Real-Time Personalization with Simple Transformers

Lin An, Andrew A. Li, Vaisnavi Nemala, and Gabriel Visotsky

Tepper School of Business, Carnegie Mellon University, Pittsburgh, Pennsylvania 15213

linan, aali1, vnemala, gvisotsk@andrew.cmu.edu

Real-time personalization has advanced significantly in recent years, with platforms utilizing machine learning models to predict user preferences based on rich behavioral data on each individual user. Traditional approaches usually rely on embedding-based machine learning models to capture user preferences, and then reduce the final real-time optimization task to one of *nearest-neighbors*, which can be performed extremely fast both theoretically and practically. However, these models struggle to capture some complex user behaviors, such as sequence effects, complementarity, or variety effects, which are essential for making accurate recommendations. Transformer-based models, on the other hand, are known for their practical ability to model sequential behaviors, and hence have been intensively used in personalization recently to overcome these limitations. However, optimizing recommendations under transformer-based models is challenging due to their complicated architectures. In this paper, we address this challenge by considering a specific class of transformers, showing its ability to represent complex user preferences, and developing efficient algorithms for real-time personalization.

We focus on a particular set of transformers, called *simple transformers*, that contain a single self-attention layer. We show that simple transformers are capable of capturing complex user preferences, such as variety effects, complementarity and substitution effects, and various choice models, which traditional embedding-based models cannot capture. We then develop an algorithm that enables fast optimization of real-time personalization tasks based on simple transformers. Our algorithm achieves near-optimal performance and has sub-linear runtime. Finally, we demonstrate the effectiveness of our approach through an empirical study on large datasets from Spotify and Trivago. Our experiment results show that (1) given data on past user behavior, simple transformers can model/predict user preferences substantially more accurately than non-transformer models and nearly as accurately as more-complex transformers, and (2) our algorithm completes simple-transformer-based recommendation tasks quickly and effectively. Comparing against two natural benchmark algorithms, our algorithm on average achieves objective values 4.5% higher than Beam Search and 6.5% higher than k-Nearest Neighbor.

1. INTRODUCTION

Personalization today is already immensely sophisticated. Media platforms, online retailers, and subscription services (just to name a few) capture rich data on their users in the form of their behavior and interactions with individual items/products. There are then two key ingredients: (1) this data is used *offline* to build machine-learning (ML) based models of users' preferences, and then (2) these models are used in *real-time* to make personalized recommendations.

Zooming out from personalization for just a moment, the most jarring improvements in ML models over the last few years have been in generative models for language, and specifically *transformer*-based models (e.g. the "T" in *ChatGPT*) that have proven to be extremely accurate in modeling *sequential* data. Perhaps unsurprisingly, these same models are well-equipped for personalization. To fix a concrete example, suppose an *Instacart* user is in the process of shopping online for groceries. This user's behavior consists of interactions with grocery items: browsing through items, viewing a subset of these in more detail, and adding a subset of these to their shopping cart. The task of learning this user's preferences essentially amounts to predicting their future interactions. The important observation here is that the user's behavior is naturally sequential, and so this prediction task is similar to completing a sentence, where the "words" are the items themselves. This connection to language suggests that the same transformer-based models may succeed in learning preferences.

This is already being done in practice, often with substantial empirical success (e.g. by *Alibaba* [2], *Amazon* [5], *Spotify* [7], and *Wayfair* [6]). However, as examples of such successes become increasingly common, there is little principled guidance on how transformers "should" be used for real-time personalization. This is the problem we seek to address.

Real-Time Personalization, Before Transformers: To make the nature of this problem more precise, it is worth reviewing how real-time personalization is performed *without* transformers. Referring to the two key ingredients mentioned at the outset: first, the ML-based models of user preferences are, by and large, pure *embedding*-based models. Using past data, each item i is mapped to some element $v_i \in \mathbb{R}^d$ in such a way that (a) "similar" items are "close" together, and perhaps more formally, (b) each user can be represented as some $u \in \mathbb{R}^d$ so that the inner products $u^\top v_i$ fully represent the preference/affinity of the user for each item i.

These pure embedding models are not necessarily the most *accurate* models that can be estimated from past data, but

they enable *fast* execution of the second ingredient, which is to optimize a set (or sequence) of items for each user in real time:

$$\begin{aligned} \max \quad & f(S,u) \\ \text{s.t.} \quad & S \subset [n], |S| = k. \end{aligned} \tag{1}$$

The (extremely general) formulation above simply highlights that real-time personalization consists of solving cardinality-constrained *set-optimization* (or *sequence-optimization*, whose equivalence to set-optimization we will discuss later on) problems where (a) the objective function is user-dependent, (b) the number of items n is potentially quite large – often in the hundreds of millions – and (c) a solution must be found in real-time, often just milliseconds. This is of course hopeless in general, but feasible when the objective function $f(S,u)$ results from a pure embedding models. In particular, $f(S,u)$ typically takes one of two forms:

1. An additive function:

$$f(S,u) = \sum_{i \in S} g(u^\top v_i),$$

for some non-decreasing function $g(\cdot)$. In this case, the optimal set S^* consists of the k items whose inner products with u are largest.

2. A monotone submodular function (in the argument S, for any u), such that each item is fully encoded by $u^\top v_i$, so that a constant-factor approximation can be found using greedy-style algorithms along with (multiple queries to) a black box which computes the item with largest inner product to a given point in $\mathbb{R}^d$ ([3]).

In both of the above cases, the pure embedding model essentially reduces the final real-time optimization task to one of *nearest-neighbors*, which can be performed extremely fast, both theoretically (using approximate nearest neighbor algorithms with runtime sub-linear in n) and practically (given the nonstop engineering and improvement of commercial vector databases).

An Attempt to Introduce Transformers: Returning to transformers now, the natural opportunity is to improve the accuracy of the pure embedding models in representing user preferences: the embedding models essentially fail to capture the effects that items may have on each other when present in the same recommended set. Such effects are well-known to exist in multiple fields of study, as we will discuss shortly, and often representable via transformers. Unfortunately, the just-described synergy between the "upstream" user preference model estimated from data, and the "downstream" optimization of user-dependent sets of items completely breaks down here. As we will see in the next section, this is true in a formal sense: Problem (1) is NP-hard, and likely hard to even approximate in linear time, if the objective $f(\cdot,u)$ is transformer-based. As of now, any practical implementation using transformers either applies a pure nearest-neighbor or greedy-style algorithm (essentially ignoring this hardness), or a random search heuristic (such as *beam search*).

In the spirit of developing a principled approach to transformer-based real-time personalization, this raises two major questions:

1. **Fast Optimization:** The formal hardness results just alluded to imply that achieving fast (ideally sub-linear in n) optimization of Problem (1) with any meaningful optimality guarantee is impossible if the objective $f(\cdot,u)$ is to allow for *all* transformers. Naturally then, is there a non-trivial sub-class of transformers for which this is possible?

2. **Modeling Power:** Assuming a positive answer to the first question, i.e. assuming the existence of a subset of transformers which enable fast optimization, does restricting to this subset come at a substantial cost in terms of modeling user preferences? Put another way, can this smaller sub-class of transformers achieve the same predictive accuracy as the family of all transformers?

2. OUR CONTRIBUTIONS

In short, our contributions provide concrete, theoretically-backed answers to both of the above questions:

1. **Modeling User Preferences with Simple Transformers:** We focus our study on a sub-class of transformers that we refer to as *Simple Transformers*. These are transformers which contain a single *self-attention* layer (to be defined in the next section), whereas transformers as a whole may contain multiple attention layers. Addressing the pair of questions above in reverse, we first formally show that simple transformers are able to represent three known, popular parametric models of user preference/choice:

   - Sequential *variety* effects in the context of marketing
   - Pairwise *complementarity* and *substitution* effects in the context of economics
   - The *Halo Multinomial Logit* choice model, a generalization of the classic multinomial logit model from economics and operations management

   It should also be emphasized that none of these models are representable via pure embedding models.

2. **Real-Time Personalization with Simple Transformers:** Our main result is an algorithm which approximately solves Problem (1) in sub-linear time, when the objective function is given by a simple transformer:

   Theorem 1 (Informal)**.** *Under additional (rank) assumptions on the simple transformer, given any $k \in \mathbb{N}$ and $\epsilon > 0$, there exists an algorithm that achieves* $\text{ALG} > (1-\epsilon)\text{OPT}$ *in expectation, with amortized runtime $\tilde{O}\left(n^{1-c(\epsilon,k)}\right)$, where $c(\epsilon,k) > 0$. Here $\tilde{O}$ hides factors of order $n^{o(1)}$.*

   Our algorithm operates under the same two-phase *retrieve* and *rank* paradigm that is used in many competition-winning personalization algorithms, though in our case both phases are adapted specifically to simple transformers, and enjoy provable guarantees (the combination of which generates our main result). Our algorithm has the added practical benefit of subsuming

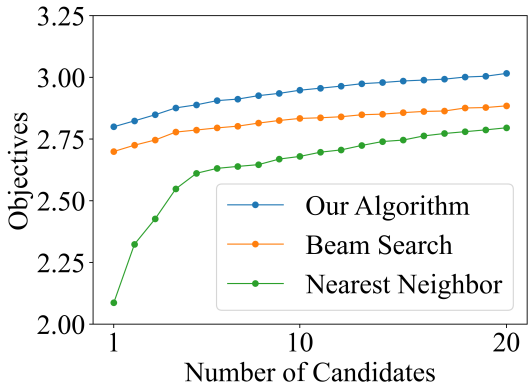
(given a particular, sub-optimal selection of tuning parameters) the *beam search* algorithm commonly used in practice.

3. **Empirical Study:**

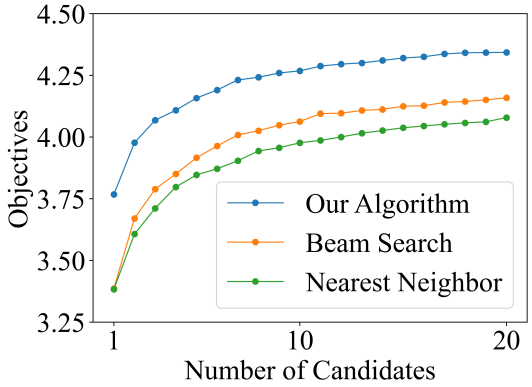
   We empirically validated the theoretical results of the previous contributions on two large datasets from *Spotify* [1] (which includes $1,000,000$ playlists with $2,262,292$ unique songs) and the travel website *Trivago* [4] (which includes user sessions of searching for hotel bookings, with around $730,000$ unique users and around $340,000$ unique hotels recorded in around $900,000$ different sessions).

   In support of the first contribution, our first set of experiments demonstrates that, given data on past user behavior, simple transformers can model/predict user preferences both (a) substantially more accurately than non-transformer models (such as pure embedding models), and (b) nearly as accurately as more-complex transformers. Specifically, simple transformers on average achieved 14.1% higher accuracy than non-attention models (e.g. logistic regression, random forest, support vector machine), and only 2.5% lower accuracy than more-complex transformers.

   In support of the second contribution, our second set of experiments demonstrates that our algorithm completes simple-transformer-based recommendation tasks quickly and effectively. We solved instances of Problem (1) using the simple transformers from the first set of experiments, and compared our algorithm to two common benchmark algorithms: k-Nearest Neighbor and Beam Search. Given a fixed budget on candidate solutions (to partially standardize run time), our algorithm on average achieved objective values 4.5% higher than Beam Search and 6.5% higher than k-Nearest Neighbor.



(a) Spotify



(b) Trivago

Figure 1: Performances of three algorithms. The x-axis is the number of candidate solutions generated by each algorithm, and the y-axis is the objective value of the current best candidate solution. Each figure is averaged across 100 instances.

3. REFERENCES

[1] C.-W. Chen, P. Lamere, M. Schedl, and H. Zamani. Recsys challenge 2018: Automatic music playlist continuation. In *Proceedings of the 12th ACM Conference on Recommender Systems*, pages 527–528, 2018.

[2] Q. Chen, H. Zhao, W. Li, P. Huang, and W. Ou. Behavior sequence transformer for e-commerce recommendation in alibaba. In *Proceedings of the 1st international workshop on deep learning practice for high-dimensional sparse data*, pages 1–4, 2019.

[3] V. F. Farias, A. A. Li, and D. Sinha. Optimizing offer sets in sub-linear time. In *Proceedings of the 21st ACM Conference on Economics and Computation*, pages 639–640, 2020.

[4] P. Knees, Y. Deldjoo, F. Bakhshandegan Moghaddam, J. Adamczak, G.-P. Leyson, and P. Monreal. Recsys challenge 2019: Session-based hotel recommendations. In *Proceedings of the Thirteenth ACM Conference on Recommender Systems*, RecSys '19, New York, NY, USA, 2019. ACM.

[5] T. Lake, S. A. Williamson, A. T. Hawk, C. C. Johnson, and B. P. Wing. Large-scale collaborative filtering with product embeddings. *arXiv preprint arXiv:1901.04321*, 2019.

[6] M. J. Mei, C. Zuber, and Y. Khazaeni. A lightweight transformer for next-item product recommendation. In *Proceedings of the 16th ACM Conference on Recommender Systems*, pages 546–549, 2022.

[7] D. Moor, Y. Yuan, R. Mehrotra, Z. Dai, and M. Lalmas. Exploiting sequential music preferences via optimisation-based sequencing. In *Proceedings of the 32nd ACM International Conference on Information and Knowledge Management*, pages 4759–4765, 2023.