Tsay, Yinan Wang, Andrew M. Dai, Zhifeng Chen, Timothy Sohn, and Yonghui Wu "Gmail Smart Compose: Real-Time Assisted Writing," "Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining," pages 2287–2295, Anchorage AK USA: ACM (2019) https://dl.acm.org/doi/10.1145/3292500.3330723.

Cheng, Haibin, Pang-Ning Tan, Jing Gao, and Jerry Scripps "Multistep-Ahead Time Series Prediction," David Hutchison, Takeo Kanade, Josef Kittler, Jon M. Kleinberg, Friedemann Mattern, John C. Mitchell, Moni Naor, Oscar Nierstrasz, C. Pandu Rangan, Bernhard Steffen, Madhu Sudan, Demetri Terzopoulos, Dough Tygar, Moshe Y. Vardi, Gerhard Weikum, Wee-Keong Ng, Masaru Kitsuregawa, Jianzhong Li, and Kuiyu Chang, editors, "Advances in Knowledge Discovery and Data Mining," Vol. 3918., pages 765–774, Berlin, Heidelberg: Springer Berlin Heidelberg (2006) http://link.springer.com/10.1007/11731139_89, series Title: Lecture Notes in Computer Science.

Crawshaw, Michael "Multi-Task Learning with Deep Neural Networks: A Survey," (2020) http://arxiv.org/abs/2009.09796, arXiv:2009.09796.

Dai, Zihang, Zhilin Yang, Yiming Yang, Jaime Carbonell, Quoc V. Le, and Ruslan Salakhutdinov (2019), "Transformer-XL: Attentive Language Models Beyond a Fixed-Length Context," https://arxiv.org/abs/1901.02860, publisher: arXiv Version Number: 3.

Danaher, Peter J. and Harald J. van Heerde (2018), "Delusion in Attribution: Caveats in Using Attribution for Multimedia Budget Allocation," *Journal of Marketing Research*, 55 (5), 667–685 http://journals.sagepub.com/doi/10.1177/0022243718802845.

Dang, Chu (Ivy), Raluca Ursu, and Pradeep K. Chintagunta (2020), "Search Revisits," *SSRN Electronic Journal* https://www.ssrn.com/abstract=3626451.

Dekimpe, Marnik G. and Dominique M. Hanssens (1999), "Sustained Spending and Persistent Response: A New Look at Long-Term Marketing Profitability," *Journal of Marketing Research*, 36 (4), 397 https://www.jstor.org/stable/3151996?origin=crossref.

Dekimpe, Marnik G. and Dominique M. Hanssens (2024), "Persistence Modeling in Marketing: Descriptive, Predictive, and Normative Uses," *Australasian Marketing Journal* http://journals.sagepub.com/doi/10.1177/14413582231222311.

Deveau, Richelle, Sonia Joseph Griffin, and Steve Reis (2023), "AI-powered marketing and sales reach new heights with generative AI," *Mckinsey & Company*.

Dew, Ryan and Asim Ansari (2018), "Bayesian Nonparametric Customer Base Analysis with Model-Based Visualizations," *Marketing Science*, 37 (2), 216–235 https://pubsonline.informs.org/doi/10.1287/mksc.2017.1050.

Farrell, Max H., Tengyuan Liang, and Sanjog Misra (2021), "Deep Neural Networks for Estimation and Inference," *Econometrica*, 89 (1), 181–213 https://www.econometricsociety.org/doi/10.3982/ECTA16901.

Gabel, Sebastian and Daniel Ringel (2024), "The Market Basket Transformer: A New Foundation Model for Retail," *SSRN Electronic Journal* https://www.ssrn.com/abstract=4335141.

Gal, Yarin and Zoubin Ghahramani "Dropout as a Bayesian approximation: representing model uncertainty in deep learning," "Proceedings of the 33rd International Conference on International Conference on Machine Learning - Volume 48," ICML'16, pages 1050–1059, JMLR.org (2016) Place: New York, NY, USA.

Goić, Marcel, Kinshuk Jerath, and Kirthi Kalyanam (2021), "The roles of multiple channels in predicting website visits and purchases: Engagers versus closers," *International Journal of Research in Marketing*, page S0167811621001166 https://linkinghub.elsevier.com/retrieve/pii/S0167811621001166.

Gordon, Mitchell A., Kevin Duh, and Nicholas Andrews "Compressing BERT: Studying the Effects of Weight Pruning on Transfer Learning," (2020) https://arxiv.org/abs/2002.08307, version Number: 2.

Huang, Ming-Hui and Roland T. Rust (2024), "The Caring Machine: Feeling AI for Customer Care," *Journal of Marketing*, page 00222429231224748 https://journals.sagepub.com/doi/10.1177/00222429231224748.

Kingma, Diederik P. and Jimmy Ba "Adam: A Method for Stochastic Optimization," (2014) https://arxiv.org/abs/1412.6980, version Number: 9.

Lemon, Katherine N. and Peter C. Verhoef (2016), "Understanding Customer Experience Throughout the Customer Journey," *Journal of Marketing*, 80 (6), 69–96 http://journals.sagepub.com/doi/10.1509/jm.15.0420.

Li, Hongshuang (Alice) and P.K. Kannan (2014), "Attributing Conversions in a Multichannel Online Marketing Environment: An Empirical Model and a Field Experiment," *Journal of Marketing Research*, 51 (1), 40–56 http://journals.sagepub.com/doi/10.1509/jmr.13.0050.

Li, Hongshuang (Alice) and Liye Ma (2020), "Charting the Path to Purchase Using Topic Models," *Journal of Marketing Research*, 57 (6), 1019–1036 http://journals.sagepub.com/doi/10.1177/0022243720954376.

Manchanda, Puneet, Peter E. Rossi, and Pradeep K. Chintagunta (2004), "Response Modeling with Nonrandom Marketing-Mix Variables," *Journal of Marketing Research*, 41 (4), 467–478 https://journals.sagepub.com/doi/10.1509/jmkr.41.4.467.47005.

Mela, Carl F., Sunil Gupta, and Donald R. Lehmann (1997), "The Long-Term Impact of Promotion and Advertising on Consumer Brand Choice," *Journal of Marketing Research*, 34 (2), 248–261 http://journals.sagepub.com/doi/10.1177/002224379703400205.

Michel, Paul, Omer Levy, and Graham Neubig "Are sixteen heads really better than one?," "Proceedings of the 33rd International Conference on Neural Information Processing Systems," Red Hook, NY, USA: Curran Associates Inc. (2019).

Moe, Wendy W. (2003), "Buying, Searching, or Browsing: Differentiating Between Online Shoppers Using In-Store Navigational Clickstream," *Journal of Consumer Psychology*, 13 (1-2), 29–39 http://doi.wiley.com/10.1207/S15327663JCP13-1&2_03.

Netzer, Oded, James M. Lattin, and V. Srinivasan (2008), "A Hidden Markov Model of Customer Relationship Dynamics," *Marketing Science*, 27 (2), 185–204 http://pubsonline.informs.org/doi/abs/10.1287/mksc.1070.0294.

Schipper, Tijmen M., Kars Mennens, Paul Preenen, Menno Vos, Marieke Van Den Tooren, and Nienke Hofstra (2023), "Interorganizational Learning: A Conceptualization of Public-Private Learning Communities," *Human Resource Development Review*, 22 (4), 494–523 http://journals.sagepub.com/doi/10.1177/15344843231198361.

Schmittlein, David C., Donald G. Morrison, and Richard Colombo (1987), "Counting Your Customers: Who-Are They and What Will They Do Next?," *Management Science*, 33 (1), 1–24 https://pubsonline.informs.org/doi/10.1287/mnsc.33.1.1.

Siami-Namini, Sima, Neda Tavakoli, and Akbar Siami Namin "A Comparison of ARIMA and LSTM in Forecasting Time Series," "2018 17th IEEE International Conference on Machine

Learning and Applications (ICMLA)," pages 1394–1401, Orlando, FL: IEEE (2018) https://ieeexplore.ieee.org/document/8614252/.

Song, Yicheng and Tianshu Sun (2024), "Ensemble Experiments to Optimize Interventions Along the Customer Journey: A Reinforcement Learning Approach," *Management Science*, 70 (8), 5115–5130 https://pubsonline.informs.org/doi/10.1287/mnsc.2023.4914.

Sundararajan, Mukund, Ankur Taly, and Qiqi Yan (2017), "Axiomatic Attribution for Deep Networks," https://arxiv.org/abs/1703.01365, publisher: arXiv Version Number: 2.

Valendin, Jan, Thomas Reutterer, Michael Platzer, and Klaudius Kalcher (2022), "Customer base analysis with recurrent neural networks," *International Journal of Research in Marketing*, 39 (4), 988–1018 https://linkinghub.elsevier.com/retrieve/pii/S0167811622000180.

Vaswani, Ashish, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Łukasz Kaiser, and Illia Polosukhin "Attention is all you need," "Proceedings of the 31st International Conference on Neural Information Processing Systems," NIPS'17, pages 6000–6010, Red Hook, NY, USA: Curran Associates Inc. (2017) Event-place: Long Beach, California, USA.

Venkatraman, Arun, Martial Hebert, and J.. Bagnell (2015), "Improving Multi-Step Prediction of Learned Time Series Models," *Proceedings of the AAAI Conference on Artificial Intelligence*, 29 (1) https://ojs.aaai.org/index.php/AAAI/article/view/9590.

Wedel, Michel and P.K. Kannan (2016), "Marketing Analytics for Data-Rich Environments," *Journal of Marketing*, 80 (6), 97–121 http://journals.sagepub.com/doi/10.1509/jm.15.0413.

Zantedeschi, Daniel, Eleanor McDonnell Feit, and Eric T. Bradlow (2017), "Measuring Multichannel Advertising Response," *Management Science*, 63 (8), 2706–2728 https://pubsonline.informs.org/doi/10.1287/mnsc.2016.2451.

Zhou, Yichao, Shaunak Mishra, Jelena Gligorijevic, Tarun Bhatia, and Narayan Bhamidipati "Understanding Consumer Journey using Attention based Recurrent Neural Networks," "Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining," pages 3102–3111, Anchorage AK USA: ACM (2019) https://dl.acm.org/doi/10.1145/3292500.3330753.

AI for Customer Journeys: A Transformer Approach

Table of Contents

Web Appendix A: Model Details and Technical Specifications

In Table W1, we compare the proposed transformer-based methodology with existing marketing models, such as HMM, Point-Process Models, and the previously best-performing machine learning model – LSTM – across various dimensions for modeling the customer journey. In this appendix we provide additional technical details on the proposed transformer model and other benchmark models, as well as models we use for simulation exercise.

Other Components of Transformer

Residual connection, layer norm and feed-forward neural network. Following the multi-head self-attention layer, the output of the attention layer is added to the original input embedding in a step called the residual connection. The goal of the residual connection is to give higher level layers direct access to information from lower layers. Next, the summed-up vector is normalized, also known as the layer norm process. These two steps are performed after each sub-layer, which can be jointly expressed as

$$\tilde{\mathbf{z}} = LayerNorm\left(\mathbf{z} + \tilde{\mathbf{x}}\right), \tag{W1}$$

where $\tilde{\mathbf{x}}$ is the input embedding of the self-attention layer, and $\mathbf{z}$ is the output of the self-attention layer, and the layer norm is a function that normalizes each input embedding vector and rescales it. For input embedding at each position i, that is, each row i of the matrix $\mathbf{z} + \tilde{\mathbf{x}}$, or $z_i + \tilde{x}_i$, the layer norm performs

$$LayerNorm(z_i + \tilde{x}_i) = \gamma \frac{(z_i + \tilde{x}_i - \mu)}{\sigma} + \beta. \tag{W2}$$

The μ and σ are the mean and standard deviation of the elements of the vector $z_i + \tilde{x}_i$. After normalizing the vector, the layer norm rescales it to a suitable range, using two "learnable" parameters γ and β. By scaling the embedding vectors to a suitable range, the layer norm

Table W1: Model Comparisons

	HMM	Point-Process	LSTM	Transformers
Modeling Customer Journey	Representing touchpoint sequences as observable events tied to hidden states, capturing underlying dynamics.	Modeling touchpoint occurrences in continuous time, aiming to capture event intensity and timing based on past outcomes of touchpoints.	Learning data patterns and dependencies using memory cells, updated sequentially by gate functions to incorporate new touchpoint information and selectively forget past states.	Leverages self-attention mechanisms to assess the significance of different input data segments, enabling parallel processing and capturing complex dependencies without relying on sequential processing.
Estimation	Bayesian/MCMC Methods.	MLE/Bayesian.	Gradient descent-based optimization methods.	Gradient descent-based methods.
Parallel Processing	Word-by-word Sequential Processing,	Word-by-word Sequential Processing.	Word-by-word Sequential Processing.	Whole-sentence parallel processing.
Ability to Handle Large Number of Unique Touchpoints	Suited for datasets with a limited number of touchpoints, typically in single digits.	Suitable for datasets with a small number of touchpoints, usually within single digits.	Capable of handling extensive datasets with thousands of unique touchpoints.	Equipped to process large datasets with thousands of unique touchpoints efficiently.
Modeling Touchpoint Relationships	Parameterizing state transition and emission probabilities to indirectly model touchpoint relationships through hidden states.	The arrival rate through a touchpoint is a function of touchpoint fixed effects and lag effects of previous touchpoints.	Indirectly captures touchpoint relationships through memory cell states updated via gate functions.	Captures touchpoint relationships through attention weights across multiple heads, emphasizing the connection between each touchpoint and previous ones.
Model Training/Estimation Time	Several days	Several days	Several hours	Several hours

Note: Estimation time is based on data used in the application section.

process improves training performance in deep neural networks by facilitating the gradient based training.

After the attention sub-layer and the layer norm operation in Equation W1, the output embedding $\tilde{\mathbf{z}}$ goes through a feed-forward neural network (FFNN) sub-layer.

$$\mathbf{y} = FFNN(\tilde{\mathbf{z}}) = W_2 \max(0, W_1\tilde{\mathbf{z}} + b_1) + b_2 \tag{W3}$$

The FFNN has a sandwich structure. It consists of two affine transformations with a Rectified Linear Unit (ReLU) activation function in between. The first affine transformation yields $(W_1\tilde{\mathbf{z}} + b_1)$, where W_1 and b_1 are the parameters. Next, it goes through the ReLU activation function $\max(0, x)$. Finally, another affine transformation is performed with a different set of parameters W_2 and b_2. Feed-forward neural networks with similar structures are widely used in many neural network models, with small variations in between. These networks help extract useful information for prediction from the input. The layer norm was performed again after the FFNN sublayer, which outputs

$$\tilde{\mathbf{y}} = LayerNorm\left(\mathbf{y} + \tilde{\mathbf{z}}\right). \tag{W4}$$

Linear and sigmoid layer. The model uses embedding output from the encoder at position t to predict the outcome of the next period at $t + 1$. A linear layer is used to project embedding $\tilde{y}_i$ to single dimension and then followed by a sigmoid layer to calculate the probability, denoted by p_{t+1} (See 4 in Figure 1 in the main text).

$$\begin{aligned} y_t^* &= W^C \tilde{y}_t, \\ p_{t+1} &= \frac{\exp(y_t^*)}{1 + \exp(y_t^*)}. \end{aligned} \tag{W5}$$

The same process is repeated for every interaction type s (we drop the s when describing the processes in encoders).

In the model training process, the outcome for each position is already known and the model minimizes a loss function based on its guess and the true outcomes. We use cross entropy for the loss function, which we describe in details in the following section.

Model Training Details for Transformer and LSTM

The transformer and LSTM models trained on the hospitality data require the most computation resources. To implement the models, we mainly use the PyTorch library (version 1.12). We have attached the code for the main parts of the two models at the end of the web appendix. The transformer and LSTM are trained on a Nvidia RTX A8000 GPU.

Hyperparameter searching. The transformer's hyperparameters to be specified by the researcher include number of heads, number of encoder layers, dimensionality of the input embedding vectors and number of nodes in the feed-forward neural network. The LSTM's hyperparameters include number of recurrent layers, dimensionality of the embedding vectors and number of features in the hidden states. Because of the large amount of parameters to be determined, it would be inefficient to do a grid search that trains a model on all combinations of parameters. We use the Ray Tune software (Liaw et al. 2018) to tune all hyperparameters in transformer and LSTM. The Ray Tune will randomly sample from the parameter search space and train the model. It also has a scheduler that stops the training early for bad parameter specifications. Based on the loss function, the best parameter combinations are returned. We list all hyperparameters for transformer tuning below. We run 300 trials with the `HyperOpt` search algorithm and early-stopping scheduler `ASHA`.

- Embedding size {50, 100, 200};
- Number of features in the hidden states of FFNN {100, 200};
- Number of layers {3, 4, 5};
- Number of heads {2, 3, 4};

- Learning rate of SGD: Uniform$(0, 1)$;
- Learning rate of Adam: Log-Uniform$(10^{-6}, 10^{-4})$.

After conducting the trials, we selected an embedding size of 100, with 100 features in the hidden states, 4 layers, and 4 attention heads for the transformer. We found that the learning rate is the most critical hyperparameter during training. For SGD, we used a learning rate of 0.2, while for Adam, the learning rate was set to 1×10^{-5}.

Loss function and loss weighting. Customer journey data, when organized as time series, is often very sparse, with most time intervals showing no recorded activity for an individual customer. We found that while the imbalanced data is not a problem for transformer, it greatly impacts LSTM's performance. The challenge of class imbalance on the performance of machine learning models is well documented in literature (Kubat and Matwin 1997; Kaur, Pannu, and Malhi 2020; Johnson and Khoshgoftaar 2019). To address this issue, following the common practice in literature (Fernando and Tsokos 2022), we apply weights to the positive class in the loss function in the training sample. Because our dependent variables are all binary, we use the binary cross-entropy (BCE) loss function. Let N_{pos} and N_{neg} represent the number of positive and negative samples in dependent variable to be predicted, respectively. To balance the loss contribution of each class, we calculate weights for the positive class as $\text{weight}_{\text{pos}} = \frac{N_{\text{neg}}}{N_{\text{pos}}}$. The weighted BCE loss becomes

$$\text{Weighted BCE Loss} = -\left(\text{weight}_{\text{pos}} \cdot y \cdot \log(p) + (1 - y) \cdot \log(1 - p)\right),$$

where y denotes the actual outcome (0 or 1), and p is the predicted probability for $y = 1$. After weighting the positive class, the loss function will act as if the dataset contains equal amount of positive samples and negative samples. In frameworks like PyTorch, these weights can be passed directly as a tensor in the `BCEWithLogitsLoss` function, allowing the model to focus more on the minority class during training. We calculate the average weighted BCE loss across all samples in the training or validating set to get the training loss and validation

loss respectively.

Probability calibration. Applying class weights adjusts the loss function to prioritize the minority class (positive class in our case), helping the model learn to correctly classify both classes. However, these weights distort the raw prediction probabilities, making them less representative of the actual likelihood of each class. This can lead to biased probabilities, often overestimating the likelihood of the minority class, especially if the weighting is substantial. Imagine if the $weight_{pos}$ is sufficiently high, the model will overestimate p to minimize the loss, because the cost to do so, i.e., the decrease of $(1-y)\log(1-p)$ for the negative class, is sufficiently low. To correct this bias brought by class weighting, we calibrate the output probability according to methods proposed in the literature (Chen et al. 2018; Tian et al. 2020; Caplin, Martin, and Marx 2022), which uses Bayes theorem to calculate the calibrated posterior probability. For each target variable (purchase, channel visit, etc.), suppose P_0 is the variable prevalence (true positive rate) in the real data, for each observation i, $OR_i = p_i/(1-p_i)$ is the odds ratio of the output probability p_i, the calibrated probability is given by

$$P(D_i = 1|y) = \frac{P_0 \cdot OR_i}{P_0 \cdot OR_i + (1 - P_0)}.$$

Parameter optimization. We train the transformer using mini-batch gradient descent, which is commonly used in parameter optimization (Khirirat, Feyzmahdavian, and Johansson 2017; Li et al. 2014). The mini-batch gradient descent updates parameters after processing a batch of data. We found that during the model training on the hospitality data, increasing the batch size (i.e., number of samples in the batch) reduces the training time without hurting the training performance, as long as the loss function converges. Therefore, we choose a batch size of 200, which is the largest batch size possible for the computational environment. For transformer training, we use the SGD class embedded in the PyTorch library with the specified batch size to train most of the parameters. After testing different optimizers, we found that SGD with mini-batches produces the most stable training process,

but is not efficient in optimizing the mixture head weights for each data point in the training sample. On the other hand, the Adam optimizer adjusts the mixture head weights more effectively but tends to overfit the data, leading to a large discrepancy between the training and validating performance. Therefore, we use a mixed-optimizer strategy. We divide the model parameters into two groups: the mixture head weights and other parameters. The head weights are optimized by the Adam, and the rest of model parameters are optimized by the SGD. All parameters are updated at the same time when processing each batch of samples. The learning rates of the two optimizers are tuned together with other hyperparameters. On top of the optimizer, we make the learning rate of the SGD decay with the increase of epochs (multiplied by 0.9 every 5 epochs). This helps the model settle into a good minimum by taking smaller, more stable steps, and reduces oscillations around the optimum and helps the model converge more reliably.

Figure W1 shows the training and validation loss over the number of epochs during the transformer model's training process. Our model is different from the majority of the machine learning models in the way that the mixture head weights are individually optimized for each customer in the training sample. For validating sample, we use the average weights across all customers to make predictions and calculate loss function. As one might expect, beyond a certain point, further optimizing the individual's weights in the training sample will hurt the out-of-sample validating performance. But choosing the model with the best out-of-sample performance will not adequately account for the customer heterogeneity in the distribution of heads. Therefore, we use the mean of training and validating loss (the green line in Figure W1) as the stopping criterion. The mean of training and validating loss converges after around 250 epochs. We stop the model training after 300 epochs. Training 300 epochs takes about 18 hours in total.

The LSTM model is trained only with the Adam optimizer. We found that the validation loss of the Adam optimizer shows more stable decline than that of the SGD. We also apply the learning rate decay over the Adam optimizer. Figure W2 shows the training and validation

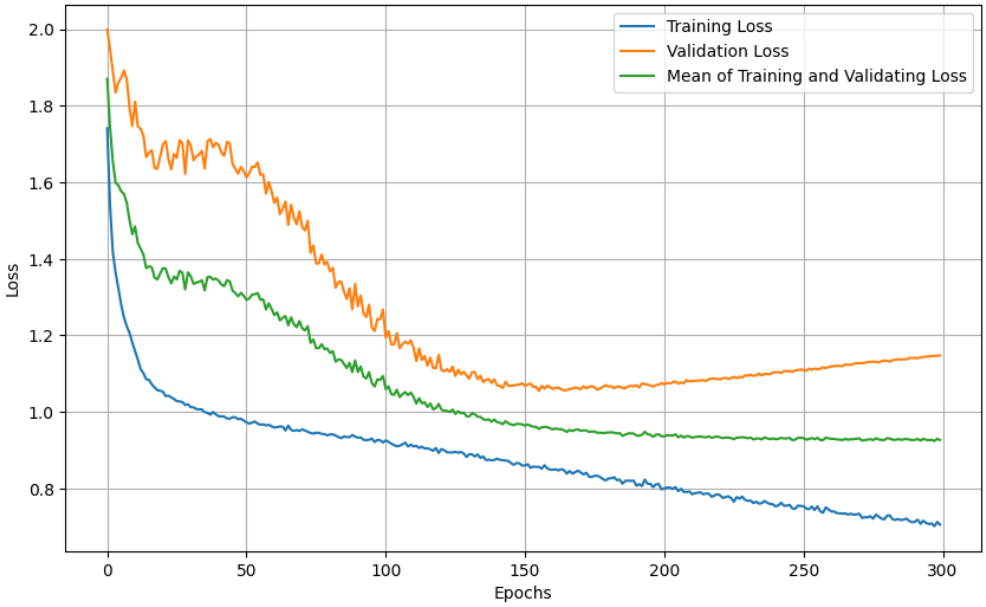


Figure W1: Transformer's Training and Validation Loss over Number of Epochs

loss over the number of epochs for LSTM. We run 300 epochs and choose the model with the lowest validation loss during the training process.

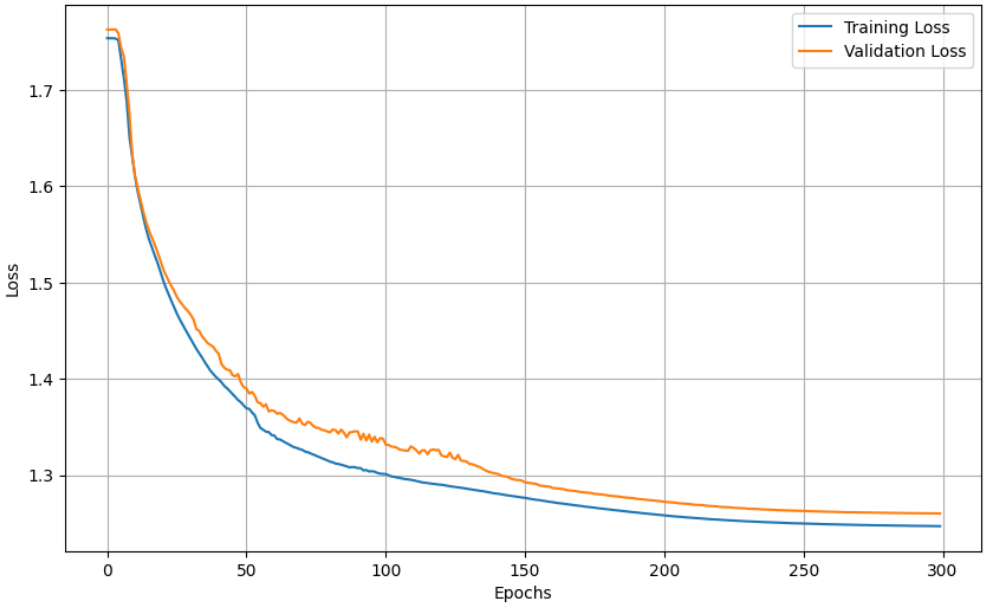


Figure W2: LSTM's Training and Validation Loss over Number of Epochs

Estimation of the HMM and Point Process

The HMM and Point Process models are estimated with the Hamiltonian Monte Carlo (HMC) algorithm, which is implemented in the Stan software. The Stan code for the two models are also attached at the end of the appendix. we use the `CmdStanPy` library in Python as the interface. The program is run on a AWS (Amazon Web Services) machine with a 4-core Intel(R) Xeon(R) Platinum 8259CL CPU (2.50GHz) and a 64 GiB memory.

Training and hold-out samples. The sampling time of both Bayesian models depend on the sample size. To control the training time, We randomly sample 2,000 users from the population as the training sample, and another 2,000 users as the hold-out sample.

Model prior. The parameters in the HMM model include the initial state probability ρ_{0s} for each channel s, the channel specific state transition matrix $\rho_{css'}$ for channel c and between state s, s'; the purchase coefficients α_s under state s, and channel visit coefficients λ_{cs} for each channel c at state s. The priors for above parameters are

$$\begin{aligned} \rho_{0s}, \rho_{css'} &\sim Dirichlet\,(1)\,, \\ \alpha_s, \lambda_{cs} &\sim Normal\,(0,1)\,. \end{aligned} \tag{W6}$$

The Poisson point process model has two parts – an arrival rate model for channel visit (Equation 6 in the main text), and a logistic model for purchase decision (Equation 7 in the main text). For the channel visit arrival rate, Equation 6 has baseline parameter μ_0, the attractiveness of last visited channel $\alpha_{c'}$ and current channel β_c, the inertia parameter θ, and the impact of the cumulative inventory of visits ρ_c for each channel c. For the logistic purchase model, parameters include a intercept ϕ_0 and a coefficient ϕ_c for the inventory of visits of each channel. The priors for these parameters are

$$\begin{aligned} \mu_0 &\sim Gamma\,(1, 0.5)\,, \\ \alpha_{c'}, \beta_c, \theta, \rho_c, \phi_0, \phi_c &\sim Normal\,(0,1)\,. \end{aligned} \tag{W7}$$

Model training and diagnostics. For both of the models, we run 4 chains, each having 500 iterations, with the first 250 warming-up iterations discarded. We use the Stan default values for all other hyperparameters of the HMC algorithm. We found that the HMM has multimodality problem in the posterior distribution, meaning that the HMC is trapped at local optimum, and it cannot be addressed through tuning the hyperparameters. Therefore, we run the HMM model multiple times from different initiations, and choose the initiation values that generate the highest log likelihood. We check the model diagnostics and ensure that all parameters have converged with a R-hat less than 1.05, indicating good chain mixing (Vehtari et al. 2021). The trace plots for some parameters are shown below in Figure W3 and Figure W4. Different chains are marked by different line styles in the trace plots. For a complete table of all parameter estimates, credible intervals and diagnostic statistics, please refer to Table W2-W7.

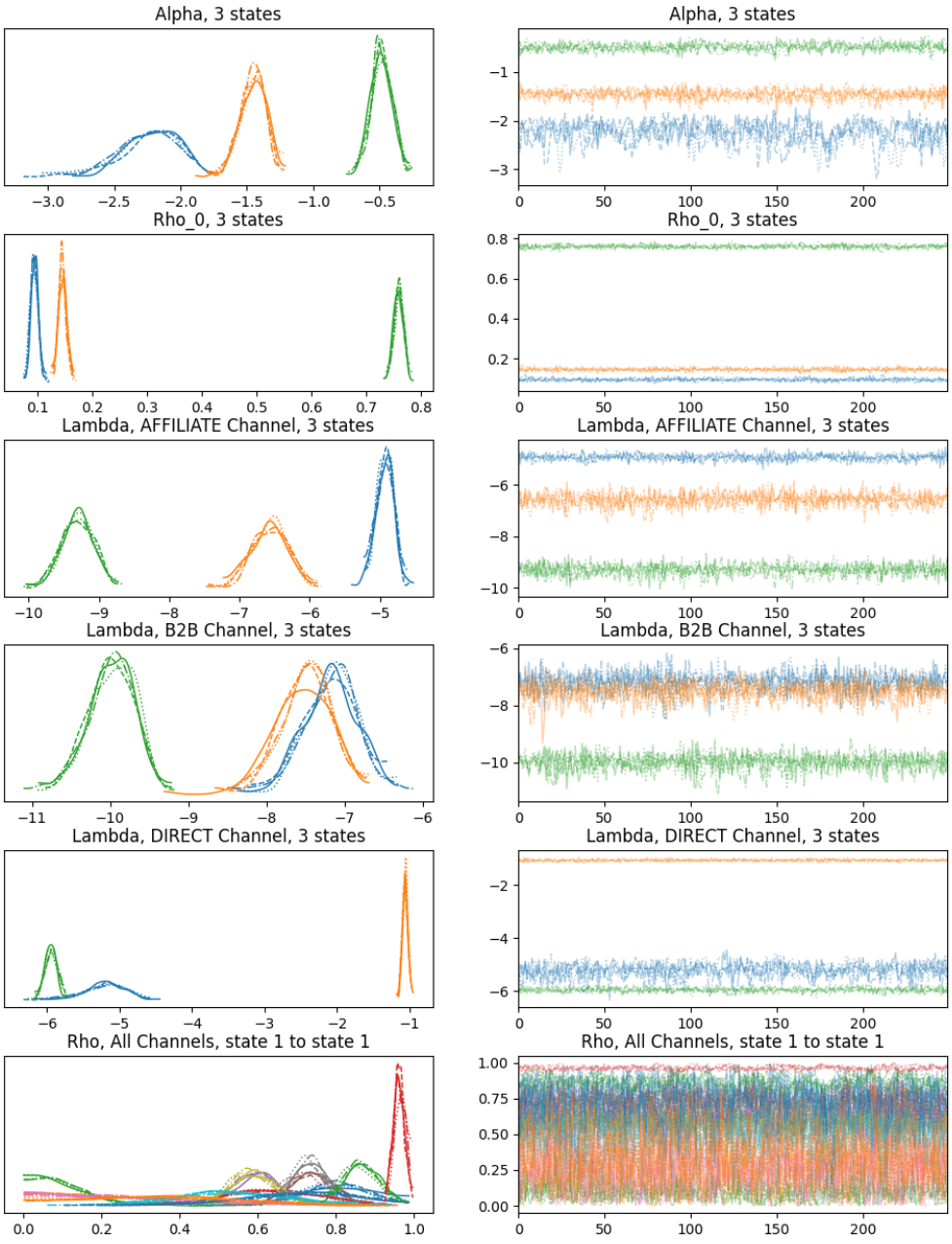


Figure W3: Trace Plots of HMM Model

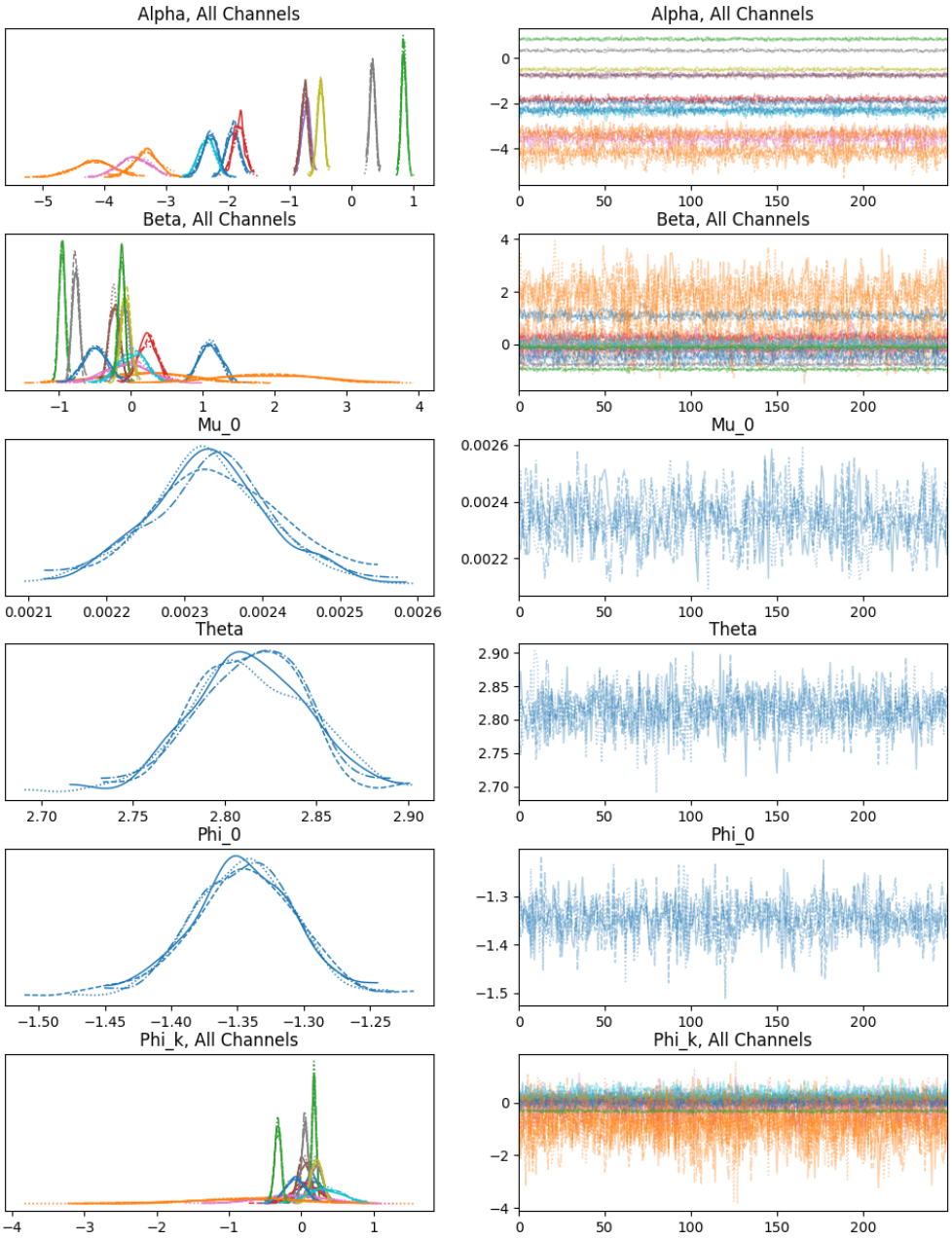


Figure W4: Trace Plots of Point Process Model

Table W2: HMM Parameter Estimates, Part 1

Variables	Mean	SD	HDI 2.5%	HDI 97.5%	ess_bulk	ess_tail	R_hat
Purchase Estimates: α_s							
State 1	-2.248	0.229	-2.687	-1.850	282	349	1.02
State 2	-1.456	0.103	-1.674	-1.270	805	653	1.00
State 3	-0.483	0.083	-0.640	-0.330	793	587	1.00
Channel Visit Estimates: λ_{cs}							
AFFILIATE, state1	-4.926	0.127	-5.183	-4.693	1056	674	1.00
AFFILIATE, state2	-6.571	0.268	-7.051	-6.033	1137	660	1.01
AFFILIATE, state3	-9.311	0.235	-9.790	-8.905	1056	779	1.00
B2B, state1	-7.176	0.356	-7.797	-6.422	1585	551	1.00
B2B, state2	-7.502	0.342	-8.173	-6.848	1448	564	1.00
B2B, state3	-9.984	0.304	-10.617	-9.463	1404	639	1.01
DIRECT, state1	-5.205	0.286	-5.819	-4.707	750	597	1.00
DIRECT, state2	-1.061	0.037	-1.130	-0.990	1307	828	1.00
DIRECT, state3	-5.945	0.086	-6.100	-5.770	736	684	1.00
DISPLAY, state1	-5.162	0.144	-5.461	-4.903	1118	672	1.00
DISPLAY, state2	-6.396	0.248	-6.918	-5.933	1584	664	1.01
DISPLAY, state3	-8.755	0.179	-9.083	-8.381	1926	778	1.01
ECONFO AND PRE-ARRIVAL EMAIL, state1	-3.635	0.079	-3.792	-3.487	618	666	1.01
ECONFO AND PRE-ARRIVAL EMAIL, state2	-5.358	0.156	-5.672	-5.062	1429	866	1.01
ECONFO AND PRE-ARRIVAL EMAIL, state3	-8.320	0.194	-8.676	-7.922	1231	686	1.00
EMAIL, state 1	-3.680	0.078	-3.827	-3.523	651	646	1.00
EMAIL, state 2	-5.418	0.162	-5.738	-5.098	1246	761	1.00
EMAIL, state 3	-7.829	0.137	-8.079	-7.544	1499	731	1.00
EMERGING TECHNOLOGIES, state1	-6.948	0.316	-7.596	-6.350	1122	722	1.00
EMERGING TECHNOLOGIES, state2	-6.860	0.262	-7.379	-6.359	1753	753	1.00
EMERGING TECHNOLOGIES, state3	-10.115	0.332	-10.806	-9.518	1803	652	1.00
NATURAL SEARCH, state1	-1.918	0.049	-2.014	-1.824	305	678	1.01
NATURAL SEARCH, state2	-4.706	0.124	-4.957	-4.478	1296	840	1.00
NATURAL SEARCH, state3	-6.338	0.080	-6.508	-6.194	1250	665	1.00
PAID SEARCH, state1	-3.316	0.072	-3.455	-3.178	615	508	1.00
PAID SEARCH, state2	-5.823	0.194	-6.201	-5.450	1114	674	1.00
PAID SEARCH, state3	-7.486	0.119	-7.707	-7.259	1192	712	1.00
REFERRAL ENGINE, state1	-5.497	0.170	-5.842	-5.187	1332	608	1.00
REFERRAL ENGINE, state2	-6.895	0.279	-7.444	-6.373	1344	729	1.00
REFERRAL ENGINE, state3	-9.455	0.261	-9.922	-8.933	1133	718	1.01
RESLINK, state1	-6.028	0.211	-6.414	-5.649	949	786	1.00
RESLINK, state2	-7.247	0.356	-7.906	-6.545	995	432	1.00
RESLINK, state3	-8.636	0.172	-8.975	-8.291	1875	708	1.01
SOCIAL MEDIA, state1	-6.577	0.297	-7.184	-6.055	1177	695	1.00
SOCIAL MEDIA, state2	-7.110	0.349	-7.823	-6.507	1116	670	1.01
SOCIAL MEDIA, state3	-9.936	0.293	-10.480	-9.356	1114	616	1.01
UNPAID REFERRER, state1	-2.110	0.054	-2.215	-2.005	338	550	1.01
UNPAID REFERRER, state2	-4.596	0.100	-4.802	-4.414	1296	490	1.01
UNPAID REFERRER, state3	-8.256	0.193	-8.645	-7.918	979	712	1.00
Initial Probability: ρ_{0s}							
State 1	0.095	0.007	0.080	0.108	527	716	1.01
State 2	0.146	0.008	0.132	0.162	1374	622	1.01
State 3	0.760	0.009	0.740	0.776	702	711	1.00

Table W3: HMM Parameter Estimates, Part 2

Variables	Mean	SD	HDI 2.5%	HDI 97.5%	ess_bulk	ess_tail	R_hat
Transition Probability: $\rho_{css'}$							
AFFILIATE, state 1 to state 1	0.795	0.091	0.611	0.962	788	627	1.01
AFFILIATE, state 1 to state 2	0.033	0.033	0.000	0.097	1029	652	1.00
AFFILIATE, state 1 to state 3	0.172	0.089	0.000	0.322	740	746	1.01
AFFILIATE, state 2 to state 1	0.502	0.212	0.100	0.897	1588	689	1.00
AFFILIATE, state 2 to state 2	0.140	0.126	0.000	0.390	1579	548	1.00
AFFILIATE, state 2 to state 3	0.359	0.210	0.000	0.716	1156	734	1.00
AFFILIATE, state 3 to state 1	0.284	0.195	0.003	0.673	1076	440	1.01
AFFILIATE, state 3 to state 2	0.089	0.082	0.000	0.247	799	444	1.00
AFFILIATE, state 3 to state 3	0.627	0.203	0.239	0.976	1243	697	1.01
B2B, state 1 to state 1	0.254	0.191	0.001	0.638	1399	605	1.00
B2B, state 1 to state 2	0.324	0.221	0.001	0.735	1475	755	1.00
B2B, state 1 to state 3	0.422	0.233	0.005	0.816	1054	625	1.01
B2B, state 2 to state 1	0.282	0.226	0.000	0.727	2595	404	1.01
B2B, state 2 to state 2	0.338	0.246	0.000	0.789	1275	754	1.01
B2B, state 2 to state 3	0.380	0.242	0.001	0.813	1389	789	1.01
B2B, state 3 to state 1	0.251	0.209	0.000	0.670	1399	495	1.00
B2B, state 3 to state 2	0.346	0.228	0.005	0.775	1385	698	1.00
B2B, state 3 to state 3	0.403	0.241	0.002	0.823	1385	513	1.00
DIRECT, state 1 to state 1	0.105	0.085	0.000	0.284	956	461	1.00
DIRECT, state 1 to state 2	0.699	0.168	0.380	0.979	1138	882	1.00
DIRECT, state 1 to state 3	0.196	0.151	0.000	0.497	938	489	1.00
DIRECT, state 2 to state 1	0.001	0.001	0.000	0.003	1192	569	1.00
DIRECT, state 2 to state 2	0.774	0.018	0.737	0.811	840	476	1.00
DIRECT, state 2 to state 3	0.225	0.018	0.189	0.262	842	500	1.01
DIRECT, state 3 to state 1	0.012	0.009	0.000	0.028	981	550	1.00
DIRECT, state 3 to state 2	0.038	0.034	0.000	0.105	614	594	1.00
DIRECT, state 3 to state 3	0.951	0.035	0.882	0.998	695	739	1.00
DISPLAY, state 1 to state 1	0.634	0.117	0.426	0.871	1228	716	1.00
DISPLAY, state 1 to state 2	0.128	0.097	0.000	0.308	546	191	1.01
DISPLAY, state 1 to state 3	0.238	0.117	0.011	0.444	933	534	1.00
DISPLAY, state 2 to state 1	0.463	0.231	0.007	0.853	950	560	1.00
DISPLAY, state 2 to state 2	0.155	0.151	0.000	0.477	1349	613	1.00
DISPLAY, state 2 to state 3	0.382	0.212	0.023	0.768	902	716	1.00
DISPLAY, state 3 to state 1	0.126	0.090	0.000	0.296	743	342	1.00
DISPLAY, state 3 to state 2	0.045	0.043	0.000	0.137	1118	429	1.00
DISPLAY, state 3 to state 3	0.829	0.096	0.647	0.989	867	726	1.00
ECONFO AND PRE-ARRIVAL EMAIL, state 1 to state 1	0.595	0.058	0.488	0.712	598	603	1.00
ECONFO AND PRE-ARRIVAL EMAIL, state 1 to state 2	0.059	0.028	0.010	0.116	1033	349	1.01
ECONFO AND PRE-ARRIVAL EMAIL, state 1 to state 3	0.347	0.057	0.233	0.456	512	624	1.01
ECONFO AND PRE-ARRIVAL EMAIL, state 2 to state 1	0.536	0.140	0.278	0.815	928	585	1.00
ECONFO AND PRE-ARRIVAL EMAIL, state 2 to state 2	0.074	0.061	0.000	0.200	1271	546	1.00
ECONFO AND PRE-ARRIVAL EMAIL, state 2 to state 3	0.389	0.139	0.112	0.652	886	875	1.00
ECONFO AND PRE-ARRIVAL EMAIL, state 3 to state 1	0.177	0.118	0.001	0.405	869	507	1.00
ECONFO AND PRE-ARRIVAL EMAIL, state 3 to state 2	0.057	0.051	0.000	0.157	805	501	1.01
ECONFO AND PRE-ARRIVAL EMAIL, state 3 to state 3	0.766	0.124	0.528	0.970	1088	675	1.00

Table W4: HMM Parameter Estimates, Part 3

Variables	Mean	SD	HDI 2.5%	HDI 97.5%	ess_bulk	ess_tail	R_hat
Transition Probability: $\rho_{css'}$							
EMERGING TECHNOLOGIES, state 1 to state 1	0.274	0.211	0.001	0.693	1456	741	1.00
EMERGING TECHNOLOGIES, state 1 to state 2	0.245	0.203	0.000	0.655	1739	654	1.00
EMERGING TECHNOLOGIES, state 1 to state 3	0.481	0.248	0.019	0.888	1422	695	1.00
EMERGING TECHNOLOGIES, state 2 to state 1	0.201	0.162	0.000	0.532	1604	617	1.00
EMERGING TECHNOLOGIES, state 2 to state 2	0.147	0.133	0.000	0.421	1226	398	1.01
EMERGING TECHNOLOGIES, state 2 to state 3	0.652	0.191	0.293	0.980	1390	667	1.00
EMERGING TECHNOLOGIES, state 3 to state 1	0.254	0.206	0.000	0.674	1167	566	1.00
EMERGING TECHNOLOGIES, state 3 to state 2	0.226	0.199	0.000	0.654	1602	667	1.00
EMERGING TECHNOLOGIES, state 3 to state 3	0.520	0.251	0.062	0.940	1187	650	1.00
NATURAL SEARCH, state 1 to state 1	0.731	0.041	0.657	0.817	285	597	1.02
NATURAL SEARCH, state 1 to state 2	0.005	0.004	0.000	0.014	640	483	1.00
NATURAL SEARCH, state 1 to state 3	0.264	0.041	0.178	0.338	283	612	1.02
NATURAL SEARCH, state 2 to state 1	0.710	0.105	0.487	0.898	760	508	1.00
NATURAL SEARCH, state 2 to state 2	0.016	0.016	0.000	0.049	1448	615	1.00
NATURAL SEARCH, state 2 to state 3	0.274	0.105	0.060	0.467	792	598	1.00
NATURAL SEARCH, state 3 to state 1	0.131	0.049	0.037	0.228	1119	427	1.01
NATURAL SEARCH, state 3 to state 2	0.005	0.005	0.000	0.014	827	405	1.00
NATURAL SEARCH, state 3 to state 3	0.864	0.049	0.769	0.959	1117	420	1.01
PAID SEARCH, state 1 to state 1	0.586	0.058	0.477	0.705	646	592	1.01
PAID SEARCH, state 1 to state 2	0.009	0.007	0.000	0.024	897	462	1.01
PAID SEARCH, state 1 to state 3	0.405	0.058	0.279	0.507	624	616	1.01
PAID SEARCH, state 2 to state 1	0.516	0.181	0.150	0.849	1533	793	1.00
PAID SEARCH, state 2 to state 2	0.060	0.059	0.000	0.174	1679	709	1.00
PAID SEARCH, state 2 to state 3	0.424	0.182	0.046	0.744	1351	818	1.01
PAID SEARCH, state 3 to state 1	0.327	0.079	0.180	0.486	1138	644	1.00
PAID SEARCH, state 3 to state 2	0.012	0.012	0.000	0.037	1207	502	1.01
PAID SEARCH, state 3 to state 3	0.660	0.078	0.504	0.805	1145	685	1.00
REFERRAL ENGINE, state 1 to state 1	0.531	0.138	0.265	0.790	895	552	1.00
REFERRAL ENGINE, state 1 to state 2	0.053	0.054	0.000	0.155	1325	556	1.00
REFERRAL ENGINE, state 1 to state 3	0.415	0.134	0.181	0.700	928	797	1.00
REFERRAL ENGINE, state 2 to state 1	0.617	0.204	0.235	0.976	1974	564	1.01
REFERRAL ENGINE, state 2 to state 2	0.146	0.132	0.000	0.411	1474	528	1.01
REFERRAL ENGINE, state 2 to state 3	0.237	0.180	0.001	0.591	1250	728	1.00
REFERRAL ENGINE, state 3 to state 1	0.376	0.191	0.056	0.761	1262	695	1.00
REFERRAL ENGINE, state 3 to state 2	0.079	0.076	0.000	0.233	1046	606	1.00
REFERRAL ENGINE, state 3 to state 3	0.545	0.198	0.150	0.882	1192	664	1.00
RESLINK, state 1 to state 1	0.696	0.151	0.409	0.970	1003	608	1.01
RESLINK, state 1 to state 2	0.115	0.106	0.000	0.330	1102	693	1.00
RESLINK, state 1 to state 3	0.190	0.131	0.001	0.428	991	544	1.00
RESLINK, state 2 to state 1	0.262	0.208	0.000	0.662	1298	828	1.00
RESLINK, state 2 to state 2	0.289	0.217	0.001	0.715	1672	729	1.00
RESLINK, state 2 to state 3	0.449	0.246	0.038	0.884	1428	895	1.01
RESLINK, state 3 to state 1	0.040	0.039	0.000	0.116	638	242	1.01
RESLINK, state 3 to state 2	0.052	0.048	0.000	0.149	861	404	1.00
RESLINK, state 3 to state 3	0.908	0.060	0.790	0.994	680	402	1.01

Table W5: HMM Parameter Estimates, Part 4

Variables	Mean	SD	HDI 2.5%	HDI 97.5%	ess_bulk	ess_tail	R_hat
Transition Probability: $\rho_{css'}$							
SOCIAL MEDIA, state 1 to state 1	0.352	0.199	0.001	0.699	799	689	1.01
SOCIAL MEDIA, state 1 to state 2	0.180	0.176	0.000	0.565	882	577	1.01
SOCIAL MEDIA, state 1 to state 3	0.468	0.195	0.118	0.876	1392	858	1.00
SOCIAL MEDIA, state 2 to state 1	0.486	0.241	0.041	0.888	862	561	1.00
SOCIAL MEDIA, state 2 to state 2	0.213	0.162	0.002	0.522	1280	646	1.00
SOCIAL MEDIA, state 2 to state 3	0.301	0.196	0.000	0.650	1246	712	1.00
SOCIAL MEDIA, state 3 to state 1	0.496	0.242	0.056	0.919	1282	590	1.00
SOCIAL MEDIA, state 3 to state 2	0.257	0.205	0.000	0.652	907	555	1.00
SOCIAL MEDIA, state 3 to state 3	0.247	0.200	0.000	0.636	837	412	1.00
UNPAID REFERRER, state 1 to state 1	0.864	0.043	0.785	0.944	210	453	1.02
UNPAID REFERRER, state 1 to state 2	0.020	0.008	0.003	0.036	879	589	1.00
UNPAID REFERRER, state 1 to state 3	0.116	0.043	0.035	0.198	199	394	1.03
UNPAID REFERRER, state 2 to state 1	0.858	0.074	0.727	0.994	538	743	1.01
UNPAID REFERRER, state 2 to state 2	0.016	0.016	0.000	0.049	1193	712	1.01
UNPAID REFERRER, state 2 to state 3	0.126	0.074	0.001	0.258	564	790	1.00
UNPAID REFERRER, state 3 to state 1	0.126	0.086	0.000	0.286	568	363	1.00
UNPAID REFERRER, state 3 to state 2	0.016	0.016	0.000	0.047	663	324	1.00
UNPAID REFERRER, state 3 to state 3	0.858	0.087	0.699	0.996	657	466	1.00
OUTSIDE CHANNEL, state 1 to state 1	0.963	0.014	0.938	0.992	205	384	1.02
OUTSIDE CHANNEL, state 1 to state 2	0.002	0.002	0.000	0.005	1102	770	1.01
OUTSIDE CHANNEL, state 1 to state 3	0.034	0.014	0.006	0.061	204	305	1.02
OUTSIDE CHANNEL, state 2 to state 1	0.000	0.000	0.000	0.001	951	511	1.00
OUTSIDE CHANNEL, state 2 to state 2	0.997	0.003	0.992	1.000	989	657	1.00
OUTSIDE CHANNEL, state 2 to state 3	0.003	0.003	0.000	0.008	952	639	1.00
OUTSIDE CHANNEL, state 3 to state 1	0.003	0.000	0.002	0.003	499	822	1.01
OUTSIDE CHANNEL, state 3 to state 2	0.005	0.000	0.004	0.005	926	793	1.00
OUTSIDE CHANNEL, state 3 to state 3	0.993	0.000	0.992	0.993	638	899	1.00

Table W6: Poisson Point Process Parameter Estimates, Part 1

Variables	Mean	SD	HDI 2.5%	HDI 97.5%	ess_bulk	ess_tail	R_hat
θ	2.814	0.031	2.758	2.876	1886	781	1.00
μ_0	0.002	0.000	0.002	0.003	628	783	1.00
$\boldsymbol{\alpha_{c'}}$							
AFFILIATE	-1.919	0.105	-2.120	-1.712	1916	951	1.02
B2B	-4.177	0.322	-4.794	-3.524	2073	726	1.00
DIRECT	0.850	0.041	0.765	0.925	624	912	1.00
DISPLAY	-1.821	0.100	-2.017	-1.632	1497	851	1.00
ECONFO AND PRE-ARRIVAL EMAIL	-0.737	0.064	-0.857	-0.613	1085	807	1.00
EMAIL	-0.762	0.062	-0.887	-0.649	921	777	1.00
EMERGING TECHNOLOGIES	-3.526	0.250	-4.020	-3.084	2309	649	1.00
NATURAL SEARCH	0.342	0.045	0.252	0.425	781	906	1.00
PAID SEARCH	-0.500	0.056	-0.615	-0.396	821	882	1.00
REFERRAL ENGINE	-2.338	0.133	-2.603	-2.087	1743	606	1.00
RESLINK	-2.294	0.122	-2.514	-2.047	1531	787	1.00
SOCIAL MEDIA	-3.329	0.207	-3.721	-2.893	2505	706	1.01
$\boldsymbol{\beta_c}$							
AFFILIATE	1.095	0.139	0.846	1.361	1104	478	1.01
B2B	0.189	0.569	-0.929	1.263	1159	716	1.00
DIRECT	-0.958	0.043	-1.034	-0.865	1620	815	1.00
DISPLAY	0.235	0.134	-0.040	0.472	1795	809	1.01
ECONFO AND PRE-ARRIVAL EMAIL	-0.094	0.078	-0.253	0.059	1633	732	1.00
EMAIL	-0.233	0.073	-0.364	-0.082	1338	964	1.00
EMERGING TECHNOLOGIES	-0.047	0.305	-0.700	0.520	1616	733	1.01
NATURAL SEARCH	-0.769	0.048	-0.863	-0.679	1506	840	1.00
PAID SEARCH	-0.072	0.065	-0.195	0.058	1450	781	1.00
REFERRAL ENGINE	0.018	0.185	-0.311	0.387	1647	758	1.01
RESLINK	-0.492	0.161	-0.805	-0.175	1689	877	1.00
SOCIAL MEDIA	1.896	0.680	0.631	3.258	1498	654	1.00
UNPAID REFERRER	-0.131	0.046	-0.218	-0.033	1270	952	1.00
$\boldsymbol{\rho}$							
AFFILIATE	0.021	0.087	-0.135	0.199	1049	775	1.01
B2B	0.162	0.438	-0.716	0.954	1228	637	1.00
DIRECT	0.190	0.018	0.156	0.227	1957	904	1.00
DISPLAY	-0.182	0.091	-0.349	-0.001	1589	721	1.01
ECONFO AND PRE-ARRIVAL EMAIL	-0.046	0.045	-0.141	0.040	1620	808	1.00
EMAIL	0.123	0.040	0.046	0.199	1395	808	1.00
EMERGING TECHNOLOGIES	-0.171	0.205	-0.554	0.246	1788	839	1.01
NATURAL SEARCH	0.159	0.024	0.110	0.203	1616	711	1.00
PAID SEARCH	-0.202	0.044	-0.289	-0.117	1714	994	1.00
REFERRAL ENGINE	0.250	0.125	-0.004	0.477	1354	913	1.00
RESLINK	0.520	0.062	0.394	0.636	1588	984	1.00
SOCIAL MEDIA	-1.633	0.604	-2.873	-0.546	1389	784	1.01
UNPAID REFERRER	0.248	0.022	0.207	0.292	1703	742	1.01

Table W7: Poisson Point Process Parameter Estimates, Part 2

Variables	Mean	SD	HDI 2.5%	HDI 97.5%	ess_bulk	ess_tail	R_hat
ϕ_0	-1.346	0.040	-1.427	-1.270	1681	834	1.01
$\boldsymbol{\phi_c}$							
AFFILIATE	0.131	0.144	-0.151	0.398	1917	638	1.01
B2B	-0.791	0.797	-2.381	0.647	2373	665	1.00
DIRECT	0.179	0.029	0.120	0.234	1733	730	1.00
DISPLAY	0.068	0.181	-0.262	0.432	2862	521	1.01
ECONFO AND PRE-ARRIVAL EMAIL	0.204	0.088	0.035	0.377	2035	609	1.01
EMAIL	0.036	0.077	-0.106	0.182	2360	732	1.01
EMERGING TECHNOLOGIES	-0.134	0.434	-0.936	0.668	2162	626	1.01
NATURAL SEARCH	0.055	0.042	-0.026	0.137	1785	725	1.00
PAID SEARCH	0.212	0.080	0.066	0.367	1633	751	1.00
REFERRAL ENGINE	0.324	0.226	-0.116	0.741	3000	910	1.00
RESLINK	-0.083	0.139	-0.375	0.170	2157	718	1.00
SOCIAL MEDIA	-0.724	0.667	-2.027	0.598	2941	723	1.00
UNPAID REFERRER	-0.323	0.044	-0.407	-0.236	2293	861	1.01

Web Appendix B: Additional Results on Customer Journey Prediction

Model-Free Evidence

Figure W5 shows the number of transactions over time in the data. The firm's website experiences four peaks of booking, with the first three occurring between mid-October to early November, and the final peak in December. Because the first three peaks fall within the calibration period, they are reflected in the booking probability prediction as external shocks in Figure 4(a) and 4(b), while the last peak that happens in the hold-out period are not captured.

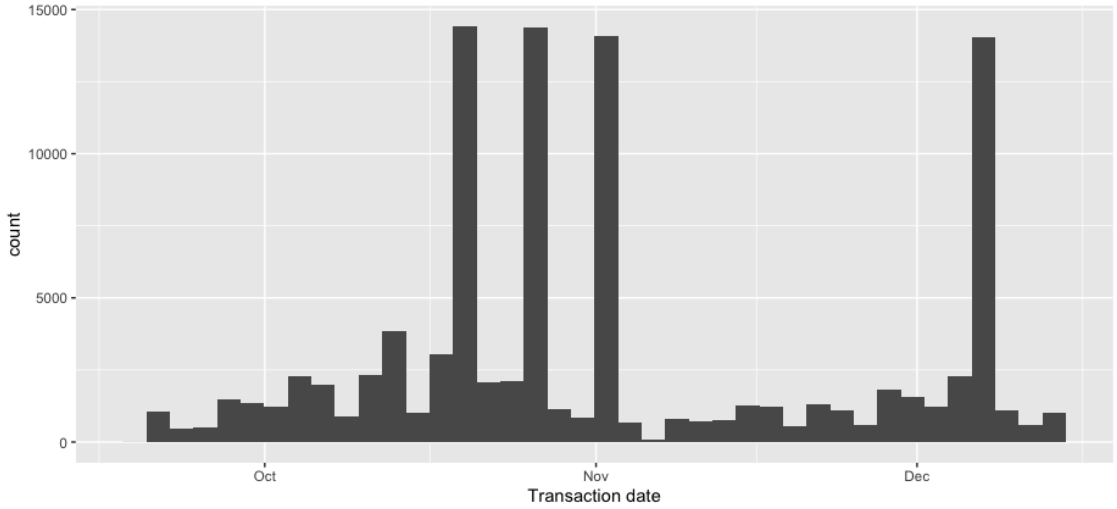


Figure W5: Number of Transactions over Time

Previous research on Customer Lifetime Value (CLV) has highlighted the prevalence of "clumpiness" in customer visit patterns. Following the approach of Zhang, Bradlow, and Small (2015), we measure the clumpiness of visits using the hotel dataset in our application section. Given that a substantial portion of the dataset comprises single-visit customers, we separately assess clumpiness for single-visit and multiple-visit customers. As shown in Table W8, 28% of all customers exhibit statistically significant clumpiness in their visits. Among multiple-visit customers, this proportion increases notably: 49% are identified as having clumpy visit patterns.

Table W8: Visit Clumpiness

	N	Nonclumpy (%)	Clumpy (%)
All Customers	92,575	72	28
Multiple-visit Customers	51,035	51	49
Single-visit Customers	41,540	98	2

In the dataset we simulate using the mixture DGP, all customers have multiple visits, and 5% are classified as visit-clumpy. While this proportion is lower than that observed in the hotel data application, we demonstrate in Web Appendix D – using a public dataset – that the degree of clumpiness can vary across different digital marketing contexts.

Balanced Accuracy, F1-Score and Precision-Recall Curve

We present the balanced accuracy of the proposed transformer model and benchmark models in Table W9 and W10. Because the dataset is very sparse, for binary classification tasks, the probability output for the positive class is very low, thus the 0.5 threshold is suboptimal (Wei and Dunbrack 2013; Buda, Maki, and Mazurowski 2018). We calculate the balanced accuracy for a wide range of different thresholds (Grandini, Bagli, and Visani 2020; Brodersen et al. 2010) and report the highest balanced accuracy across all thresholds, following the approach of Kim, Lee, and Jeon (2020) and Johnson and Khoshgoftaar (2019).

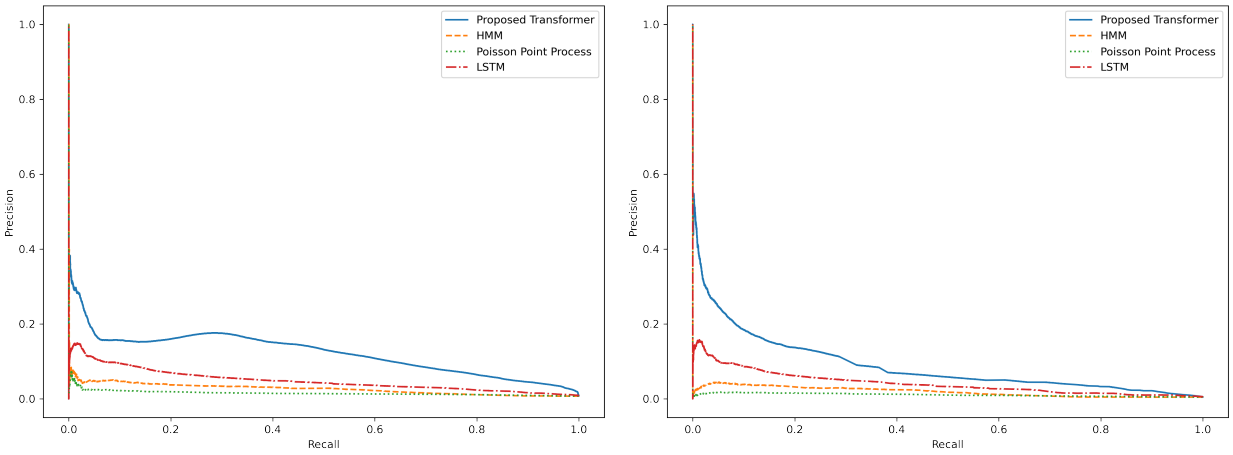
To evaluate the precision-recall tradeoff, we also plot the Precision-Recall Curve (Figure W6) for model comparison during the calibration period and report both the Area Under the Curve (PR-AUC) (Table W13, W14) and the best F1-score (Table W11, W12). Due to the class imbalance in the data, with a substantially larger negative class, both the F1-scores and PR-AUC values are relatively low, which is consistently reflected across all models. This is expected, as Precision-recall curve, and thus the F_β score, explicitly depends on the ratio of positive to negative test cases (Brabec et al. 2020). It is shown that class imbalance can significantly suppress both PR-AUC and F1-score values even when classifiers are well-calibrated (Jeni, Cohn, and De La Torre 2013; Davis and Goadrich 2006).

Table W9: Balanced Accuracy Comparison in the Calibration Period

Dependent Variable	In-Sample Balanced Accuracy				Out-of-Sample Balanced Accuracy			
	Proposed Transformer	LSTM	HMM	Point Process	Proposed Transformer	LSTM	HMM	Point Process
Booking	0.8729	0.7770	0.7050	0.6451	0.8610	0.7755	0.693	0.6481
Channel Visit								
AFFILIATE	0.9737	0.8208	0.7694	0.8232	0.8509	0.7910	0.7342	0.7543
B2B	0.9951	0.7329	0.7049	0.7998	0.8937	0.7497	0.733	0.8497
DIRECT	0.8517	0.7660	0.7372	0.7377	0.8231	0.7611	0.7262	0.7345
DISPLAY	0.9361	0.7631	0.6953	0.6733	0.8339	0.7295	0.7244	0.6878
ECONFO AND PRE-ARRIVAL EMAIL	0.9188	0.8021	0.7643	0.7336	0.8403	0.8026	0.7508	0.7116
EMAIL	0.9273	0.7613	0.7230	0.7148	0.8345	0.7602	0.6969	0.7131
EMERGING TECHNOLOGIES	0.9753	0.7503	0.7338	0.6070	0.8304	0.7357	0.7822	0.7271
NATURAL SEARCH	0.8748	0.7572	0.7266	0.7201	0.8277	0.7254	0.6974	0.6827
PAID SEARCH	0.9034	0.7313	0.6757	0.6764	0.8149	0.7079	0.6469	0.6509
REFERRAL ENGINE	0.9502	0.7359	0.7384	0.6926	0.8507	0.7303	0.7157	0.7072
RESLINK	0.9483	0.7792	0.6007	0.7225	0.8520	0.7339	0.6148	0.6684
SOCIAL MEDIA	0.9815	0.8622	0.8077	0.8050	0.8580	0.8322	0.8091	0.6427
UNPAID REFERRER	0.9116	0.8321	0.8235	0.8085	0.8433	0.8226	0.7973	0.7705

Table W10: Balanced Accuracy Comparison in the Hold-out Period ($t \geq 140$)

Dependent Variable	In-Sample Balanced Accuracy				Out-of-Sample Balanced Accuracy			
	Proposed Transformer	LSTM	HMM	Point Process	Proposed Transformer	LSTM	HMM	Point Process
Booking	0.8681	0.6274	0.5007	0.5105	0.8358	0.5993	0.5012	0.5163
Channel Visit								
AFFILIATE	0.7626	0.5762	0.7707	0.8975	0.7146	0.5709	0.5861	0.6875
B2B	0.6920	0.5127	0.5000	0.5000	0.6560	0.5155	-	-
DIRECT	0.8513	0.5827	0.5643	0.5713	0.7285	0.5764	0.5828	0.5576
DISPLAY	0.5631	0.5754	0.5416	0.5609	0.5394	0.5504	0.5812	0.5717
ECONFO AND PRE-ARRIVAL EMAIL	0.6800	0.5700	0.5213	0.6014	0.6857	0.5476	0.6067	0.5813
EMAIL	0.6211	0.5709	0.5999	0.5357	0.5924	0.5577	0.5714	0.5765
EMERGING TECHNOLOGIES	0.5062	0.6124	0.5000	0.5000	0.5111	0.5836	0.5	0.5
NATURAL SEARCH	0.8275	0.6273	0.5749	0.5543	0.7511	0.5787	0.5763	0.5633
PAID SEARCH	0.7872	0.6162	0.6174	0.6565	0.7768	0.5703	0.6275	0.6781
REFERRAL ENGINE	0.6091	0.6206	0.6213	0.5000	0.5674	0.5298	0.5717	0.6505
RESLINK	0.6363	0.5852	0.5691	0.5659	0.5441	0.5410	0.5	0.6661
SOCIAL MEDIA	0.5364	0.5743	0.5000	0.5000	0.5626	0.6743	0.5	0.5
UNPAID REFERRER	0.7500	0.6244	0.5873	0.6481	0.6803	0.6003	0.5954	0.6318

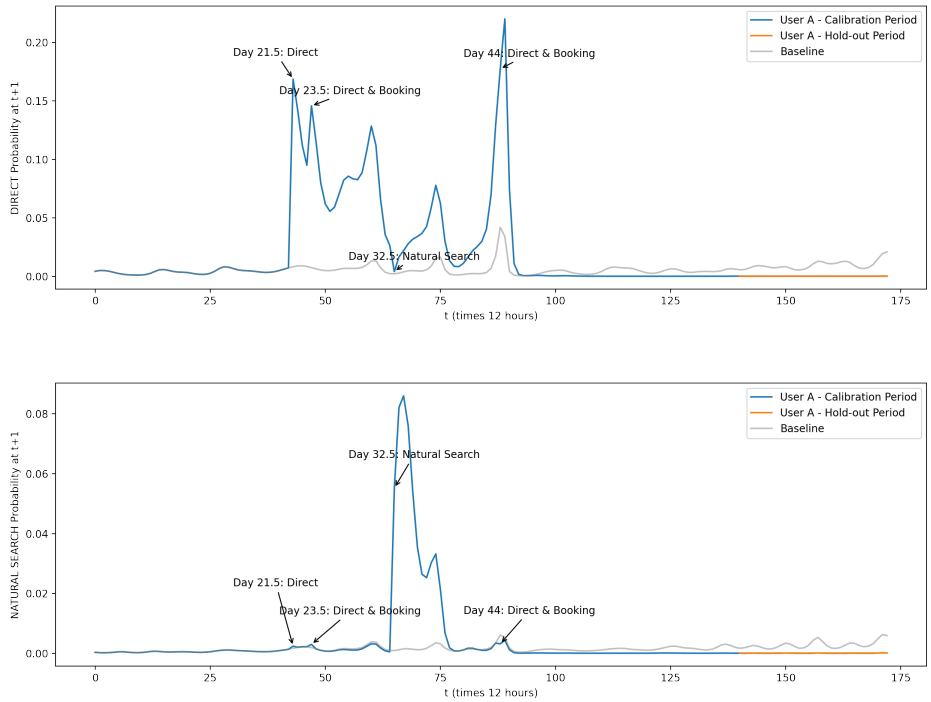


(a) In-Sample **(b)** Out-of-Sample

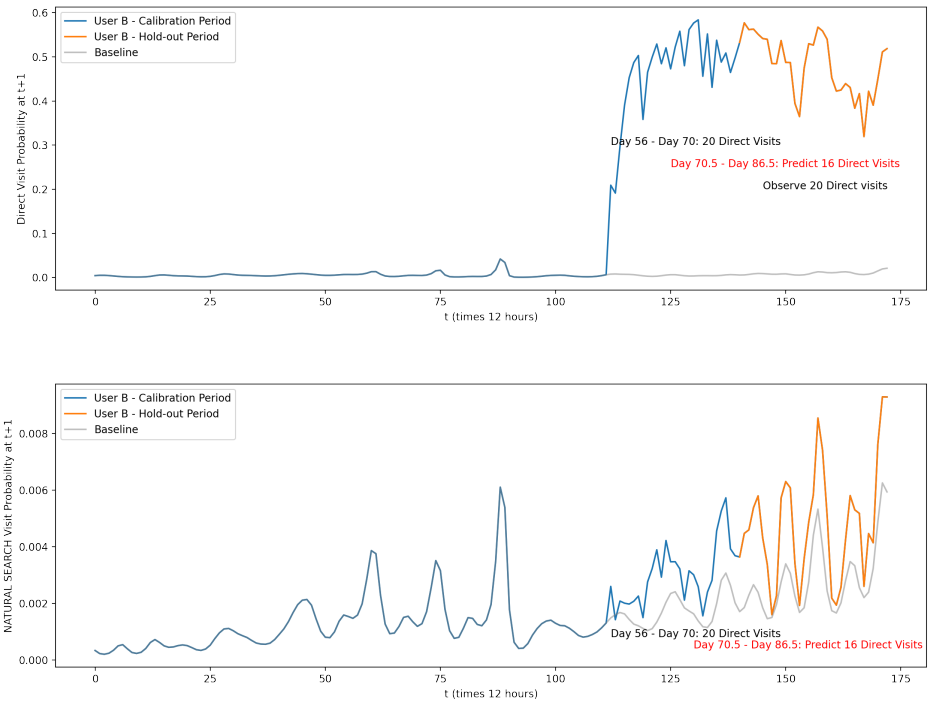
Figure W6: Precision-Recall curve of proposed model versus three benchmark models on the first 140 time periods.

Probability of Channel Visit over Time

In the *Model Training and Customer Journey Prediction* section in the main text, we showed how the predicted booking probability evolves over time using two user examples (Figure 4a, 4b). Likewise, the probability of channel visits over time can be visualized in a similar manner. Figure W7a and Figure W7b shows the evolvement of user A and user B's probability of visit through Direct and Natural Search channels in each period. User A uses both Direct and Natural Search channels while user B only uses Direct channel. Overall user A has a much lower probability of making direct visit to the website than that of user B, but a higher probability of visit through Natural Search than user B. This is inferred from the different visiting patterns of the two users. User A's probability of direct visit drops after she finished a booking at Day 23.5, and rises again after a visit was made through natural search at Day 32.5. User B has a higher probability of direct visit due to having a higher direct visit frequency than user A. Since user B never visits through Natural Search, the probability of visit through Natural Search is close to the baseline, which is lower than 0.01.



(a) User A - Direct and Natural Search Visit Probability



(b) User B - Direct and Natural Search Visit Probability

Figure W7: Predicted Direct and Natural Search Visit Probability of the Subsequent Period

Table W11: F1-Score Comparison in the Calibration Period ($0 \leq t < 140$)

Dependent Variable	In-Sample F1-Score				Out-of-Sample F1-Score			
	Proposed Transformer	LSTM	HMM	Point Process	Proposed Transformer	LSTM	HMM	Point Process
Booking	0.2293	0.0881	0.0666	0.0378	0.1705	0.0806	0.0591	0.0304
Channel Visit								
AFFILIATE	0.1770	0.1861	0.0111	0.1126	0.1819	0.1750	0.0235	0.1097
B2B	0.1624	0.0002	0.0003	0.1364	0.1488	0.0002	0.0011	0.119
DIRECT	0.2651	0.2576	0.2366	0.1007	0.2460	0.2453	0.2386	0.1065
DISPLAY	0.1498	0.1374	0.0064	0.0484	0.1234	0.1166	0.0131	0.0784
ECONFO AND PRE-ARRIVAL EMAIL	0.1877	0.1812	0.0257	0.0876	0.1702	0.1667	0.0169	0.0793
EMAIL	0.1747	0.1608	0.0343	0.0566	0.1612	0.1632	0.0234	0.0428
EMERGING TECHNOLOGIES	0.1388	0.0004	0.0006	0.1053	0.1027	0.0003	0.001	0.4286
NATURAL SEARCH	0.2144	0.2021	0.1216	0.0837	0.1898	0.1768	0.1075	0.061
PAID SEARCH	0.1418	0.1251	0.0294	0.0646	0.1183	0.1108	0.0231	0.0565
REFERRAL ENGINE	0.1130	0.0586	0.0056	0.0549	0.0867	0.0570	0.0044	0.12
RESLINK	0.1953	0.1638	0.0056	0.1169	0.1657	0.1415	0.0024	0.0315
SOCIAL MEDIA	0.2097	0.1770	0.0030	0.1600	0.1676	0.1626	0.0027	0.1239
UNPAID REFERRER	0.3074	0.3134	0.3314	0.1530	0.2849	0.2936	0.2643	0.1073

Time-Varying Importance of Touchpoints

Recently, a number of attribution methods were proposed to increase the interpretability of deep learning models (Lundberg and Lee 2017; Shrikumar, Greenside, and Kundaje 2017; Sundararajan, Taly, and Yan 2017). In this research, we use the Integrated Gradients (IG) method proposed by Sundararajan, Taly, and Yan (2017) to calculate the time-varying importance score for each customer interaction event in the journey. It has been used in a wide range of disciplines beyond computer science (Senior et al. 2020; Davies et al. 2021; Novakovsky et al. 2022). Nevertheless, we have reviewed related papers to better understand how IG compares to Shapley in contexts similar to ours (e.g., Sundararajan and Najmi 2020; Feng et al. 2022). While Sundararajan and Najmi (2020) note that Shapley values encompass several methods, including Integrated Gradients, Feng et al. (2022) compare Baseline Shapley (BShap) values to Integrated Gradients using simulations. They examined common model classes where BShap and IG produce identical explanations and where they differ. Their simulations show that the differences are not significantly large unless tree-based algorithms (which are not differentiable) are involved. Consequently, the authors conclude

Table W12: F1-Score Comparison in the Hold-out Period ($t \geq 140$)

Dependent Variable	In-Sample F1-Score				Out-of-Sample F1-Score			
	Proposed Transformer	LSTM	HMM	Point Process	Proposed Transformer	LSTM	HMM	Point Process
Booking	0.0731	0.0253	0.0116	0.0334	0.0631	0.019	0.0117	0.0193
Channel Visit								
AFFILIATE	0.1546	0.1071	0.2667	0.1538	0.0958	0.1042	0.0057	0.0158
B2B	0.0251	0.0001	0.0002	0.0448	0.0381	0.0001	-	-
DIRECT	0.1445	0.0522	0.0732	0.0331	0.1368	0.0438	0.0783	0.0393
DISPLAY	0.0467	0.0050	0.0006	0.0190	0.0370	0.0046	0.053	0.0476
ECONFO AND PRE-ARRIVAL EMAIL	0.0446	0.0799	0.0036	0.0214	0.0339	0.0573	0.0174	0.0259
EMAIL	0.0539	0.0621	0.0316	0.0588	0.0486	0.0533	0.0232	0.0136
EMERGING TECHNOLOGIES	0.0035	0.0153	0.0121	0.0001	0.0004	0.0041	0.0121	0.0002
NATURAL SEARCH	0.1207	0.0528	0.0275	0.0213	0.0722	0.0402	0.0209	0.0252
PAID SEARCH	0.0925	0.0622	0.0523	0.0112	0.0524	0.0372	0.0574	0.0226
REFERRAL ENGINE	0.0647	0.0008	0.0007	0.0005	0.0275	0.0034	0.0087	0.07
RESLINK	0.0682	0.0076	0.0135	0.0077	0.0108	0.0034	0.0046	0.0111
SOCIAL MEDIA	0.0960	0.0008	0.0001	0.0001	0.0311	0.0041	0.0046	0.0002
UNPAID REFERRER	0.1508	0.0609	0.0962	0.0317	0.1333	0.0579	0.125	0.1023

that the choice between the two methods is largely based on convenience and task suitability. Since we do not use tree-based algorithms, we believe our results are not significantly impacted by this choice.

Here we briefly describe the algorithm. To begin with, a baseline is defined to compare the importance of each input variable to the baseline. In our application the baseline defined as a period with no activity where the input is $X_s = 0$ for all channel s. Denote the baseline embedding by $\mathbf{X}^0$. Let F denote the neural network that takes input $\mathbf{X}_t$ for each t and output a probability in $[0, 1]$. Consider the straightline between the baseline $\mathbf{X}^0$ and the input $\mathbf{X}_t$, the integrated gradients are calculated by cumulating the gradients at all points along the path. According to Sundararajan, Taly, and Yan (2017), the integrated gradient along the s^{th} dimension for the input $\mathbf{X}_t$ and baseline $\mathbf{X}^0$ is defined as the path integral

$$IntegratedGrads_s(\mathbf{X}_t) ::= (X_{st} - X_s^0) \times \int_{\alpha=0}^{1} \frac{\partial F(X_s^0 + \alpha \times (X_{st} - X_s^0)}{\partial X_{st}} d\alpha \tag{W8}$$

To get the overall importance of $\mathbf{X}_t$, one can sum up $IntegratedGrads_s(\mathbf{X}_t)$ across all dimension s. The Sundararajan, Taly, and Yan (2017) paper gives the result that the overall

Table W13: Precision-Recall AUC Comparison in the Calibration Period ($0 \leq t < 140$)

Dependent Variable	In-Sample PR AUC				Out-of-Sample PR AUC			
	Proposed Transformer	LSTM	HMM	Point Process	Proposed Transformer	LSTM	HMM	Point Process
Booking	0.1401	0.0402	0.027	0.0153	0.0948	0.0334	0.019	0.0108
Channel Visit								
AFFILIATE	0.0916	0.0849	0.0029	0.0444	0.0771	0.0826	0.0036	0.0193
B2B	0.0775	0.0001	0.0001	0.0322	0.0533	0.0001	0.0002	0.0358
DIRECT	0.1824	0.1581	0.1277	0.0477	0.1534	0.1411	0.1302	0.0435
DISPLAY	0.0680	0.0520	0.0014	0.0095	0.0396	0.0376	0.0027	0.0162
ECONFO AND PRE-ARRIVAL EMAIL	0.0974	0.0855	0.0073	0.0272	0.0733	0.0736	0.0041	0.0199
EMAIL	0.0933	0.0729	0.0082	0.0138	0.0704	0.0715	0.005	0.0113
EMERGING TECHNOLOGIES	0.0578	0.0001	0.0002	0.0110	0.0262	0.0001	0.0002	0.3732
NATURAL SEARCH	0.1345	0.1085	0.0415	0.0365	0.1016	0.0835	0.0297	0.0175
PAID SEARCH	0.0709	0.0500	0.0071	0.0325	0.0456	0.0386	0.0041	0.0100
REFERRAL ENGINE	0.0456	0.0096	0.0011	0.0090	0.0229	0.0086	0.0007	0.0138
RESLINK	0.1006	0.0694	0.0006	0.0199	0.0617	0.0501	0.0006	0.0057
SOCIAL MEDIA	0.0977	0.0569	0.0007	0.1221	0.0618	0.0432	0.0006	0.0645
UNPAID REFERRER	0.2088	0.2012	0.2020	0.0649	0.1747	0.1749	0.1256	0.0368

importance of $\mathbf{X}_t$ is $F(\mathbf{X}_t) - F(\mathbf{X}^0)$ under the condition that F is differentiable almost everywhere.

Following Equation W8, $IntegratedGrads_s(\mathbf{X}_t)$ provides the importance score for each channel s on conversion prediction of the targeting time period. Let $a_{n\tau st}$ denote the importance score of touchpoint s that happens at period t for the prediction of customer n's conversion probability at period τ. A greater positive importance score $a_{n\tau st}$ indicates interacting with the touchpoint s is associated with higher probability of conversion for n at time τ. A negative importance score indicates s is associated with lower probability of conversion. The aggregate importance score is calculated as $A_s = \sum_{n,\tau,t} a_{n\tau st}$.

For the time-varying effect of a touchpoint, let $\delta = t - \tau$ denote the time difference between the interaction with the touchpoint and the purchase. The mean importance for a touchpoint s at δ period of time before purchase is $\mu_c(\delta) = \sum_{n,t,\tau=t-\delta} a_{n\tau st} / \sum_{n,\tau=t-\delta} \mathbb{1}_{n\tau s}$ ($\sum_{n,\tau=t-\delta} \mathbb{1}_{n\tau s} > 0$), where $\mathbb{1}_{n\tau s} = \{0, 1\}$ is the indicator of whether customer n visits through s at time τ. $\mu_s(\delta)$ denotes the average impact of a visit through touchpoint c on the future conversion probability after a period of time δ. In the case when $\sum_{n,\tau=t-\delta} \mathbb{1}_{n\tau s} = 0$,

Table W14: Precision-Recall AUC Comparison in the Calibration Period ($\mathbf{t \geq 140}$)

Dependent Variable	In-Sample PR AUC				Out-of-Sample PR AUC			
	Proposed Transformer	LSTM	HMM	Point Process	Proposed Transformer	LSTM	HMM	Point Process
Booking	0.0347	0.0104	0.0044	0.0067	0.0278	0.0082	0.0043	0.0085
Channel Visit								
AFFILIATE	0.0698	0.0373	0.0721	0.0404	0.0294	0.0368	0.0005	0.0024
B2B	0.0037	0.0000	0.0000	0.0088	0.0046	0.0000	-	-
DIRECT	0.0898	0.0247	0.0283	0.0196	0.0683	0.0196	0.0275	0.0147
DISPLAY	0.0042	0.0005	0.0002	0.0008	0.0027	0.0005	0.0035	0.0161
ECONFO AND PRE-ARRIVAL EMAIL	0.0128	0.0161	0.0010	0.0031	0.0108	0.0110	0.0019	0.0028
EMAIL	0.0146	0.0111	0.0025	0.0074	0.0098	0.0079	0.0023	0.0024
EMERGING TECHNOLOGIES	0.0001	0.0004	0.0016	0.0000	0.0001	0.0002	0.0024	0.000
NATURAL SEARCH	0.0549	0.0190	0.0085	0.0072	0.0315	0.0147	0.0082	0.0073
PAID SEARCH	0.0213	0.0103	0.0042	0.0024	0.0100	0.0081	0.0063	0.0218
REFERRAL ENGINE	0.0153	0.0002	0.0002	0.0002	0.0033	0.0003	0.0008	0.0085
RESLINK	0.0131	0.0006	0.0007	0.0007	0.0010	0.0005	0.0007	0.0015
SOCIAL MEDIA	0.0295	0.0001	0.0000	0.0000	0.0020	0.0004	0.0014	0.0000
UNPAID REFERRER	0.0709	0.0207	0.0215	0.0087	0.0492	0.0209	0.0433	0.0370

$\mu_s(\delta) = 0$.

Figure W8 compares the aggregate importance of each channel or variable. The aggregate importance score is the total increases in conversion when the channel visits or variable indicator equals one as compared to the counterfactual baseline where the channel or variable indicator equals zero. Direct visits to the website and previous bookings have the most positive impact on conversion prediction, both signaling a strong likelihood of purchase. This is not surprising given that the sample consists of loyalty-program members who diretcy visit the website to make their bookings. The results also show that the conversion probability would be lower if the previous booking was a weekend stay, as such bookings may indicate leisure travel and thus a lowering effect. Customer-initiated channels such as natural search and unpaid referrer have higher impact on conversion than firm-initiated channels such as paid search and email. Among firm-initiated channels, paid search has the highest importance, followed by email and affiliate.

In Figures 7 and 8 of the paper (main text), we present the time-varying impact of direct and email visits. Here, we extend the analysis to showcase the impact of other channels.

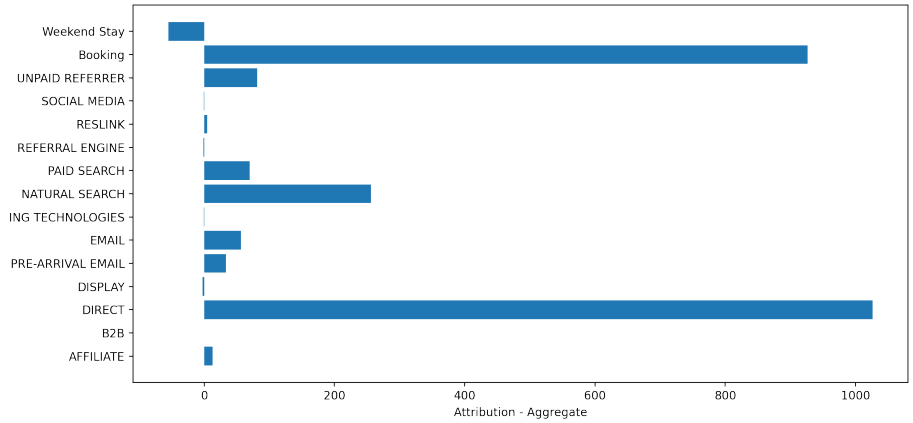


Figure W8: Aggregate importance of input variables on booking conversion

Figures W10 through W12 display mean importance score for three channels and previous purchase. Notably, natural search has a more significant effect on conversion probability compared to paid search, suggesting higher purchase intent among natural search users. Remarkably, most touchpoints exhibit positive impact on conversion prediction up to a certain time threshold, typically around 30 days before purchase. Visits occurring earlier than this threshold tend to have slightly negative impact, indicating lower purchase likelihood on the website.

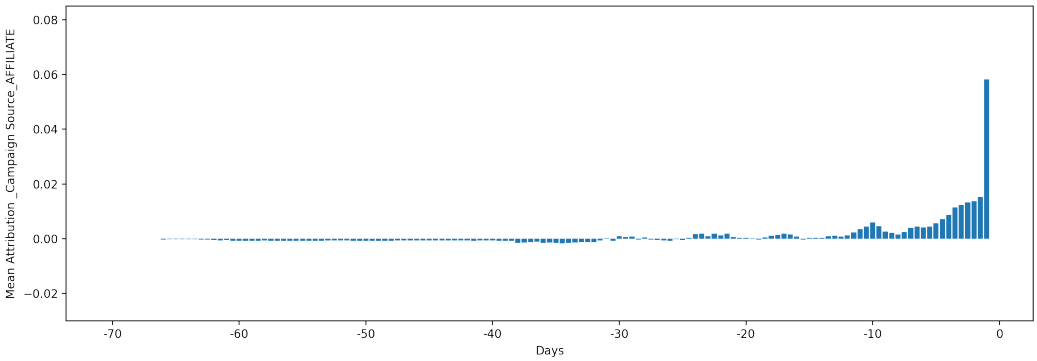


Figure W9: Attribution of Affiliate channel

Figure W20 illustrates the impact of prior booking history on predicted purchase probability. Notably, the impact of previous bookings on conversion differs significantly from

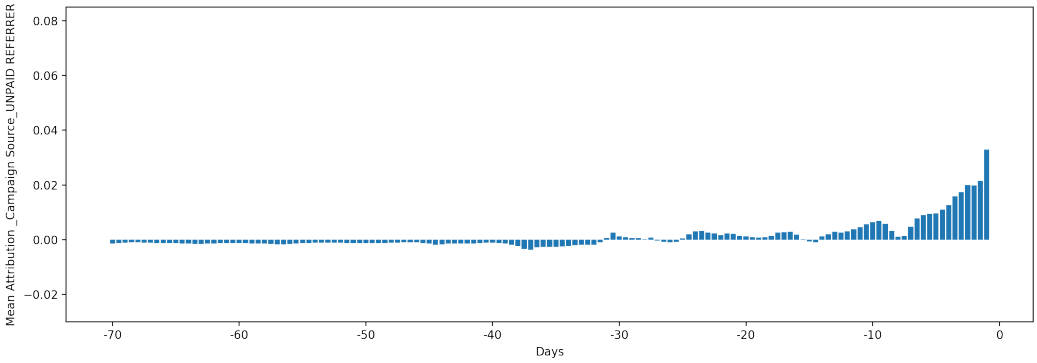


Figure W10: Attribution of Unpaid Referrer channel

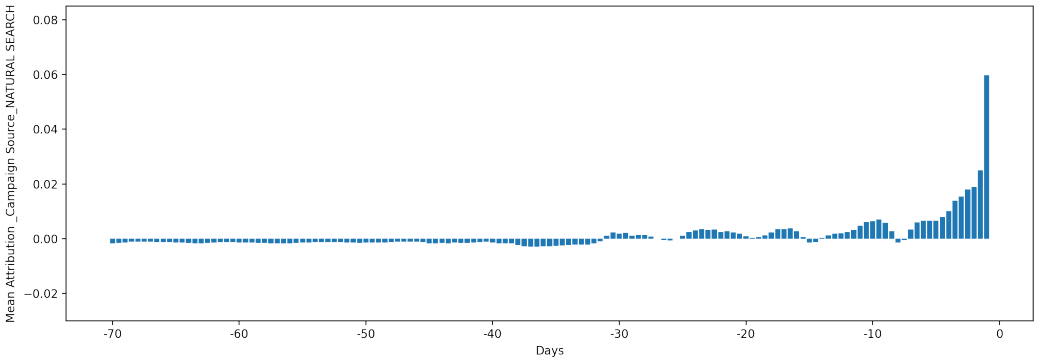


Figure W11: Attribution of Natural Search channel

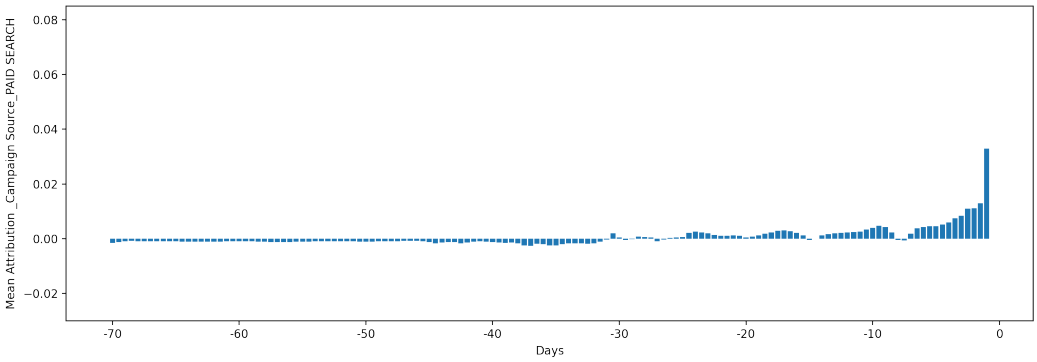


Figure W12: Attribution of Paid Search channel

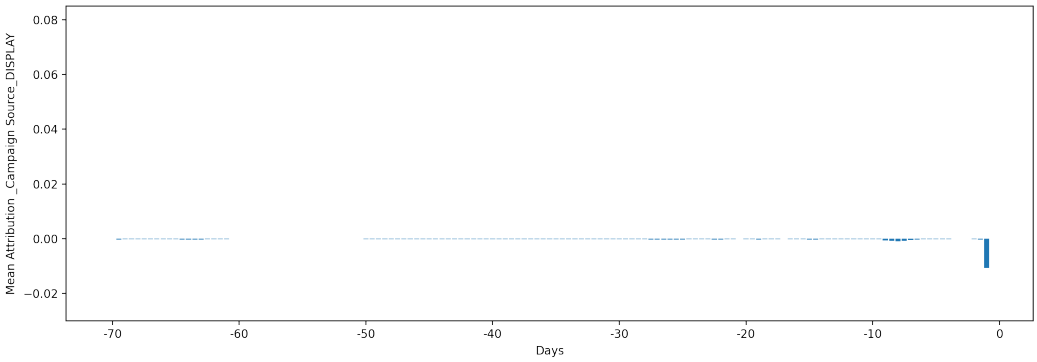


Figure W13: Attribution of Display channel

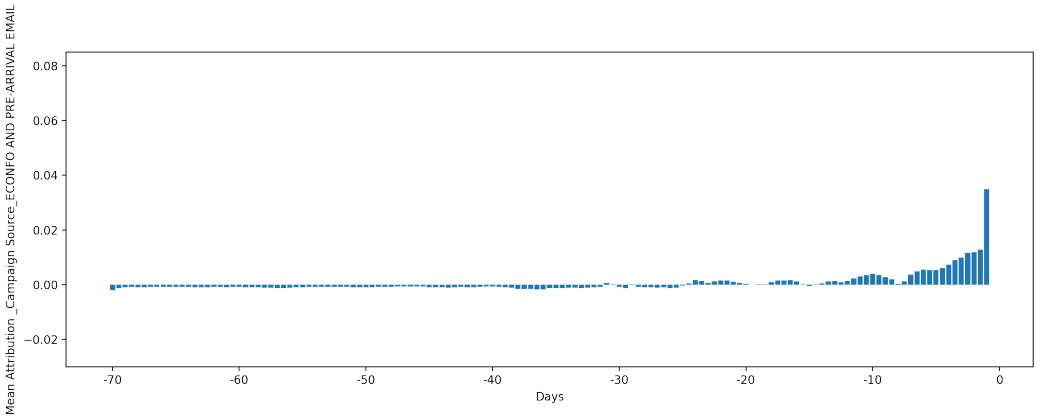


Figure W14: Attribution of Pre-Arrival Email channel

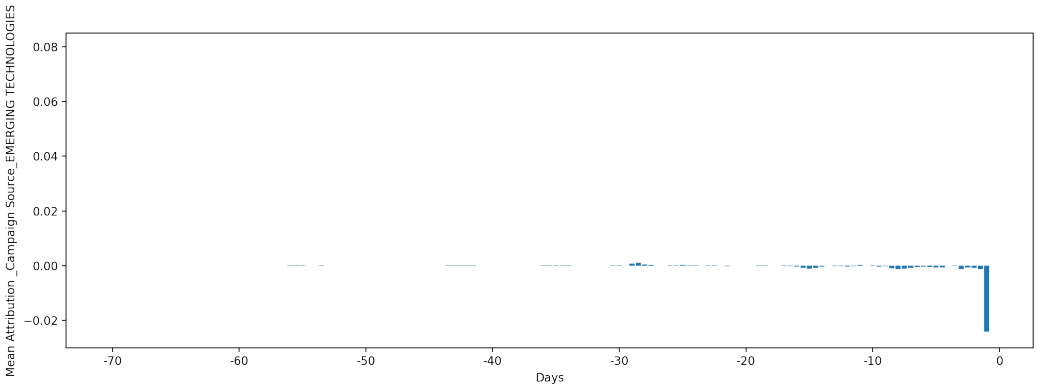


Figure W15: Attribution of Emerging Tech channel

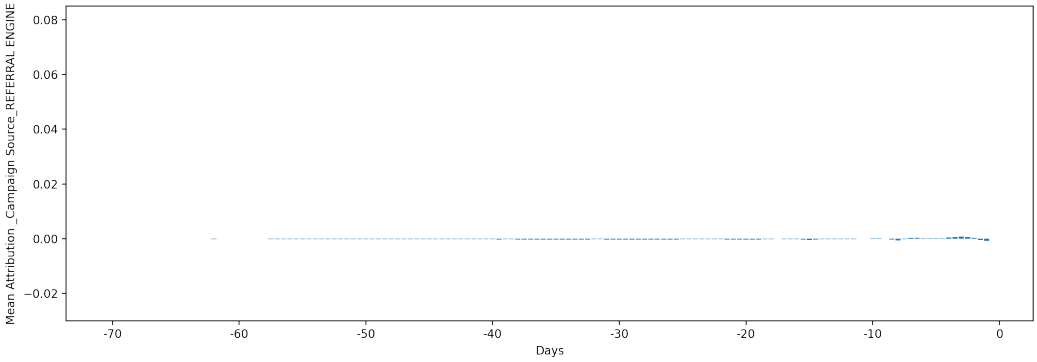


Figure W16: Attribution of Referral Engine channel

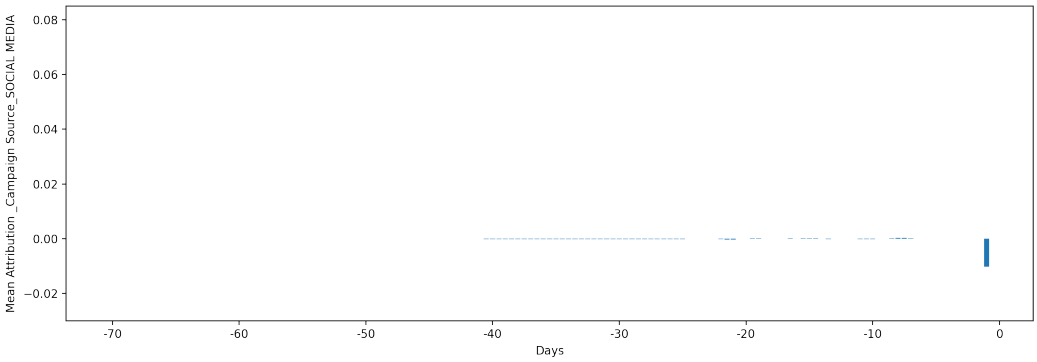


Figure W17: Attribution of Social Media channel

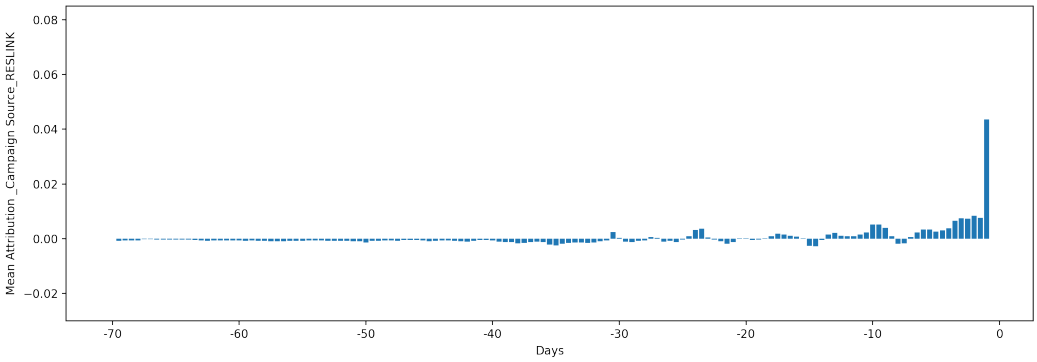


Figure W18: Attribution of Reservation Link channel

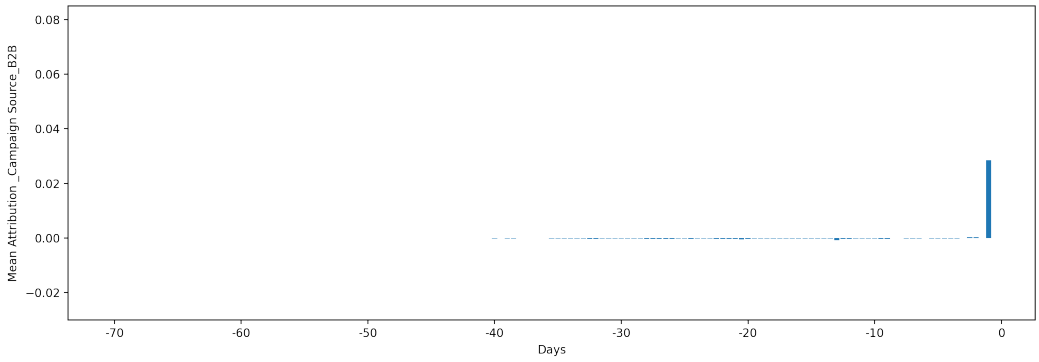


Figure W19: Attribution of B2B channel

other touchpoints. Generally, having a booking history with the firm boosts a customer's purchase likelihood, particularly within approximately 35 days, except when a booking occurs within 12 hours, leading to a sharp decline in predicted booking probability. This finding underscores the likelihood of historical purchases enhancing customer loyalty, given their membership in a loyalty program, but also indicates that consecutive purchases within a short timeframe are unlikely. Compared to other touchpoints, the influence of booking history exhibits a slower decay within the 30-day window, suggesting a more sustained loyalty effect.

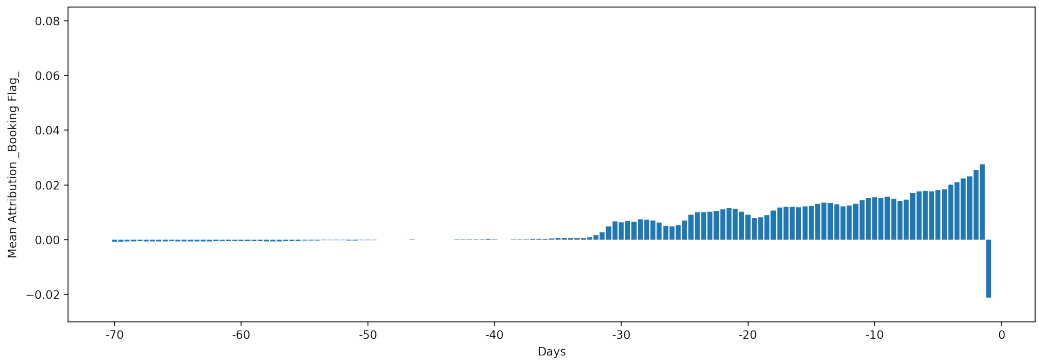


Figure W20: Attribution of previous booking

Web Appendix C: Advertising Targeting

In this Appendix, we examine how our transformer model can identify marketing actions to improve return-on-investment. A formal optimal determination would require detailed data on marketing actions, allowing joint modeling of both supply-side and demand-side effects (e.g., Manchanda, Rossi, and Chintagunta, 2004) or their integration through the NEIO modeling framework. Since such data is unavailable, we conduct analyses based on conservative assumptions of marketing effectiveness. These analyses illustrate the potential utility of our model, provided suitable data become available.

Our dataset reveals substantial variation in firm-initiated touchpoints compared to customer-initiated touchpoints. Specifically, demand for room nights is derived primarily from customer decisions, such as vacation planning or attending conferences and events, indicating most interactions are customer-driven. Firm-initiated actions primarily serve to guide customers toward the firm's website following initial customer engagement. Analysis shows that firm-initiated touchpoints occur rather randomly relative to customer-driven activities. Given this variability, our model is particularly relevant for typical operational scenarios ("business as usual"). If standard firm-initiated marketing efforts continue, our model can effectively suggest beneficial actions and accurately estimate their impact on customer conversion rates. Additionally, the email campaigns analyzed were general (with conservative assumptions of marketing effectiveness) and aggregate-focused rather than personalized or retargeted, meaning emails reached customers at varied stages within their customer journey.

Email targeting counterfactual.

Among the thirteen available channels, firm-initiated channels include Paid Search, Email, Pre-arrival Email, Affiliate, Display, and Social Media. These channels are designed to target customers and influence their journey. Our model estimates the impact of a firm's advertising efforts on user visitation and purchasing behavior. Using Email advertising as an example,

we demonstrate the effectiveness of ad retargeting across different strategies as predicted by the transformer approach.

We simulate an Email advertising retargeting campaign that would begin after the end of the calibration period ($t = 139$) retaining all regular campaigns that already exist in the data. Then we predict user visit and purchase probabilities for the following 15 days or 30 12-hour periods (Figure W21) and compare these predicted outcomes against the baseline probability with no interventions. Using the hold-out sample of 9,258 users, we explore the predictions of different targeting strategies: (a) indiscriminate targeting across all 9,258 users in the sample; (b) targeting users who are estimated to have the highest baseline purchase probability at $t = 139$; (c) behavioral targeting using a simple heuristic, such as total visits in the last 10 days of the calibration period, selecting users with the most visits, similar to the common practice in behavioral targeting based on user clicks and browsing history and (d) targeting users with the highest potential increase in purchase probability in the next 30 periods.

For selective targeting b), c) and d), suppose the firm selects 2,000 users from the hold-out sample. To measure the effectiveness of the advertising campaign, we use the average conversion increase per targeted individual in the observation period. Let p_{nt} denote the purchase probability of customer n at period t with the campaign and p^0_{nt} denote the purchase probability of customer n at period t without the campaign, the average conversion increase is given by $\frac{1}{N}\sum_{n=1}^{N}\sum_{t=140}^{169}(p_{nt} - p^0_{nt})$. A higher average conversion increase per targeted customer, under a fixed marginal cost of targeting an individual, indicates a more effective ad campaign.

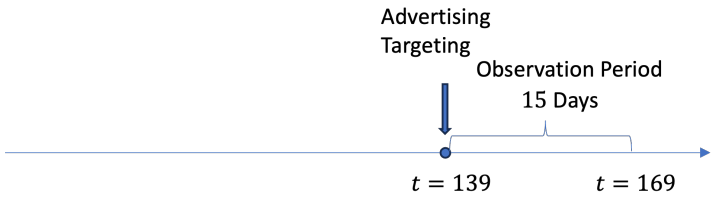


Figure W21: Illustration of Firm Advertising Targeting

We apply a 5% click-through rate to all targeted users, based on industry benchmarks (CampaignMonitor 2022). Table W15 compares the outcomes of various targeting strategies. (Note that this is a conservative assumption - those users in the predicted groups are likely to have a higher click-through rate). Indiscriminate targeting is the least effective, yielding an average conversion increase of just 0.0011 per targeted user, or 1.1 conversions per 1,000 targeted users. Behavioral targeting, based on visit history, performs slightly better, with 1.2 conversions per thousand users. In contrast, the two model-based strategies deliver significantly higher results. Targeting users with the highest baseline purchase probability leads to 2.1 conversions per thousand targeted users. The most effective approach, however, is targeting users with the highest potential increase in purchase probability, resulting in a striking 4.5 conversions per thousand targeted users. For example, targeting the 2,000 users with the highest predicted purchase probability increase achieves nearly the same total conversion gain (9 conversions) as indiscriminate targeting of all 9,258 users (10 conversions). This smaller scale delivers five times the ROI, showcasing the precision and efficiency of our model. However, this strategy requires simulating outcomes for each user, making it computationally intensive. Despite this trade-off, it is highly effective for driving high-value conversions.

Table W15: Email Advertising Targeting

Targeted Population	Size	Avg. Conversion Increase		Total Conversion Gain
		Per Targeted User	Per Click-Through	
Everyone	9,258	0.0011	0.0216	10.0
Users with most visits in the past 10 days	2,000	0.0012	0.0237	2.4
Users with highest baseline purchase probability	2,000	0.0021	0.0429	4.3
Users with highest increase in purchase probability	2,000	0.0045	0.0900	9.0

* Under 5% uniform click-through rate.
** Results based on the predictions of the subsequent 15 days.

If we do not know the ground truth in real-world data, as in the above case, what can we do? To address this potential limitation, we further investigate these findings using a simulation exercise. We consider three channels A, B, and Email and use AR3 as the data generating process (DGP) to simulate the visit and purchase data for 20 periods for 2,000

customers with an overall average conversion rate of 18%. An e-mail targeting campaign is undertaken at Period 80 and the conversion lift is determined as the difference between the purchase probabilities with and without the e-mail targeting in the next 20 periods. Table W16 provides the total conversion lift (and average conversion lift per targeted user) achieved under each of the competing models compared against the same metrics for the DGP prediction (calculated using he same DGP model that generates the data) for different four scenarios: targeting (a) all the 2000 users, (b) top 10% customers with highest baseline purchase probability, (c) the top 10% customers with most visits in the past 20 periods, and (d) top 10% customers with highest increase in conversion probabilities. Table W16 shows that the Transformer model is closest among all the competing models to the DGP prediction in all the four cases, with LSTM coming second, and HMM and Point Process models a distant third and fourth. This simulation illustrates that the transformer model performs the best among all competing models at identifying the best intervention policy.

Targeting timing.

A user's customer journey consists of both customer-initiated and firm-initiated touchpoints. This raises a critical question: when is the optimal time for a firm to target a customer based on an observed customer-initiated touchpoint that might signal a potential sale? Targeting too early, before the customer is ready, or too late, after the purchase decision has already been made, can lead to ineffective ad targeting. The optimal timing of targeting has not been extensively explored in the existing literature, partly due to the sparsity of customer visit or transaction data over time. However, with the proposed transformer model, we can now dynamically tailor targeting strategies for each individual, leveraging their observed history of visits and purchases to optimize touchpoint timing and maximize overall impact.

We conduct an individual-level analysis of email targeting following a direct visit by a customer, focusing on the customer journeys of two users: User A and User B. Specifically,

Table W16: Results Summary of Simulated Targeting

Targeted Population Selected by Each Model	Size	Model Estimate		DGP Prediction	
		Total Conversion Lift†	Avg. Conversion Lift Per Targeted	Total Conversion Lift	Avg. Conversion Lift Per Targeted
Everyone					
Transformer	2,000	5.1871	0.0026	6.7405	0.0034
HMM	2,000	-3.2195	-0.0016	6.7405	0.0034
Poisson Point Process	2,000	0.4896	0.0002	6.7405	0.0034
LSTM	2,000	4.3937	0.0022	6.7405	0.0034
Top 10% customers with highest baseline purchase probability					
Transformer	200	0.0477	0.0002	0.6766	0.0034
HMM	200	-0.1946	-0.0010	0.3605	0.0018
Poisson Point Process	200	-0.0490	-0.0002	0.0952	0.0005
LSTM	200	0.0532	0.0003	0.6949	0.0035
Top 10% customers with most visits in past 20 periods					
Transformer	200	0.3885	0.0019	0.4034	0.0020
HMM	200	-0.3277	-0.0016	0.4034	0.0020
Poisson Point Process	200	0.0207	0.0001	0.4034	0.0020
LSTM	200	0.3205	0.0016	0.4034	0.0020
Top 10% customers with highest increase in conversion probability					
Transformer	200	1.4358	0.0072	1.5733	0.0079
HMM	200	0.1395	0.0007	0.9041	0.0045
Poisson Point Process	200	0.2557	0.0013	1.1777	0.0059
LSTM	200	1.1735	0.0059	1.3093	0.0065

Note. a) We use AR3 as the DGP to simulate the visit and purchase data. A targeting campaign is simulated at period 80 using the Email Channel. b) Results are based on targeting the top 10% individuals with the highest predicted conversion lift in the next 20 periods after targeting. Conversion lift is given by the difference between the purchase probability with and without the targeting, under the assumption of 5% click-through rate. c) The table shows the total conversion lift from the top 10% individuals and the average conversion lift per individual. Model estimate is the predicted conversion lift given by the models in the left column. DGP prediction is the predicted conversion lift calculated using the same DGP model that generates the data.

† Note that the negative lifts means that the values for these cases are below the baseline values and could be attributed to the fact that we use AR3 which typically favor Transformers and LSTM, but our simulation studies show that, in general, HMM and Point Process models underperform in most cases.

we examine how email targeting 12 hours after a direct visit compares to targeting 5 days after the visit in influencing conversion outcomes. For both users, we compare the baseline probability of conversion without email targeting (represented by the blue line in Figures X and Y) to the probabilities of conversion with email targeting at two different times: (a) within 12 hours of the direct visit (orange dotted line) and (b) within 5 days of the direct visit (green dotted line). These conversion lifts are accumulated over 15-day period following the email targeting. To ensure robust estimates, we calculate average conversion probabilities using 150 iterations of the transformer-based prediction algorithm. For each period, we test whether the conversion probabilities with and without email targeting are significantly different, retaining only the significant incremental lifts for further analysis.

For User A (Figure W22), who has had only one direct visit in their history on Day 21.5, email targeting within the next 12 hours of the direct visit results in a next-period instantaneous conversion lift of 0.011 and a 15-day total conversion lift of 0.135. In contrast, if the email targeting occurs 5 days after the direct visit, the next-period instantaneous conversion lift is 0.013, but the 15-day total conversion lift decreases to 0.110. Thus, for User A, targeting within 12 hours of the direct visit is clearly the superior strategy.

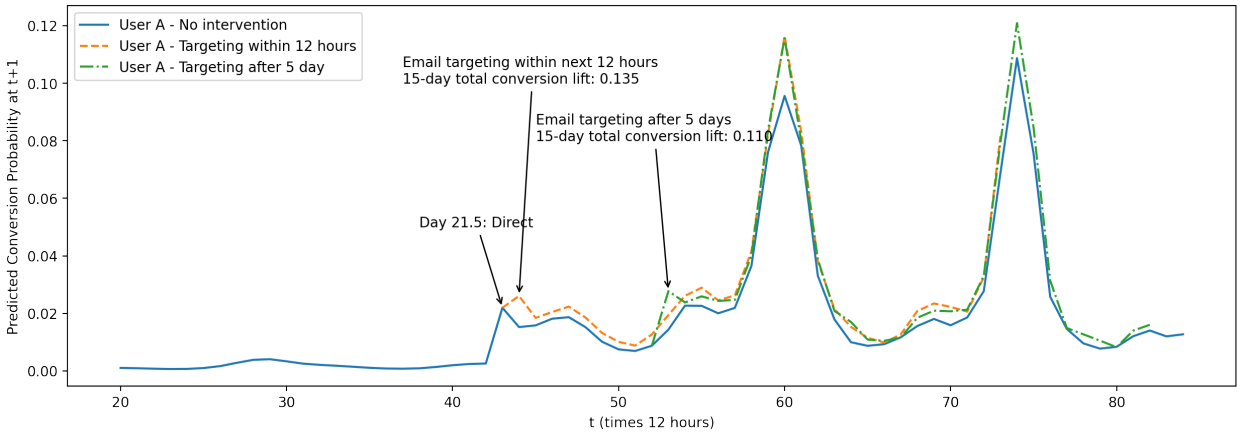


Figure W22: Email Targeting of User A

For User B (Figure W23), who has had five direct visits in their history, targeting within the next 12 hours after the most recent direct visit on Day 59.5 results in a next-period in-

stantaneous conversion lift of 0.006 but a 15-day total conversion lift of -0.011. This suggests that while the email initially increases the conversion probability, it lowers conversion probabilities below the baseline in subsequent periods. (This effect is similar to retail promotions causing forward buying at the expense of future sales, which is more prominent for User B who is at a later stage close to conversion.) In contrast, targeting 5 days after the direct visit produces a next-period instantaneous conversion lift of 0.019 and a 15-day total conversion lift of 0.019, as the incremental lifts for subsequent periods are statistically insignificant. Therefore, for User B, it is clearly more effective to target 5 days after the direct visit rather than immediately after.

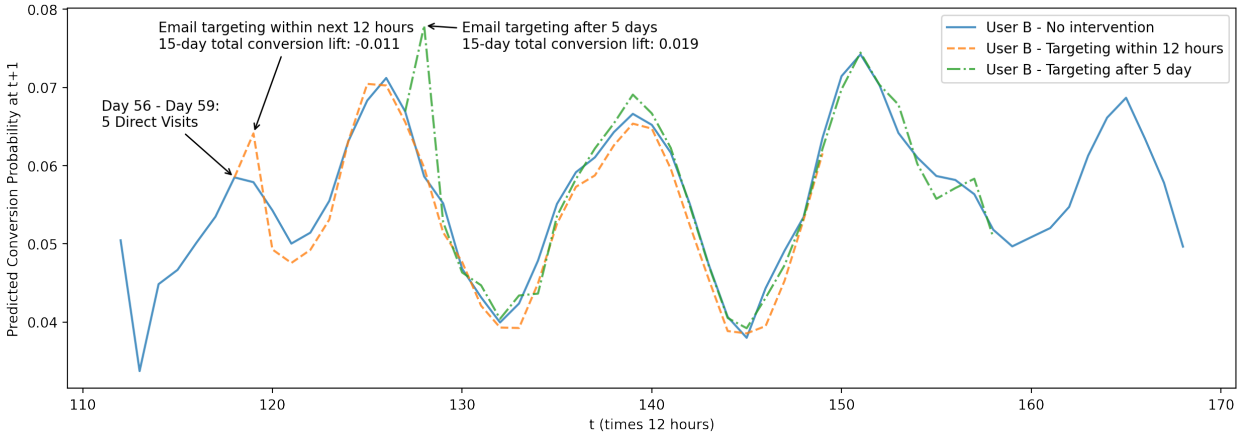


Figure W23: Email Targeting of User B

This example highlights the value of personalized targeting enabled by our model. While implementing such individualized strategies involves intensive computation, the process can be significantly simplified by conducting the analysis at the cohort level, allowing for easier execution without sacrificing effectiveness, as the next example reveals.

Cohort-level targeting timing.

The time-varying impact estimates for direct visits, as shown in Figure 7, reveal an intriguing pattern. The impacts tend to decay quickly moving backward from Day 0 (the day of conversion), reaching a low around Day -7, then increasing until Day -9, before

declining again near Day -15 and peaking at Day -17, with similar oscillations observed over time. This pattern provides valuable insights into the timing of targeting campaigns for cohorts making direct visits on specific days. For instance, the impact of a direct visit on a potential conversion seven days later is likely to be lower compared to its impact on a potential conversion nine days later. If the objective of an email targeting campaign is to enhance this impact, it is more effective to "strike when the iron is hot." Targeting with an email campaign nine days after the direct visit would result in higher incremental conversions than targeting seven days after the visit.

To test these targeting policies, we selected a cohort of users (numbering 489) who visited the hotel website on a specific day (Day 25) and simulated their purchase conversions and lifts compared to the baseline under two scenarios: targeting them seven days after the direct visit versus nine days after the direct visit. Based on 50 prediction simulations, targeting the cohort seven days after the visit resulted in 97.58 incremental conversions over the subsequent 15-day period, whereas targeting them nine days later yielded 104.08 incremental conversions during the same time frame. These results validate the insights derived from the time-varying impact estimates and demonstrate how our model can be leveraged to pinpoint the optimal timing for targeting, thereby achieving higher returns.

Web Appendix D: Application to a Public Dataset

We apply the proposed transformer to a public dataset on Kaggle[1]. The dataset is also in the digital marketing context and has the similar structure as the application data. It includes 586,737 visit sessions from 240,108 unique cookie IDs. Each visit session is from one of the five marketing channels – Facebook, Instagram, Online Display, Online Video and Paid Search. Some sessions are associated with a conversion and the transaction value is available. Table W17 shows the summary statistics for the marketing channels. Table W18 shows the clumpiness of the visits in the dataset (Zhang, Bradlow, and Small 2015).

Table W17: Channel Summary Statistics

Channel	N	Conversion	Conversion Rate
Facebook	175,741	5,301	3.02%
Paid Search	151,440	4,547	3.00%
Online Video	113,302	3,408	3.01%
Instagram	75,201	2,244	2.98%
Online Display	71,053	2,139	3.01%
Total	586,737	17,639	3.01%

Table W18: Visit Clumpiness of the Public Dataset

	N	Nonclumpy (%)	Clumpy (%)
All Customers	240,108	94	6
Multiple-visit Customers	87,445	90	10
Single-visit Customers	152,663	97	3

We apply the same processing steps as in the application section to prepare the data. We treat each unique cookie ID as an individual customer and organize the dataset into a panel data structure. Each period represents a 12-hour window of activity. 50% of the customers in the dataset are held out and the remaining data is divided into five folds for training and validation. Then we train the proposed transformer model, as well as the LSTM

[1]The webpage link for the dataset is https://www.kaggle.com/code/hughhuyton/multitouch-attribution-modelling/notebook

model described in the *Model Comparison* section on the dataset. We include six variables – five channel visit indicators and a conversion indicator. In- and out-of-sample performance comparison is shown in Table W19.

Table W19: Performance Comparison on the Public Dataset

Dependent Variable	In-Sample AUC		Out-of-Sample AUC	
	Proposed Transformer	LSTM	Proposed Transformer	LSTM
Conversion	0.6949	0.5554	0.6904	0.5536
Channel Visit				
Facebook	0.7453	0.6782	0.7445	0.6784
Instagram	0.7515	0.6919	0.7508	0.6914
Online Display	0.7826	0.6895	0.7814	0.6900
Online Video	0.8311	0.8096	0.8316	0.8103
Paid Search	0.6985	0.6592	0.6974	0.6572

Although the performance differs from the results presented in the application due to different and sparser data pattern, the model comparison still demonstrates the superior performance of the proposed transformer model over LSTM.

Web Appendix E: Ablation Experiments

To identify the key components driving the transformer's superior performance, we focus on two critical features: positional encoding and multi-head self-attention, which set it apart from earlier deep learning models. These components enable the transformer to flexibly model time effects and event dependencies. Positional encoding represents time as vectors, while self-attention captures inter-temporal dependencies through attention weights. Multiple heads further enhance this by capturing diverse aspects of these dependencies, complementing each other. The detailed mechanisms are discussed in the *Model* section.

We run an ablation study on the transformer model using simulated datasets generated by autoregressive models (AR1, AR3, AR5, with varying degree of calendar effects, as described in Web Appendix F). We compare three ablation models – transformer without positional encoder, transformer with attention mask that restricts attention solely on the immediate preceding period, and transformer with single head. Table W20 shows the results of the ablation experiment, comparing the performance of the proposed transformer model with various components disabled against the fully configured model. To make the comparison more salient, Figure W24 shows the performance *deviations* in mean cross entropy compared to the fully configured model for each ablation model.

We first remove the positional encoder from the proposed transformer. Since the positional encoder allows the model to recognize the order of touchpoints, its absence prevents the transformer from distinguishing between close and distant events, effectively reducing the history to a "bag of words." As expected, this leads to a performance decline, as confirmed by the ablation experiment results.

Secondly, using attention masks, we restrict the self-attention mechanism to focus solely on the immediate preceding period, masking all other past periods during the prediction of the current period. This step essentially turns the transformer into a first-order Markov model. If transformer relies on self-attention to identify the inter-temporal relationship,

Table W20: Transformer Ablation Experiment on AR Datasets

DGP	Mean Cross Entropy with DGP Probability				Mean AUC			
	Proposed Transformer	Transformer without Positional Encoder	Transformer with Attention Mask	Transformer with Single Head	Proposed Transformer	Transformer without Positional Encoder	Transformer with Attention Mask	Transformer with Single Head
No Calendar Effect								
AR1	0.4521	0.4522	0.4521	0.4529	0.6025	0.6021	0.6009	0.6001
AR3	0.4494	0.4597	0.4657	0.4602	0.6624	0.6260	0.5913	0.6165
AR5	0.4707	0.4816	0.4982	0.4775	0.6790	0.6478	0.5933	0.6618
Weak Calendar Effect								
AR1	0.452	0.4662	0.4523	0.4525	0.6950	0.6589	0.6940	0.6932
AR3	0.3973	0.4168	0.4082	0.4015	0.7067	0.6428	0.6794	0.6964
AR5	0.4735	0.5066	0.5065	0.4837	0.7286	0.6525	0.6640	0.7107
Strong Calendar Effect								
AR1	0.4728	0.5236	0.4957	0.4744	0.8047	0.7404	0.7791	0.8029
AR3	0.3563	0.4432	0.3793	0.3568	0.8359	0.6863	0.8073	0.8353
AR5	0.4196	0.5353	0.4708	0.4191	0.8586	0.7360	0.8140	0.8588

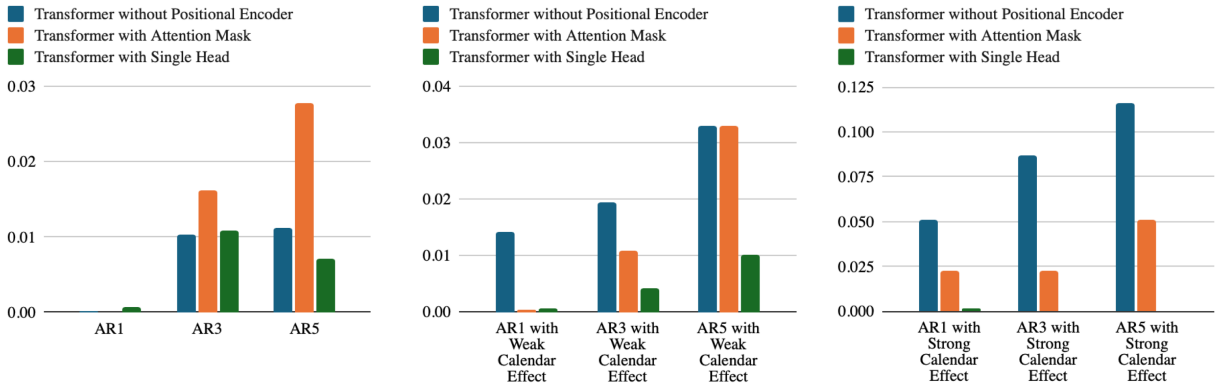


Figure W24: Mean Cross-Entropy Deviation of Ablation Models Compared to the Fully Configured Model on AR Datasets

one would expect the performance to decline significantly when the attention is masked, and such gap will enlarge as the order of the AR DGP becomes larger. Note that for AR1, however, the performance will not change because the prediction only relies on the immediate preceding period. Our simulation results further confirm this (See Figure W24 Transformer with Attention Mask).

Furthermore, identifying calendar effects relies on the time information of each touchpoint, which is embedded in the positional encoding. Thus, compared with the full model specification, the model performance without the positional encoder will decline *more* when there is a calendar effect in the DGP (Figure W24 middle and right sub-figures), compared with when there is no calendar effect (Figure W24 left sub-figure). We observe that as the calendar effect becomes stronger, the performance gap between with and without the positional encoder also becomes larger, showcasing its the important role in identifying the time effects.

Lastly, we reduce the number of heads in the transformer and compare the performance of one head with the default of four heads. The results show that multiple heads give better prediction accuracy (see Figure W24 Transformer with Single Head) than one head, although the performance gap is relatively small because the DGP is not very complex.

We repeat all the ablation experiments on another simulated dataset – the mixture DGP, as described above. We observe varying degree of performance decline when different components are shut off (see Table W21 and Figure W25). Notably, reducing the number of heads has a larger impact on the mixture DGP compared with the AR DGPs, highlighting the critical role of multiple heads in modeling more complex relationships.

Table W21: Transformer Ablation Experiment on the Mixture DGP

Variable	Mean Cross Entropy				Mean AUC			
	Proposed Transformer	Transformer without Positional Encoder	Transformer with Attention Mask	Transformer with Single Head	Proposed Transformer	Transformer without Positional Encoder	Transformer with Attention Mask	Transformer with Single Head
Channel 1	0.6605	0.6618	0.6609	0.6618	0.5312	0.5031	0.5257	0.5073
Channel 2	0.4959	0.4965	0.4962	0.4965	0.5241	0.4982	0.5154	0.5059
Channel 3	0.4072	0.4085	0.4079	0.4085	0.5391	0.5026	0.5250	0.5074
Purchase	0.5383	0.5409	0.5387	0.5404	0.5495	0.5045	0.5444	0.5169

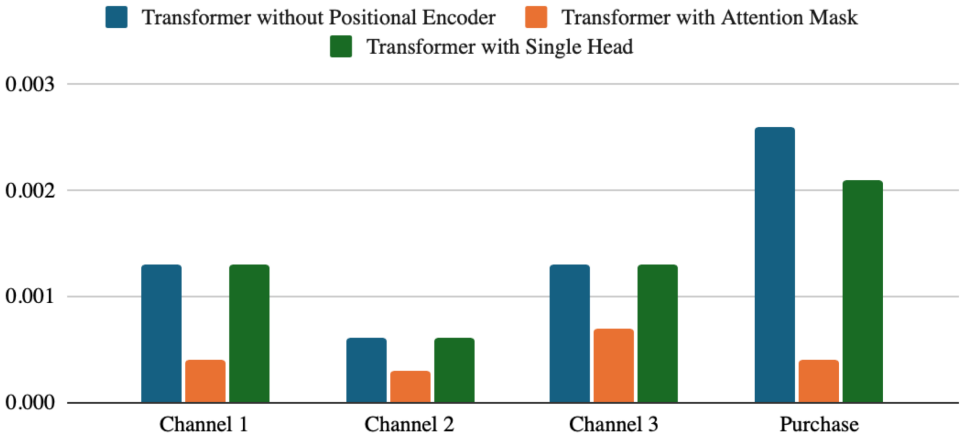


Figure W25: Mean Cross-Entropy Deviation of Ablation Models Compared to the Fully Configured Model on the Mixture DGP

Web Appendix F: Additional Details and Tables on the Simulation Experiments

Autoregressive Model for Simulation

In the *Simulation* section, we use the autoregressive (AR) model with different orders (1, 3, and 5) as the underlying DGP. Here we provide the details on model specifications.

In each period t, a customer i first chooses whether to visit through a channel c, then decide whether to make a purchase at the end of visit. We simulate three marketing channels ($c = 1, 2, 3$). Let $y_{ict} \in \{0, 1\}$ denote whether the customer i has a visit through channel c at period t, and $p_{it} \in \{0, 1\}$ denotes whether the customer i makes a purchase at the end of period t. The utility u_{it}^c for channel c at t is influenced by the customer's visit and purchase behaviors in the previous L periods ($L = 1, 3, 5$). Specifically, for $t > L$,

$$\begin{aligned} P\left(y_{ict} = 1\right) &= \frac{1}{1 + \exp\left(-u_{it}^c\right)}, \\ u_{it}^c &= \alpha_c + \sum_{c'=1,2,3} \sum_{l=1}^{L} \beta_{c'l}^c y_{ic',t-l} + \sum_{l=1}^{L} \rho_l^c p_{i,t-l}. \end{aligned} \tag{W9}$$

where α_c is the baseline utility for channel c. For the first L periods, $u_{it}^c = \alpha_c$. $\beta_{c'l}^c$ is the coefficient that represents the influence of a *visit* through channel c' at $t - l$ on utility for channel c at t. ρ_l^c is the coefficient that represents the influence of a *purchase* at $t - l$ on utility for channel c at t.

If the customer has a visit through any of the three channels, at the end of the visit, they make a decision on whether to make a purchase. The utility for purchase u_{it}^p is constructed similar to the channel utility, which takes the form

$$u_{it}^p = \alpha_p + \sum_{c'=1,2,3} \sum_{l=1}^{L} \beta_{c'l}^p y_{ic',t-l} + \sum_{l=1}^{L} \rho_l^p p_{i,t-l}. \tag{W10}$$

Similarly, α_p is the baseline utility for purchase. For the first L periods, $u_{it}^p = \alpha_p$. $\beta_{c'l}^p$ is

the coefficient that represents the influence of a visit through channel c' at $t-l$ on utility for purchase at t. ρ_l^p is the coefficient that represents the influence of a purchase at $t-l$ on utility for purchase at t. Conditional on having at least one visit, the purchase decision is modeled by $P(p_{it}=1) = 1/\left(1+\exp\left(-u_{it}^p\right)\right)$.

All coefficients are drawn from two uniform distributions where

$$\begin{aligned} \alpha_c, \alpha_p &\sim Uniform\left(-2,-1\right), \\ \beta_{c'l}^c, \rho_l^c; \beta_{c'l}^p, \rho_l^p &\sim Uniform\left(-1,1\right). \end{aligned} \tag{W11}$$

Calendar effects. We further simulate day-of-week and month-of-year calendar effects on top of the AR process described above. The updated utility for channel c is

$$\begin{aligned} u_{it}^c = \alpha_c &+ \sum_{c'=1,2,3}\sum_{l=1}^{L}\beta_{c'l}^c y_{ic',t-l} + \sum_{l=1}^{L}\rho_l^c p_{i,t-l} \\ &+ \sum_{d=1}^{7}\delta_d \cdot \mathbb{I}_{\mathrm{DoW}(t)=d} + \sum_{m=1}^{12}\lambda_m \cdot \mathbb{I}_{\mathrm{MoY}(t)=m}. \end{aligned} \tag{W12}$$

δ_d is the coefficient for the effect of the d-th day of the week ($d=1,2,...,7$, with $d=1$ for Sunday and $d=7$ for Saturday). $\mathbb{I}_{\mathrm{DoW}(t)=d}$ is the indicator function, which equals to 1 if period t corresponds to day d, otherwise 0. λ_m is the coefficient for the effect of the m-th month ($m=1,2,...,12$). $\mathbb{I}_{\mathrm{MoY}(t)=m}$ is the indicator function which equals to 1 if time t falls in month m, otherwise 0.

Similarly, the utility for purchase with calendar effects is

$$\begin{aligned} u_{it}^p = \alpha_p &+ \sum_{c'=1,2,3}\sum_{l=1}^{L}\beta_{c'l}^p y_{ic',t-l} + \sum_{l=1}^{L}\rho_l^p p_{i,t-l} \\ &+ \sum_{d=1}^{7}\delta_d \cdot \mathbb{I}_{\mathrm{DoW}(t)=d} + \sum_{m=1}^{12}\lambda_m \cdot \mathbb{I}_{\mathrm{MoY}(t)=m}. \end{aligned} \tag{W13}$$

We draw two sets of coefficients for calendar effects which we call "weak" and "strong" calendar effects. The coefficients for weak calendar effects are drawn from two uniform

distributions where

$$\begin{aligned} \delta_d^{weak} &\sim Uniform\left(-0.5, 0.5\right), \\ \lambda_m^{weak} &\sim Uniform\left(-1, 1\right). \end{aligned} \tag{W14}$$

And the coefficients for strong calendar effects are drawn from

$$\begin{aligned} \delta_d^{strong} &\sim Uniform\left(-2, 2\right), \\ \lambda_m^{strong} &\sim Uniform\left(-2, 2\right). \end{aligned} \tag{W15}$$

Simulation Experiment Results

We present complete simulation experiment results in the tables below.

Table W22: Model Comparisons on Simulated HMM & Point Process Datasets

Model	Mean Absolute Deviation from the Best Performing Model across the 50 Simulated Datasets			
	Cross Entropy	AUC	Balanced Accuracy	F1 Score
DGP - HMM				
Transformer	0.0004	0.0099	0.0116	0.0006
LSTM	0.0025	0.0281	0.0240	0.0011
HMM	0.0055	0.0042	0.0020	0.0014
Point Process	0.1025	0.0195	0.0160	0.0011
DGP - Point Process				
Transformer	0.0003	0.0091	0.0113	0.0050
LSTM	0.0016	0.0272	0.0245	0.0117
HMM	0.0020	0.0316	0.0177	0.0089
Point Process	0.0101	0.0086	0.0003	0.0005

a) Cross entropy measures the alignment between the distribution of model estimated probability and the true data-generating probability. b) The mean absolute deviation is the absolute deviation of each model from the best performing model averaged across all 50 datasets.

Table W23: Model Comparisons on Simulated AR Datasets

DGP	Mean Cross Entropy				Mean AUC			
	Proposed Transformer	HMM	Point Process	LSTM	Proposed Transformer	HMM	Point Process	LSTM
AR1	0.4521	0.4603	0.4675	0.4521	0.6025	0.5590	0.5307	0.6013
AR3	0.4494	0.4692	0.4754	0.4391	0.6624	0.5894	0.5530	0.6918
AR5	0.4704	0.4972	0.5106	0.4413	0.6810	0.5963	0.5693	0.7460
AR1 with Weak Calendar Effect	0.4520	0.4871	0.4962	0.4572	0.6950	0.5823	0.5522	0.6805
AR3 with Weak Calendar Effect	0.3973	0.4312	0.4351	0.4022	0.7067	0.6005	0.5638	0.6929
AR5 with Weak Calendar Effect	0.4735	0.5236	0.5407	0.4521	0.7286	0.5844	0.5607	0.7661
AR1 with Strong Calendar Effect	0.4728	0.6033	0.6155	0.5079	0.8047	0.6454	0.6207	0.7656
AR3 with Strong Calendar Effect	0.3563	0.4901	0.4791	0.3471	0.8359	0.5783	0.5724	0.8461
AR5 with Strong Calendar Effect	0.4196	0.5973	0.6194	0.3938	0.8586	0.6410	0.5610	0.8775

Table W24: Transformer and LSTM Performance under Different Sample Size under AR5 DGP

Sample Size	Mean Cross Entropy		Mean AUC	
	Proposed Tranformer	LSTM	Proposed Tranformer	LSTM
10,000	0.4704	0.4413	0.6810	0.7460
20,000	0.4803	0.4406	0.7004	0.7469
50,000	0.4504	0.4404	0.7294	0.7473
100,000	0.4435	0.4402	0.7420	0.7473

Table W25: Model Comparisons under Mixture DGP

Variable	Mean Cross Entropy				Mean AUC			
	Proposed Transformer	HMM	Point Process	LSTM	Proposed Transformer	HMM	Point Process	LSTM
Channel 1	0.6605	0.6616	0.6946	0.6616	0.5312	0.5055	0.5028	0.5115
Channel 2	0.4959	0.4963	0.5009	0.4963	0.5241	0.5034	0.5040	0.5088
Channel 3	0.4072	0.4078	0.4098	0.4081	0.5391	0.5003	0.5033	0.5036
Purchase	0.5383	0.5399	0.5459	0.5403	0.5495	0.5065	0.5174	0.5175

Time Series Analysis of the Application Data

We take three channels – Direct, Natural Search, and Email in our applications along with the booking variable, and fit these variables with a simple logistic regression to extract the time fixed effect,

$$y_{ct} = \text{logit}(\lambda_{ct}), \tag{W16}$$

where y_{ct} is the binary variable indicating whether a customer makes a visit or purchase at period t for variable c, and λ_{ct} denotes the time fixed effect to be estimated for variable c. Then we treat λ_{ct} as a time series for each c, and examine the ACF (Autocorrelation Function) and the PACF (Partial-Autocorrelation Function) plots of λ_{ct} for each variable c. We present the two plots for the purchase variable and the direct visit variable respectively in Figure W26 and Figure W27 below.

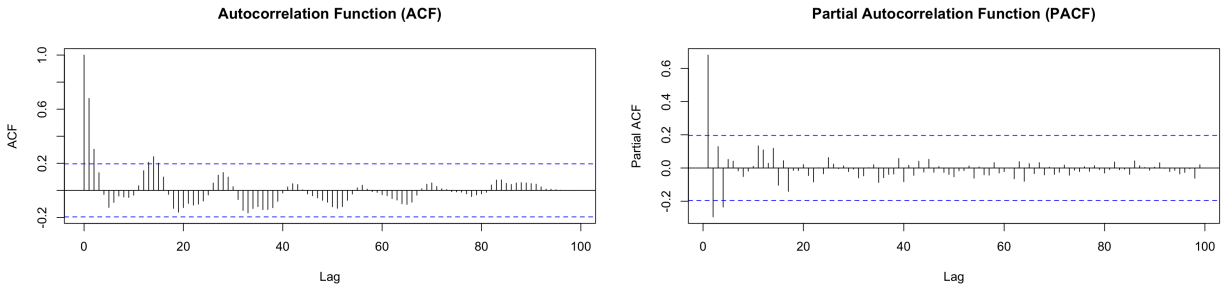


Figure W26: ACF and PACF Plots for Purchase Time Fixed Effect

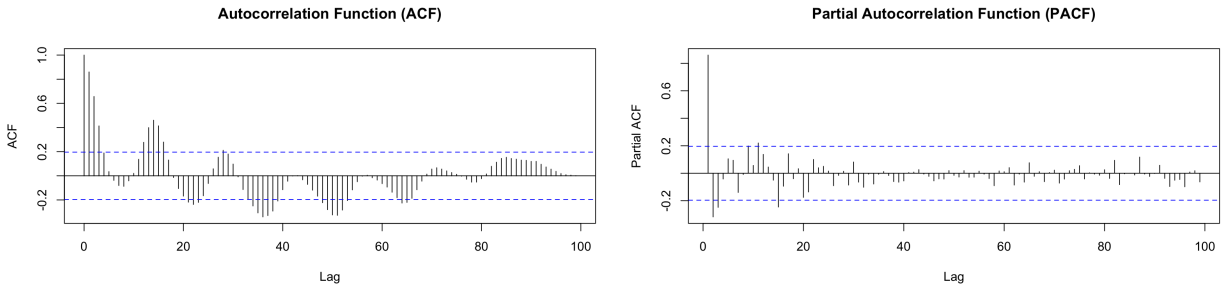


Figure W27: ACF and PACF Plots for Direct Visit Time Fixed Effect

Based on the plots, we fit an ARMA(2,2) model[2] to each λ_{ct}. Then we take the coefficients

[2]We also fit ARMA models of different orders and use the AIC and BIC to guide order selection.

and generate a new time series u_{ct} based on the coefficients of each model. The random errors are sampled from the standard normal distribution. Figure W28 shows the ACF plot of the simulated series for the purchase variable, and Figure W29 shows the ACF plot of the simulated series for the direct visit variable. Both figures show that the generated series have similar autocorrelation structure as the real data shown in Figure W26, W27. Based on the generated time fixed effect u_{ct}, we simulate channel visits from the logistic function $P\left(y_{ict}=1\right)=1/\left(1+\exp\left(-u_{ct}\right)\right)$, where y_{ict} denotes whether customer i has a visit at channel c at period t. Conditional on having a visit, the purchase decision is simulated by $P(p_{it}=1)=1/\left(1+\exp\left(-u_t^p\right)\right)$, where p_{it} is the purchase decision and u_t^p is the generated time fixed effect for purchase.

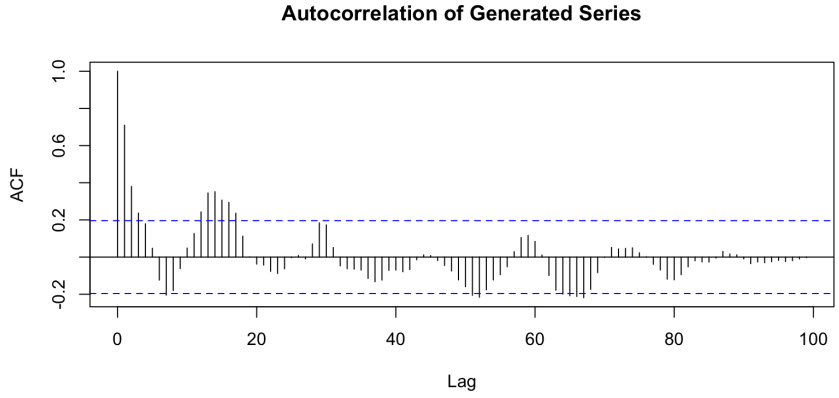


Figure W28: ACF Plot for *Simulated* Purchase Time Fixed Effect

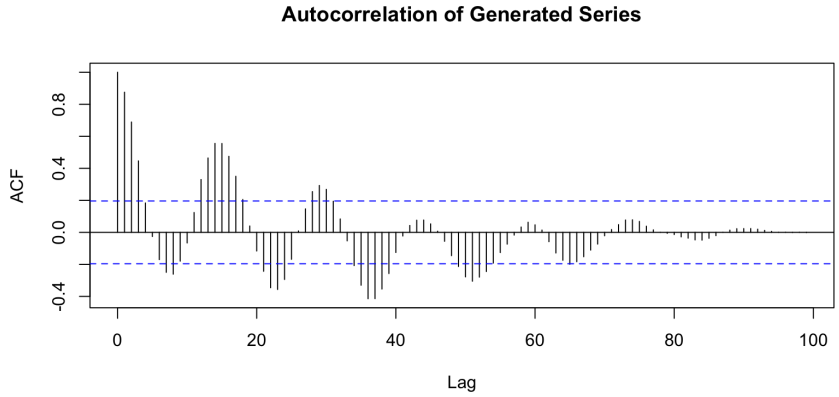


Figure W29: ACF Plot for *Simulated* Direct Visit Time Fixed Effect

Web Appendix G: Results for 6-Hour and 24-Hour Periods

The application data in the main text is organized into 12-hour periods. Here, we conduct a robustness check using alternative period lengths of 6 hours and 24 hours. Table W26 shows the in-sample and out-of-sample AUC and balanced accuracy for 6-hour period. And Table W27 shows the results for 24-hour period. Both results are comparable to the result of the 12-hour period.

Table W26: Performance of Proposed Transformer on 6-Hour Period

Dependent Variable	AUC		Balanced Accuracy	
	In-Sample	Out-of-sample	In-Sample	Out-of-sample
Purchase				
Booking	0.9592	0.9285	0.9063	0.8673
Weekend Stay Booking	0.9658	0.9189	0.9093	0.8668
Channel Visit				
AFFILIATE	0.9959	0.9246	0.9824	0.8595
B2B	0.9992	0.8986	0.9939	0.8231
DIRECT	0.9322	0.9012	0.8706	0.8302
DISPLAY	0.9914	0.9083	0.9655	0.8357
ECONFO AND PRE-ARRIVAL EMAIL	0.9852	0.9246	0.9622	0.8556
EMAIL	0.9841	0.9267	0.9607	0.8409
EMERGING TECHNOLOGIES	0.9973	0.8926	0.9811	0.8319
NATURAL SEARCH	0.9567	0.9058	0.9110	0.8392
PAID SEARCH	0.9770	0.9015	0.9485	0.8217
REFERRAL ENGINE	0.9963	0.9273	0.9824	0.8634
RESLINK	0.9943	0.9322	0.9731	0.8622
SOCIAL MEDIA	0.9992	0.9218	0.9953	0.8581
UNPAID REFERRER	0.9782	0.9287	0.9393	0.8471

Table W27: Performance of Proposed Transformer on 24-Hour Period

Dependent Variable	AUC		Balanced Accuracy	
	In-Sample	Out-of-sample	In-Sample	Out-of-sample
Purchase				
Booking	0.9344	0.9111	0.8605	0.8499
Weekend Stay Booking	0.9359	0.9038	0.8636	0.8474
Channel Visit				
AFFILIATE	0.9926	0.9112	0.9703	0.8334
B2B	0.9993	0.9404	0.9894	0.8911
DIRECT	0.9233	0.8831	0.8556	0.8084
DISPLAY	0.9748	0.9004	0.9234	0.8222
ECONFO AND PRE-ARRIVAL EMAIL	0.9745	0.9016	0.9231	0.8137
EMAIL	0.9749	0.9064	0.9277	0.8176
EMERGING TECHNOLOGIES	0.9990	0.8931	0.9930	0.8193
NATURAL SEARCH	0.9319	0.8865	0.8614	0.8126
PAID SEARCH	0.9581	0.8770	0.9133	0.7999
REFERRAL ENGINE	0.9807	0.9114	0.9334	0.8450
RESLINK	0.9807	0.9025	0.9399	0.8320
SOCIAL MEDIA	0.9943	0.8967	0.9678	0.8369
UNPAID REFERRER	0.9678	0.9139	0.9131	0.8301

References

Brabec, Jan, Tomáš Komárek, Vojtěch Franc, and Lukáš Machlica "On Model Evaluation Under Non-constant Class Imbalance," Valeria V. Krzhizhanovskaya, Gábor Závodszky, Michael H. Lees, Jack J. Dongarra, Peter M. A. Sloot, Sérgio Brissos, and João Teixeira, editors, "Computational Science – ICCS 2020," Vol. 12140., pages 74–87, Cham: Springer International Publishing (2020) https://link.springer.com/10.1007/978-3-030-50423-6_6, series Title: Lecture Notes in Computer Science.

Brodersen, Kay Henning, Cheng Soon Ong, Klaas Enno Stephan, and Joachim M. Buhmann "The Balanced Accuracy and Its Posterior Distribution," "2010 20th International Conference on Pattern Recognition," pages 3121–3124, Istanbul, Turkey: IEEE (2010) http://ieeexplore.ieee.org/document/5597285/.

Buda, Mateusz, Atsuto Maki, and Maciej A. Mazurowski (2018), "A systematic study of the class imbalance problem in convolutional neural networks," *Neural Networks*, 106, 249–259 https://linkinghub.elsevier.com/retrieve/pii/S0893608018302107.

CampaignMonitor "Ultimate Email Marketing Benchmarks for 2022: By Industry and Day," Technical report, Campaign Monitor by MARIGOLD (2022) https://www.campaignmonitor.com/resources/guides/email-marketing-benchmarks/#five.

Caplin, Andrew, Daniel Martin, and Philip Marx "Calibrating for Class Weights by Modeling Machine Learning," (2022) https://arxiv.org/abs/2205.04613, version Number: 2.

Chen, Weijie, Berkman Sahiner, Frank Samuelson, Aria Pezeshk, and Nicholas Petrick (2018), "Calibration of medical diagnostic classifier scores to the probability of disease," *Statistical Methods in Medical Research*, 27 (5), 1394–1409 https://journals.sagepub.com/doi/10.1177/0962280216661371.

Davies, Alex, Petar Veličković, Lars Buesing, Sam Blackwell, Daniel Zheng, Nenad Tomašev, Richard Tanburn, Peter Battaglia, Charles Blundell, András Juhász, Marc Lackenby, Geordie Williamson, Demis Hassabis, and Pushmeet Kohli (2021), "Advancing mathematics by guiding human intuition with AI," *Nature*, 600 (7887), 70–74 https://www.nature.com/articles/s41586-021-04086-x.

Davis, Jesse and Mark Goadrich "The relationship between Precision-Recall and ROC curves," "Proceedings of the 23rd international conference on Machine learning - ICML '06," pages 233–240, Pittsburgh, Pennsylvania: ACM Press (2006) http://portal.acm.org/citation.cfm?doid=1143844.1143874.

Feng, Tianshu, Zhipu Zhou, Joshi Tarun, and Vijayan N. Nair "Comparing Baseline Shapley and Integrated Gradients for Local Explanation: Some Additional Insights," (2022) https://arxiv.org/abs/2208.06096, version Number: 1.

Fernando, K. Ruwani M. and Chris P. Tsokos (2022), "Dynamically Weighted Balanced Loss: Class Imbalanced Learning and Confidence Calibration of Deep Neural Networks," *IEEE Transactions on Neural Networks and Learning Systems*, 33 (7), 2940–2951 https://ieeexplore.ieee.org/document/9324926/.

Grandini, Margherita, Enrico Bagli, and Giorgio Visani "Metrics for Multi-Class Classification: an Overview," (2020) https://arxiv.org/abs/2008.05756, version Number: 1.

Jeni, Laszlo A., Jeffrey F. Cohn, and Fernando De La Torre "Facing Imbalanced Data–Recommendations for the Use of Performance Metrics," "2013 Humaine Association Conference on Affective Computing and Intelligent Interaction," pages 245–251, Geneva, Switzerland: IEEE (2013) http://ieeexplore.ieee.org/document/6681438/.

Johnson, Justin M. and Taghi M. Khoshgoftaar (2019), "Survey on deep learning with class imbalance," *Journal of Big Data*, 6 (1), 27 https://journalofbigdata.springeropen.com/articles/10.1186/s40537-019-0192-5.

Kaur, Harsurinder, Husanbir Singh Pannu, and Avleen Kaur Malhi (2020), "A Systematic Review on Imbalanced Data Challenges in Machine Learning: Applications and Solutions," *ACM Computing Surveys*, 52 (4), 1–36 https://dl.acm.org/doi/10.1145/3343440.

Khirirat, Sarit, Hamid Reza Feyzmahdavian, and Mikael Johansson "Mini-batch gradient descent: Faster convergence under data sparsity," "2017 IEEE 56th Annual Conference on Decision and Control (CDC)," pages 2880–2887, Melbourne, Australia: IEEE (2017) http://ieeexplore.ieee.org/document/8264077/.

Kim, Yechan, Younkwan Lee, and Moongu Jeon "Imbalanced Image Classification with Complement Cross Entropy," (2020) https://arxiv.org/abs/2009.02189, version Number: 4.

Kubat, Miroslav and Stan Matwin (1997), "Addressing the curse of imbalanced training sets: one-sided selection.," *Icml*, 97 (1), 179.

Li, Mu, Tong Zhang, Yuqiang Chen, and Alexander J. Smola "Efficient mini-batch training for stochastic optimization," "Proceedings of the 20th ACM SIGKDD international conference on Knowledge discovery and data mining," pages 661–670, New York New York USA: ACM (2014) https://dl.acm.org/doi/10.1145/2623330.2623612.

Liaw, Richard, Eric Liang, Robert Nishihara, Philipp Moritz, Joseph E. Gonzalez, and Ion Stoica "Tune: A Research Platform for Distributed Model Selection and Training," (2018) http://arxiv.org/abs/1807.05118, arXiv:1807.05118 [cs].

Lundberg, Scott and Su-In Lee (2017), "A Unified Approach to Interpreting Model Predictions," https://arxiv.org/abs/1705.07874, publisher: arXiv Version Number: 2.

Novakovsky, Gherman, Nick Dexter, Maxwell W. Libbrecht, Wyeth W. Wasserman, and Sara Mostafavi (2022), "Obtaining genetics insights from deep learning via explainable artificial intelligence," *Nature Reviews Genetics* https://www.nature.com/articles/s41576-022-00532-2.

Senior, Andrew W., Richard Evans, John Jumper, James Kirkpatrick, Laurent Sifre, Tim Green, Chongli Qin, Augustin Žídek, Alexander W. R. Nelson, Alex Bridgland, Hugo Penedones, Stig Petersen, Karen Simonyan, Steve Crossan, Pushmeet Kohli, David T. Jones, David Silver, Koray Kavukcuoglu, and Demis Hassabis (2020), "Improved protein structure prediction using potentials from deep learning," *Nature*, 577 (7792), 706–710 http://www.nature.com/articles/s41586-019-1923-7.

Shrikumar, Avanti, Peyton Greenside, and Anshul Kundaje (2017), "Learning Important Features Through Propagating Activation Differences," https://arxiv.org/abs/1704.02685, publisher: arXiv Version Number: 2.

Sundararajan, Mukund and Amir Najmi "The Many Shapley Values for Model Explanation," Hal Daumé III and Aarti Singh, editors, "Proceedings of the 37th International Conference on Machine Learning," Vol. 119. of *Proceedings of Machine Learning Research*, pages 9269–9278, PMLR (2020) https://proceedings.mlr.press/v119/sundararajan20b.html.

Sundararajan, Mukund, Ankur Taly, and Qiqi Yan (2017), "Axiomatic Attribution for Deep Networks," https://arxiv.org/abs/1703.01365, publisher: arXiv Version Number: 2.

Tian, Junjiao, Yen-Cheng Liu, Nathaniel Glaser, Yen-Chang Hsu, and Zsolt Kira "Posterior recalibration for imbalanced datasets," "Proceedings of the 34th International Conference on Neural Information Processing Systems," NIPS '20, Red Hook, NY, USA: Curran Associates Inc. (2020) Event-place: Vancouver, BC, Canada.

Vehtari, Aki, Andrew Gelman, Daniel Simpson, Bob Carpenter, and Paul-Christian Bürkner (2021), "Rank-normalization, folding, and localization: An improved $\widehat{R}$ for assessing convergence of MCMC," *Bayesian Analysis*, 16 (2) http://arxiv.org/abs/1903.08008, arXiv:1903.08008 [stat].

Wei, Qiong and Roland L. Dunbrack (2013), "The role of balanced training and testing data sets for binary classifiers in bioinformatics," *PloS One*, 8 (7), e67863.

Zhang, Yao, Eric T. Bradlow, and Dylan S. Small (2015), "Predicting Customer Value Using Clumpiness: From RFM to RFMC," *Marketing Science*, 34 (2), 195–208 https://pubsonline.informs.org/doi/10.1287/mksc.2014.0873.