2023 IEEE/ACM International Conference on Advances in Social Networks Analysis and Mining (ASONAM)

# Quantifying customer interactions on ML optimized page layouts

Nikki Gupta
*gupnikk@amazon.com*

Prakash Mandayam Comar
*prakasc@amazon.com*

***Abstract*—In online businesses, personalization of site content is crucial for providing a better user experience and increasing customer engagement. Machine learning algorithms are often used to analyze customer data such as browsing behavior, purchase history to tailor the website content to each individual customer's preferences and needs. However, measuring the success of these personalized experiences can be challenging. While the ultimate goal is to convert customer visits into purchase sessions, tracking individual customer behavior can be difficult at scale. As a result, businesses often rely on aggregate metrics such as site-wide conversion rates, sales, and revenue to evaluate the effectiveness of their personalization efforts. However, it's important to understand individual customers' experiences with these ML-optimized pages. To address this, we propose a supervised ML model that quantifies customer engagement while browsing auto-optimized web pages, by building a customer site interaction score (*CSI* score). We first introduce a novel representation of customer click logs as a tree data structure induced by the webpage's DOM structure. Then, we propose a novel attention model on the tree structure that performs vertical attention across the depth of the tree and horizontal attention across the sequence of trees to summarize customer interactions. The effectiveness of the proposed approach is evaluated using click logs obtained from the e-commerce domain.**

## I. Introduction

In the realm of e-commerce, customers frequently seek detailed information about products such as descriptions, images, reviews and ratings, pricing history, return policies, and more, in order to make informed purchasing decisions. This has resulted in businesses competing to organize and present personalized information about their products/services that is relevant to their customers. However, the amount of relevant content often exceeds the website's capacity, requiring an automated algorithm to render the contents consistently, while adhering to business policies and metric guidelines. To accomplish this, online businesses often use machine learning systems [36, 2] to optimize the page layouts and content presented to customers, focusing on critical business metrics such as click rate for search results or conversions on e-commerce domains. These ML systems include techniques such as pagewise recommendation systems, layoutGAN, and optimization algorithms, among others [13, 17, 25, 28].

The effectiveness of machine learning models that optimize page layouts and content is typically evaluated based on aggregate business metrics, such as clicks and conversions, which often follow Pareto's law, [12] in that most metrics are realized ($\sim$ 80%) from a smaller number of customers ($\sim$ 20%). However, understanding the pain points experienced by individual customers during their interactions with ML-rendered pages is crucial for providing feedback and improving the models. To achieve this, it is necessary to quantify the difficulty that customers experience while navigating web pages with various user interface (UI) elements as they progress through the shopping funnel. In this study, we propose a supervised ML model that quantifies customer engagement while browsing auto-optimized web pages by generating a customer site interaction score *CSI score*. The model considers all interactions performed by the customer on the site and produces a real number between 0 and 1, where scores close to 1 indicate that the auto-rendering has successfully guided the customer towards a purchase, while scores close to 0 suggest that the auto-rendering is causing customers to abandon their session.

In this work, we propose Treeformer model, a novel architecture that utilizes transformer attention to process multivariate time series data organized in a tree-like structure. We bring twofold novelty to this problem, as given below.

- **Browse tree**: We represent the customer click data on a given page as a browse tree induced by the corresponding webpage DOM structure [34, 24, 3] and process the sequence of interactions across pages as a sequence of trees.
- **Tree Attention**: We propose a novel attention model on tree structure that performs vertical attention across elements at different depth and horizontal attention across a sequence of trees to summarize the customer interactions on the site.

The rest of the paper is organized as follows. In section III we introduce the notations, data and discuss the site DOM structure which we leverage to convert the clicklogs into a tree like structure. In section IV we discuss two transformer attention based ML approaches to process sequence data: - one that uses clickstream as classic multivariate timeseries data and other using the proposed tree structure. Finally, in section V, we evaluate the performance between the two approaches and

*ASONAM '23*, November 6-9, 2023, Kusadasi, Turkey
© 2023 is held by the owner/author(s)
ACM ISBN 979-8-4007-0409-3/23/11
https://doi.org/10.1145/3625007.3627519

highlight the advantages of the proposed approach.

## II. Related Works

The fragmentation of web pages into distinct components offers a dynamic framework for rendering customized designs that cater to the preferences of users. A common approach is to predict relevance scores for each module using a pointwise model [1, 6]. Ranking systems have been utilized to optimize constrained objectives from multiple stakeholders [8, 1]. Multi-objective optimization problems has also been tackled in search pages [15, 27, 31, 32] to tailor the page layout to the user's preferences in parallel with search ranking results. These approaches aim to optimize the layout of the web-page to improve various business metrics, including click-through rate [9, 23], revenue [8], and customer conversion [21]. In addition to probabilistic ranking models [33], contextual bandits [14] provide a natural framework to learn from user behavior. Recent approaches have extended the success of contextual bandits in news and Ad recommendations [18, 9] by modeling the problem as a Markov decision process and solving it using deep reinforcement learning [28]. Existing methods do not explicitly account for the customer's browsing experience on the rendered content. To the best of our knowledge, this is the first work that attempts to quantify the customer-site interactions into a numeric score which can be fed as an explicit input to enhance existing page layout optimization systems.

We utilize customer clicklogs which is a multi-variate time series sequence to compute the *CSI* score. Given the recent advancements of self-attention in sequence modelling, we leverage transformers [29] to summarize the click sequence. Several recent studies have proposed hybrid attention frameworks that combine local and global attention [22, 19, 4, 26] to achieve better performance, such that global attention captures long-range dependencies and local attention captures short-term dependencies. We leverage this paradigm to attend over our tree-like hierarchical data structure. The closest approach to ours is the multivariate sequence transformer [10] described as Mformer in section IV-A. However, this approach flattens the multi-variate sequence into a long sequence, $N * L$ where $N$ is the number of features and $L$ is the sequence length. Our proposed approach, Treeformer, incorporates a sophisticated attention mechanism that utilizes masking to preserve the tree-like structure of the data and exploit its inherent characteristics. Our experiments, both offline and in A/B testing, demonstrate the effectiveness of preserving the tree structure of the data while utilizing attention mechanisms.

## III. Preliminaries

The layout of webpages follow the DOM structure [7, 20, 11, 5, 37] which organizes the page in object-oriented framework for easy and programmable manipulation of the content and rendering. This introduces a hierarchical tree like structure in organizing the entire site, as a result the sequence of customer interactions captured in the click logs has a hierarchical structure [30]. The first three features in the hierarchy, denoted by $h = 1, ..3$, provide insight into the specific activity page being viewed by the customer, such as "Product", "Search", "Checkout" etc. Moreover, the hierarchy structure allows for an increasingly granular representation of the rendered page, as we move from $h = 1$ to $h = 3$. Feature $h = 4$ provides information on the specific product or category being browsed by the customer, while feature $h = 5$ refers to actions taken on widgets within the page. In addition, we incorporate the timestamp associated with each customer click as a feature denoted by $h = 6$. Table I lists these hierarchical sequence features along with their order in hierarchy and the number of unique tokens present in each sequence. Notice the number of unique tokens increase as h increases from 1 to 6, suggesting an increase in information as we move towards the leaf node. However, the number of transient and missing entries also increase as we move towards the leaf node.

The customer browse tree data is represented as a multivariate sequence represented by $\mathbf{x}^h_{it}$ where index $i = 1..N$ denote the customer, $t = 1..L$ index into the length of sequence and $h = 1, ..H$ denote various sequence features ordered by the hierarchy. For a customer, $c_i$ we have

$$c_i = [\{x^0_{i0}, x^0_{i1}, ...x^0_{iL}\}, .....\{x^5_{i0}, x^5_{i1}, ...x^5_{iL}\}, \{t_{i0}, t_{i1}, ...t_{iL}\}]$$

where $\mathbf{x}^6_{it}$ is denoted by $t_{i0}, ..t_{it}$ as it corresponds to timestamps. In this work, we model the site interaction score estimation as a supervised problem using a pairwise rank loss, where the model takes as input a pair of customer browse tree data and rank the customer with higher conversion ahead of customer with lower conversion. This is because the final goal is to adapt the UX such that it results in increased conversions and lower session abandonment. Here, we postulate that the customer's ease of interaction with the UX is correlated with conversion rate and projecting the conversion rate on the browse activities, with no other input feature, would help unravel this latent score. It can be argued that instead of learning an ML model, the rank ordering induced by conversion rate can be directly used as a proxy, however, this metric is not available for customers who are not logged in to the website.

The conversion rate can be computed at many granularities namely conversion per visit or weekly, monthly and yearly conversions. We use a combination of these conversion rates to learn the *CSI* scores, as explained in Section V. Let $y_i$ denote the weighted combination of conversion rates, then our problem is to design a neural architecture that takes as input the customer browse tree to learn the rank ordering induced by this weighted conversion rate. One simple approach to learn this conversion rate is to design a neural network function that consumes each sequence feature separately and generates an embedding per sequence, which is then passed on to a back propagation network to learn $y_i$. Consider the following structure

$$\mathbf{e}^h_{it} = g_h(\mathbf{x}^h_{it}), \qquad \mathbf{e}_i = (\mathbf{e}^1_{it}|\mathbf{e}^2_{it}|\mathbf{e}^3_{it}...|\mathbf{e}^H_{it}), \qquad \hat{y}_i = f(\mathbf{e}_i) \quad (7)$$

where $g_h$ could represent a sequence to embedding function like RNN, LSTM or GRU units and $\mathbf{e}_i$ denote concatenation of embeddings generated for each feature and finally $f$ denote

TABLE I
THIS TABLE LISTS ALL THE SEQUENCE FEATURES USED BY THE ML MODEL. HERE $h$ REFER TO HIERARCHY WITH HIGHER $h$ INDICATING CLOSER TO LEAF NODE IN THE BROWSE TREE. AS SHOWN THE NUMBER OF MISSING TOKENS AND THE NUMBER OF OVERLAPPING TOKENS BETWEEN SUCCESSIVE TIME INTERVAL INCREASE AS WE MOVE FROM ROOT TO LEAF.

| Hier(h) | Description | Unique token | Overlap | Missing |
|---|---|---|---|---|
| 1 | Rendered Page - Product, Search, Browse, Checkout | <10 | 100% | 0 |
| 2 | Refinement of h=1 | 1000+ | 80% | 17% |
| 3 | Refinement of h=2 | 5000+ | 32% | 20% |
| 4 | Product/Browse Category | 1M+ | 11% | 58% |
| 5 | Actions on widgets within Page | 1M+ | 9% | 76 % |
| 6 | Click Timestamp | Inf | 0% | 0% |

$$\begin{aligned}
&\text{Embed tokens} & \mathbf{e}_{it}^{h} &= \text{Embed}(\mathbf{x}_{it}^{h}), \quad \mathbf{e}_{it}^{h} \in \mathbb{R}^{d} & & h \leq 5 & (1)\\
&\text{Embed Time} & \mathbf{e}_{it}^{time} &= \texttt{Time2Vec}(\mathbf{x}_{it}^{6}), \ \mathbf{e}_{it}^{time} \in \mathbb{R}^{d} & & & (2)\\
&\text{Project token-time} & \mathbf{e}_{it}^{ht} &= \text{MLP}(\mathbf{e}_{it}^{h}, \mathbf{e}_{it}^{tme}) & & 2 \leq h \leq 5 & (3)\\
&\text{Embed Hierarchy} & \mathbf{v}_{h} &= \text{Embed}(h) \quad \mathbf{v}_{h} \in \mathbb{R}^{d} & & 2 \leq h \leq 5 & (4)\\
&\text{Horizontal Attn} & \underline{\mathbf{a}_{it}^{h}} &= \text{MSA Encoder}(v_h + \mathbf{e}_{it}^{ht} + \mathbf{e}_{it}^{1}) & & O(L^2) & (5)\\
&\text{Vertical Attn} & \mathbf{m}_{it}^{h} &= \text{FFR}(\underline{\mathbf{a}_{it}^{2}}|\underline{\mathbf{a}_{it}^{3}}|\underline{\mathbf{a}_{it}^{4}}|\underline{\mathbf{a}_{it}^{5}}) & & O(Ld) & (6)
\end{aligned}$$

a multi layer perceptron (MLP). A key drawback with this approach is that interactions between features are not explicit and indirect interactions happens at MLP layer where different feature embeddings in $\mathbf{e}_i$ interact with each other to determine the activation threshold. Understanding the explicit interactions across the hierarchy is required to distinguish similar looking customer interactions. For example, two customers spend same time on Product page but one keeps jumping across product by clicking on the *recommended products* widget whereas other spent time on reading descriptions, reviews and rating etc should be attended to differently even if they have same conversion rates. In the absence of explicit interactions, the model may learn the conversion based on time spent on a particular page instead of the customer's interactions on the page. In the following, we propose an approach that uses transformer based architecture to score the browse tree data for each page for their propensity towards conversion.

## IV. PROPOSED APPROACH

Our goal is to design a neural architecture that takes as input all the sitewide interactions and generates distinct interaction scores for each page type, reflecting the ease of browsing the page. In doing so, the model should primarily consider the impact of interactions in the page towards conversion, as well as accurately quantify the secondary effect (downstream effect) of these interactions on other pages towards predicting the conversion rate. In the following, we propose a novel transformer based architecture that takes as input the entire customer browse tree and predicts distinct page-wise scores for each page by considering the page specific interactions as well as allowing only relevant information from other pages using clever gating mechanism. Broadly, our approach has two steps:- 1) horizontal self attention among the value or tokens in the feature sequence and 2) vertical attention between tokens/values across the features in hierarchy.

### A. Mformer: Multivariate Transformer

In this section we present a baseline network architecture that takes as input the six sequence features $\mathbf{x}_{it}^{h}$, transforms them into sequence of embedding $\mathbf{ae}_{it}^{h}$, using the transformer based self attention mechanism. Our architecture, summarized below, is based on the multivariate sequence transformer [10].

First, each sequence feature is transformed into sequence of embeddings by applying a look up embedding on each token in the sequence. Then the ordering of the interactions are encoded using the Time2Vec [16] layer applied on the sequence of timestamps. This transformer converts the absolute date-time value into sinusoidal patterns of learned offsets and wavelengths. We append the token embedding and time embedding and project onto new embedding space using a simple feed forward network. This sequence is denoted by variable $\mathbf{e}_{it}^{h-tme}$, which can be interpreted as time aware encoding of customer interactions across all the features. We encode the position of each sequence feature in the hierarchy using simple look up embedding and add this to $\mathbf{e}_{it}^{h-tme}$ before passing into a multihead self attention layer. Here a token at position $t$ in a feature attends to other tokens positioned before $t$ in order to understand the context of past interactions that resulted in customer journey towards it as well as peek into the future, tokens at positions greater than $t$ to understand the

$$\begin{aligned}
&\text{Embed tokens} & \mathbf{e}_{it}^{h} &= \text{Embed}(\mathbf{x}_{it}^{h}), & \mathbf{e}_{it}^{h} &\in \mathbb{R}^{d^h} & h &\leq 5 && (8)\\
&\text{Embed Time} & \mathbf{e}_{it}^{time} &= \texttt{Time2Vec}(\mathbf{x}_{it}^{6}), & \mathbf{e}_{it}^{time} &\in \mathbb{R}^{d^t} & & && (9)\\
&\text{Project token-time} & \mathbf{e}_{it}^{ht} &= \text{MLP}(\mathbf{e}_{it}^{h}, \mathbf{e}_{it}^{tme}), & \mathbf{e}_{it}^{h} &\in \mathbb{R}^{d^h} & 2 \leq h &\leq 5 && (10)\\
&\text{Embed Hierarchy} & \mathbf{v}_{h} &= \text{Embed}(h) & \mathbf{v}_{h} &\in \mathbb{R}^{d^h} & 2 \leq h &\leq 5 && (11)\\
&\text{Horizontal Attn} & \underline{\mathbf{a}_{it}^{h}} &= \text{MSA Encoder}_h(v_h + \mathbf{e}_{it}^{ht} + \mathbf{e}_{it}^{1}) & & & O(L^2) & && (12)\\
&\text{Projection up} & \mathbf{u}_{it}^{h} &= \text{MLP}(\underline{\mathbf{a}_{it}^{h}}), & \mathbf{u}_{it}^{h} &\in \mathbb{R}^{d} & & && (13)\\
&\text{Vertical Attn} & \underline{\mathbf{z}_{it}^{h}} &= \text{FFR(Concat(}\mathbf{u}_{it}^{h})) & & & p &\leq 4 && (14)\\
&\text{Projection Down} & \mathbf{m}_{it}^{h} &= \text{MLP}(\underline{\mathbf{z}_{it}^{h}}),\ \mathbf{m}_{it}^{h} \in \mathbb{R}^{d^h} & & & & && (15)
\end{aligned}$$

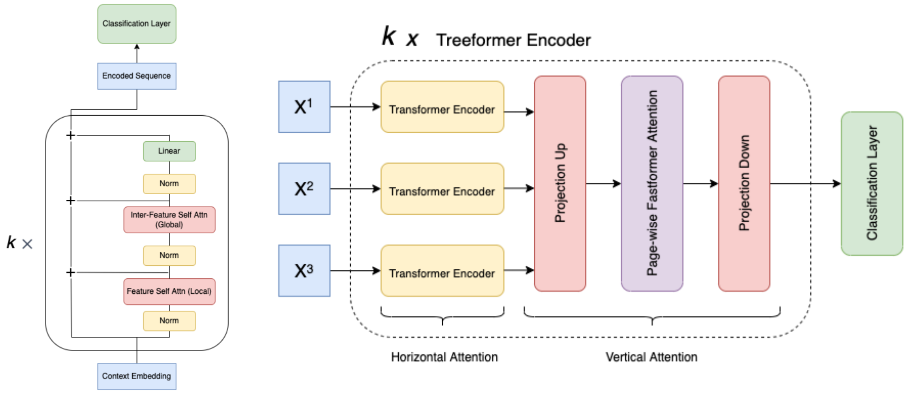

Fig. 1. This figure highlights the difference between Mformer(left) and Treeformer(right) architecture.

downstream effect of the current interaction. We label this as horizontal attention as the field of attention is restricted to a fixed depth in the browse tree (i.e. one feature at a time). Finally, the horizontally attended sequence of features are appended next to each other (denoted by symbol |) on which we run another transformer named Fastformer (FFR) [35]. We label this procedure as vertical attention, as the focus of attention is across the feature hierarchy in the browse tree. The horizontal attention is done on each feature of length L, resulting in $O(L^2)$ complexity, and the vertical attention is done on appended sequence of length 5*L. Here, the regular self attention results in result in complexity by a factor of 25, so we use the additive attention described in [35].

The aforementioned architecture uses two set of transformer encoding, the first one operates on sequence of length L, which combines all the sitewide interactions inside each feature. The second one operates on sequence of length $H \times L$, obtained by appending all the H hierarchical features. Since the self attention is of $O(L^2)$ computational complexity, we use additive transformer called Fastformer given in [35]. However, there are two problems with this approach, 1) Since we're feeding all sequences to a single encoder, they share a common embedding and model size. However, there is a drastic increase in vocabulary size from less than 10 unique tokens in $h = 1$ to tens of thousands for $h = 5$. A model size proportional to the vocabulary size would be sufficient to process each sequence, but we are projecting all tokens to a single higher dimension, which increases our compute and memory requirements without any added advantage. 2) As observed in our data, there is a non-uniform distribution of clicks across page types. For instance, the Product page has maximum no. of clicks while the browse page has minimum no. of clicks. Taking a global attention across all tokens for all page types fails to capture intra-page patterns within the browse page due to attention bias towards the large number of Product page tokens. This leads to poor performance for pages with less number of clicks. To tackle these two issues, we improve upon the above approach to propose Treeformer

### B. Treeformer

In this section, we present the Treeformer which addresses the drawbacks of Mformer discussed in Section IV-A, by modifying the local and global attention mechanism. The overall approach is summarized in equations 8-15.

In contrast to Mformer, we embed tokens in each feature to variable embedding size $d^h$, that is proportional to its vocabulary size. Firstly, this makes the model computationally efficient. We also get significant performance gain, as larger than required embedding dimension cause noisy learning that does not generalize well. As with Mformer, the Time2Vec embeddings are shared across all features where, they are concatenated to the token embeddings and then projected to a feature space of size same as token embedding using MLP. The projected token-time embeddings and hierarchy embeddings are added and fed into the Treeformer encoder which, like Mformer, consists of two attention modules - horizontal and vertical attention. But their functionalities differ from Mformer in following ways: 1) The encoder deals with varying sequence length using projection matrices to transform the embedding to common dimension and back to original dimension. 2) The vertical attention is restricted to contents of a given broad page type (subtree under the page type as denoted by $h = 1$) and does not use information across all browsing activities. Equations 12 to 15 capture this process where the projection matrices are used to bring sequences to the same dimension for performing self attention. Restricting the attention to subtree greatly reduce the computation cost of attending to all interactions, yet does not lower performance as horizontal attention has already gathered the information about past and future interactions.

**Classification Layer and Loss function**: Both Treeformer and Mformer are encoding layers which transform the raw input sequence into sequence of embedding per token using self attention mechanism. For each of the page types, we perform attention pooling on the embedding representations of all the tokens under the subtree, which is then passed through a feed-forward layer to generate the final scalar output. As mentioned earlier, we train the model with pairwise rank loss where the ordering is determined by the daily, weekly conversion rate. Let $y_i^d$ and $y_i^w$ respectively denote daily and weekly conversion rates for customer $i$. Then for any customer pair (i,j) such that the daily conversion $y_j^d > y_i^d$, the model minimizes the loss $L_d = ReLU(\hat{y}_i^d \ - \ \hat{y}_j^d \ + \alpha)$ of scoring customer $i$ ahead of customer $j$ with $\alpha$ used to impose a minimum margin in separating the interaction score. Similarly, we define the weekly conversion loss $Lw$ based on weekly conversion rates $y_i^w$. The final loss is a combination of daily and weekly loss given by $L = \beta L_d + (1 - \beta)L_w$.

## V. Experiments

In this section, we report the findings from experimental evaluation of the proposed approach on a private proprietary dataset.

**Dataset**: We consider all the customer interactions in two month window and partition the data into label window and training window. The former is used to generate the labels, namely conversions rates, and the latter is used in training the ML model. For constructing the browse tree, we consider the most recent 400 activities in the training duration and eliminate all customer sessions with less than ten interactions. For each customer who visited the site in the training duration, we compute the daily and weekly conversion in the label window. The conversion is computed as the ratio of total add to carts to number of days (or weeks) visited in the label window. The data of customers who did not visit the site in the label window is discarded from training data. The customer browse tree, represented by six sequence features and the daily/weekly conversion rates are used to train models to minimize the loss $L$. We trained the Mformer model with $K = 4$ Mformer encoder layers with embedding dimension $d$ set to $64$ for all tokens. Similarly, the Treeformer was trained with $K = 4$ Treeformer encoding layers, but the embedding dimension of each token in Treeformer varied with the feature type. We set the trade-off parameter $\beta$ in the loss function $L$ to $0.4$, giving slightly more weight to weekly conversion, as it was found to be less noisy than daily conversion.

**Evaluation Metrics:** The model performance can be evaluated on two parameters:- 1) effectiveness in optimizing the rank loss and 2) understanding the impact of user experience (UX) on various segments of customers defined by the model generated browse score. The former is measured on offline data (test set performance) and the latter can be inferred from A/B test. In order to test the model's ability in rank ordering sessions (with known customers), such that customers with higher conversion get higher score, we measure the correlation of the model score with conversion metric of customers. Table II presents the correlation between the *CSI* scores produced by Treeformer and Mformer models. As seen, the Treeformer based encoding outperforms that of Mformer based encoding for both daily and weekly conversion rate. This shows that representing the site interaction as trees induced by the underlying DOM structure help us capture good patterns in data that indicate conversion than representing the data as simple multivariate sequence data. Next, we present results highlighting the usefulness of *cis score* in identifying cohorts of session that find a given UX valuable or overwhelming.

**Cart page interaction score**: To understand the usefulness of the proposed *CSI* score, we analyzed the clicklogs of sessions that satisfied following conditions on cart page: 1) clicked on at least one widget 2) had sitewide interactions for more than 7 days with at least 10 clicks. We computed the *cis score* for these sessions and partitioned into two bins - low scoring and high scoring sessions. Table III captures the difference in sales between these two session cohorts. As expected, session with higher cis score generated 41bps increase in sales and sessions with lower interaction score resulted in decreasing sales, reflecting that *cis scores* reflect conversion by only looking at sitewide interactions.

Given that the model takes only the browse tree as input, the segments are purely a function of customer interaction and no demographic, tenure or purchase data was used in segmenting the sessions. These factors help us comprehend the usefulness of the *cis scores* in identifying the right segments (customer/session) that respond positively or negatively to a given user experience (UX). As a next step, we will be working on integrating these scores as an input into ML optimized page layout system to personalize each page in the site. A key drawback of the score is that it cannot be computed in real-time and that we need to observe the session for a few days to generate a reasonable score. As a result, using the layout optimizing model, we need to use stale scores to render the page. In the future, we will also work towards computing this score as real time as possible.

TABLE II

This table shows the correlation between the model generated scores and daily and weekly conversions on test set.

| Page Type | Treeformer | | Mformer | |
|---|---|---|---|---|
| | Daily | Weekly | Daily | Weekly |
| Product | 0.42 | 0.47 | 0.37 | 0.34 |
| Cart | 0.27 | 0.32 | 0.14 | 0.28 |
| Search | 0.38 | 0.46 | 0.22 | 0.31 |
| Browse | 0.36 | 0.44 | 0.11 | 0.20 |

TABLE III

This table presents the results of live test of effectiveness of *cis scores* on browse sessions. Simply applying a threshold on *cis* score, we could partition sessions to those that generated positive sales from sessions that generated negative sales.

| *cis* | # Sessions | Revenue (CR) | Units |
|---|---|---|---|
| Low | 37515699 | -21 bps | -11 bps |
| High | 8208190 | 41 bps | 11 bps |

## VI. Conclusion

In this paper, we have presented a novel approach to quantify customer friction in browsing web pages through a sequence of real-time clicks made by the customer during a session. Our proposed Treeformer architecture effectively processes multi-variate time series data organized in a tree-like structure using transformer attention in a compute and memory efficient way. We leverage conversion metrics to induce a rank ordering among customers based on their ease of browsing to generate a *CSI* score that can effectively identify customer segments with positive/negative impacts in A/B tests for better user targeting. The generated signal can be utilized as input to the auto-ranking engine, providing explicit feedback on the customer's ease of browsing. In summary, our work highlights the potential of our proposed methodology to enhance web page optimization systems and drive better business outcomes.

## References

[1] Deepak Agarwal et al. "Constrained Optimization for Homepage Relevance". In: WWW '15 Companion. New York, NY, USA: Association for Computing Machinery, 2015, pp. 375–384. URL: https://doi.org/10.1145/2740908.2745398.

[2] Qingpeng Cai et al. "Reinforcement Mechanism Design for E-Commerce". In: *Proceedings of the 2018 World Wide Web Conference*. WWW '18. International World Wide Web Conferences Steering Committee, 2018, pp. 1339–1348. ISBN: 9781450356398.

[3] Soumen Chakrabarti. "Integrating the document object model with hyperlinks for enhanced topic distillation and information extraction". In: *Proceedings of the 10th international conference on World Wide Web*. 2001, pp. 211–220.

[4] Chun-Fu Chen, Rameswar Panda, and Quanfu Fan. "RegionViT: Regional-to-Local Attention for Vision Transformers". In: *ArXiv* abs/2106.02689 (2021).

[5] Lu Chen et al. "WebSRC: A Dataset for Web-Based Structural Reading Comprehension". In: *Conference on Empirical Methods in Natural Language Processing*. 2021.

[6] Paul Covington, Jay Adams, and Emre Sargin. "Deep Neural Networks for YouTube Recommendations". In: *Proceedings of the 10th ACM Conference on Recommender Systems*. RecSys '16. Association for Computing Machinery, 2016, pp. 191–198.

[7] Xiang Deng et al. "DOM-LM: Learning Generalizable Representations for HTML Documents". In: *ArXiv* abs/2201.10608 (2022).

[8] Weicong Ding, Dinesh Govindaraj, and S. V. N. Vishwanathan. "Whole Page Optimization with Global Constraints". In: *Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining* (2019).

[9] Thore Graepel et al. "Web-Scale Bayesian Click-Through rate Prediction for Sponsored Search Advertising in Microsoft's Bing Search Engine". In: *International Conference on Machine Learning*. 2010.

[10] Jake Grigsby, Zhe Wang, and Yanjun Qi. "Long-Range Transformers for Dynamic Spatiotemporal Forecasting". In: *ArXiv* abs/2109.12218 (2021).

[11] Qiang Hao et al. "From one tree to a forest: a unified solution for structured web data extraction". In: *Proceedings of the 34th international ACM SIGIR conference on Research and development in Information Retrieval* (2011).

[12] Michael Hardy. "Pareto's Law". In: *Math. Intell.* 32 (Sept. 2010), pp. 38–43. DOI: 10.1007/s00283-010-9159-2.

[13] Daniel N Hill et al. “An efficient bandit algorithm for realtime multivariate optimization”. In: *Proceedings of the 23rd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. 2017, pp. 1813–1821.

[14] Daniel N. Hill et al. “An Efficient Bandit Algorithm for Realtime Multivariate Optimization”. In: *Proceedings of the 23rd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining* (2017).

[15] Luo Jie et al. “A unified search federation system based on online user feedback”. In: *Proceedings of the 19th ACM SIGKDD international conference on Knowledge discovery and data mining* (2013).

[16] Seyed Mehran Kazemi et al. “Time2Vec: Learning a Vector Representation of Time”. In: *ArXiv* abs/1907.05321 (2019).

[17] Jianan Li et al. “Layoutgan: Generating graphic layouts with wireframe discriminators”. In: *arXiv preprint arXiv:1901.06767* (2019).

[18] Lihong Li et al. “A Contextual-Bandit Approach to Personalized News Article Recommendation”. In: *Proceedings of the 19th International Conference on World Wide Web*. WWW ’10. 2010, pp. 661–670.

[19] Linghui Li et al. “Image Caption with Global-Local Attention”. In: *AAAI Conference on Artificial Intelligence*. 2017.

[20] Bill Yuchen Lin et al. “FreeDOM: A Transferable Neural Architecture for Structured Information Extraction on Web Documents”. In: *Proceedings of the 26th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining* (2020).

[21] Chieh Lo et al. “Page-Level Optimization of e-Commerce Item Recommendations”. In: *Proceedings of the 15th ACM Conference on Recommender Systems*. RecSys ’21. 2021, pp. 495–504.

[22] Thang Luong, Hieu Pham, and Christopher D. Manning. “Effective Approaches to Attention-based Neural Machine Translation”. In: *ArXiv* abs/1508.04025 (2015).

[23] Aditya Mantha et al. “A Real-Time Whole Page Personalization Framework for E-Commerce”. In: *CoRR* abs/2012.04681 (2020).

[24] Joe Marini. *Document object model*. McGraw-Hill, Inc., 2002.

[25] J González Penalver and JJ Merelo. “Optimizing web page layout using an annealed genetic algorithm as client-side script”. In: *Parallel Problem Solving from Nature—PPSN V: 5th International Conference Amsterdam, The Netherlands September 27–30, 1998 Proceedings 5*. Springer. 1998, pp. 1018–1027.

[26] Yifan Peng et al. “Branchformer: Parallel MLP-Attention Architectures to Capture Local and Global Context for Speech Recognition and Understanding”. In: *ArXiv* abs/2207.02971 (2022).

[27] Ashok Kumar Ponnuswami et al. “On composition of a federated web search result page: using online users to provide pairwise preference for heterogeneous verticals”. In: *Web Search and Data Mining*. 2011.

[28] Zhou Qin and Wenyang Liu. “Automate page layout optimization: An offline deep Q-learning approach”. In: *RecSys 2022*. 2022. URL: https://www.amazon.science/publications/automate-page-layout-optimization-an-offline-deep-q-learning-approach.

[29] Ashish Vaswani et al. “Attention is All you Need”. In: *NIPS*. 2017.

[30] Qifan Wang et al. “WebFormer: The Web-page Transformer for Structure Information Extraction”. In: *Proceedings of the ACM Web Conference 2022* (2022).

[31] Yue Wang et al. “Beyond Ranking: Optimizing Whole-Page Presentation”. In: *Proceedings of the Ninth ACM International Conference on Web Search and Data Mining*. WSDM ’16. 2016, pp. 103–112.

[32] Yue Wang et al. “Optimizing Whole-Page Presentation for Web Search”. In: *ACM Trans. Web* 12.3 (2018).

[33] Mark Wilhelm et al. “Practical Diversified Recommendations on YouTube with Determinantal Point Processes”. In: *Proceedings of the 27th ACM International Conference on Information and Knowledge Management*. CIKM ’18. Torino, Italy: Association for Computing Machinery, 2018, pp. 2165–2173. ISBN: 9781450360142. DOI: 10.1145/3269206.3272018. URL: https://doi.org/10.1145/3269206.3272018.

[34] Lauren Wood et al. “Document object model (dom) level 1 specification”. In: *W3C recommendation* 1 (1998).

[35] Chuhan Wu et al. “Fastformer: Additive Attention is All You Need”. In: 2021.

[36] Xiangyu Zhao et al. “Deep Reinforcement Learning for Page-Wise Recommendations”. In: *Proceedings of the 12th ACM Conference on Recommender Systems*. RecSys ’18. Vancouver, British Columbia, Canada: Association for Computing Machinery, 2018, pp. 95–103. ISBN: 9781450359016.

[37] Yichao Zhou et al. “Simplified DOM Trees for Transferable Attribute Extraction from the Web”. In: *ArXiv* abs/2101.02415 (2021).