Editable User Profiles for Controllable Text Recommendations

Sheshera Mysore
University of Massachusetts Amherst
United States
smysore@cs.umass.edu

Mahmood Jasim
University of Massachusetts Amherst
United States
mjasim@cs.umass.edu

Andrew McCallum
University of Massachusetts Amherst
United States
mccallum@cs.umass.edu

Hamed Zamani
University of Massachusetts Amherst
United States
zamani@cs.umass.edu

ABSTRACT

Methods for making high-quality recommendations often rely on learning latent representations from interaction data. These methods, while performant, do not provide ready mechanisms for users to control the recommendation they receive. Our work tackles this problem by proposing LACE, a novel *concept value bottleneck model* for controllable text recommendations. LACE represents each user with a succinct set of human-readable concepts through retrieval given user-interacted documents and learns personalized representations of the concepts based on user documents. This concept based user profile is then leveraged to make recommendations. The design of our model affords control over the recommendations through a number of intuitive interactions with a *transparent user profile*. We first establish the quality of recommendations obtained from LACE in an offline evaluation on three recommendation tasks spanning six datasets in warm-start, cold-start, and zero-shot setups. Next, we validate the controllability of LACE under simulated user interactions. Finally, we implement LACE in an interactive controllable recommender system and conduct a user study to demonstrate that users are able to improve the quality of recommendations they receive through interactions with an editable user profile.

CCS CONCEPTS

• **Information systems** → **Personalization**.

KEYWORDS

interactive recommendation systems; text recommendations; concept bottleneck models; pre-trained language models

ACM Reference Format:
Sheshera Mysore, Mahmood Jasim, Andrew McCallum, and Hamed Zamani. 2023. Editable User Profiles for Controllable Text Recommendations. In *Proceedings of the 46th International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR '23), July 23–27, 2023, Taipei, Taiwan.* ACM, New York, NY, USA, 11 pages. https://doi.org/10.1145/3539618.3591677

SIGIR '23, July 23–27, 2023, Taipei, Taiwan

ACM ISBN 978-1-4503-9408-6/23/07…$15.00
https://doi.org/10.1145/3539618.3591677

1 INTRODUCTION

Recommendation systems play a ubiquitous role in influencing the information we consume and the decisions we make. Despite this, these systems fall short of allowing sufficient control to users [18] or any transparency to system aspects [36]. And effective recommenders involve learning opaque user profiles and item representations from user interaction data [74]. The value of control in recommendation has been emphasized by prior work demonstrating greater user satisfaction [30], improved trust in the system [28], and an intention to continue consuming content [71]. However, prior work also notes that while users value control, they often prefer hybrid strategies combining automatic methods with interactive strategies for preference elicitation and control, indicating there to be a sweet spot between automation and control [31, 33, 71].

Given the importance of this tradeoff, we develop a performant recommender that facilitates control over recommendations through an editable user profile. For our model, we lay the following goals: (1) to facilitate interactive control, user profiles should be human-readable i.e. transparent, (2) users should be able to edit the profile to express their preferences in various intuitive ways with the recommender system interactively updating its recommendations after profile edits, and (3) the recommender system should make performant recommendations – ensuring that controllability does not degrade the recommendation quality. Some recent work is relevant to these goals [7, 52]. Balog et al. [7] construct transparent user profiles as a set of weighted tags, subsequently used for scrutable recommendation. Despite desirable aspects, their fully transparent approach presents drawbacks in relying on pre-tagged items, not leveraging item content beyond determination of item tags, and remaining inapplicable in the absence of interaction data. On the other hand, Radlinski et al. [52] make a case for natural language user profiles that may be used to prompt large-language models (LLM) for few-shot scrutable recommendations - while representing an exciting prospect the scrutability of LLMs approaches remains unknown [13].

In this paper, we introduce a fundamentally different approach for controllable recommendations where our model formulation ensures controllability, effective use of item content, and its use of pre-trained LMs allows effective performance in challenging zero and cold-start scenarios. The starting point for our approach is provided by Concept Bottleneck Models [35] developed for controllable prediction. Our approach, LACE[1], builds each user profile as a small set of readable concepts retrieved from a large inventory of

[1] Retrieval Enhanced Concept Value Bottleneck Model

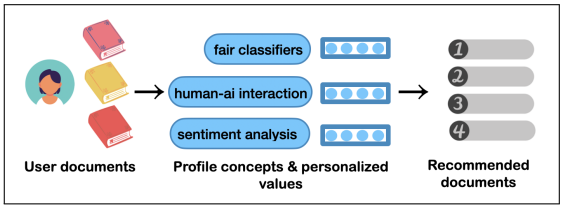

Figure 1: Our proposed approach, LACE represents users with human readable concept profiles and uses these for controllable recommendations. LACE presents two key novelties: a retrieval-enhanced concept profile allowing users to edit the profile and personalized concept values computed from user documents for performant recommendations.

concepts given user interacted documents. Next, to effectively use user documents, it computes a *personalized concept value* as a function of user documents and the profile concepts. These personalized concept values are computed through a sparse matching of user content to profile concepts computed with an Optimal Transport procedure [51]. These are then used for computing recommendations. LACE admits edits such as positive or negative selections to specify preferences on profile concepts or textual edits to the concepts, which then change the personalized concept values and the recommendations.

We evaluate several aspects of LACE in a series of extensive experiments. We conduct offline evaluations on six real-world datasets spanning three recommendation tasks: scientific paper recommendation, TED Talk recommendation, and paper-reviewer matching for peer review. We validate the efficacy of LACE for generating effective recommendations in three evaluation settings: warm-start, cold-start, and zero-shot. Next, we validate the ability of our model to demonstrate control under *simulated user interactions*. Finally, we implemented LACE in an interactive system and conducted a *user study* to evaluate its interaction ability in a realistic usage scenario. We find LACE to outperform several reasonable baselines in offline evaluations and interactively allow users to make significant improvements to their recommendations.

Therefore, our contributions include: 1) Proposing a performant model for controllable recommendations, 2) Demonstrating effective empirical performance in numerous evaluation settings, and 3) Establishing the effectiveness of LACE in a realistic user study. Our code is online: https://github.com/iesl/editable_user_profiles-lace

2 PROBLEM FORMULATION

Desiderata. In building a controllable recommender we assume access to users $u \in \mathcal{U}$, where each user u has interacted with a set of items $D_u = \{d_i\}_{i=1}^{|D_u|}$. Each d_i represents a text document of natural language sentences. To generate recommendations for a user u, a ranking system f must generate a ranking R_u over candidate documents $\mathcal{D}$. In building an editable user profile, we aim to build a human-readable representation, $\mathcal{P}_u$ of user interactions D_u. This is subsequently used for making recommendations as $R_u = f(\mathcal{P}_u, \mathcal{D})$. A user may control R_u by manipulating $\mathcal{P}_u$. This editable user profile $\mathcal{P}_u$ must fulfill the following desiderata:

D1: Communicate interests to the user. Users must be able to understand their profile to edit it and control their recommendations. Thus, the profile $\mathcal{P}_u$ should communicate their interests as captured by D_u. Importantly, we only aim to have a transparent user profile to facilitate interactive *control* over recommendations. This does not necessitate a "white-box" or transparent model [37] – therefore, we don't pursue this goal.

D2: Control recommendations via profile edits. The profile $\mathcal{P}_u$ should provide edit operations to the user, which are then reflected in the recommended results $R' = g(\mathcal{P}'_u, \mathcal{D})$. The profile should broadly support positive and negative preference specifications for interests and correction of errors represented in $\mathcal{P}_u$. Further, to enable users to develop a mental model for control over recommendations, the system must allow fast inference with updated recommendations serving as feedback for user actions [15, 58]. Our goal of control draws on prior work illustrating its benefits [27, 71].

D3: Performant recommendations. Finally, the profile $\mathcal{P}_u$ should allow for high-quality recommendations before and after profile edits. This follows from users' desire for a sweet spot between automation and control over recommendations [33, 71].

Profile Design. In this work, we choose to represent $\mathcal{P}_u$ as a set of natural language concepts describing a user's interests. This design follows from a common choice in prior work [5, 24, 34] and findings suggesting that users often find concepts/keyphrases to be intuitive descriptors of groups of items [7, 11]. Specifically, given user documents D_u, and a inventory of concepts $\mathcal{K}$, a profile construction model g must induce a user profile of P concepts, $\mathcal{P}_u = \{k_1, \ldots k_P\}$ to describe D_u, where $k \in \mathcal{K}$. In our work, interactions include positive or negative selection of concepts in $\mathcal{P}_u$ to indicate interest or disinterest in them or edit actions like adding, removing, or renaming concepts to account for variations or errors in $\mathcal{P}_u$.

3 PROPOSED APPROACH

The problem of building a controllable recommender that represents users with human-readable concepts and uses these for making controllable recommendations may be viewed as an instance of a *concept bottleneck model* (CBM) [35, 44]. CBMs are neural network models representing input data x with human-readable concepts k and then using these to predict targets: $x \rightarrow k \rightarrow y$. The concepts allow examination of the model and interventions on y.

CBMs involve learning functions $g : x \rightarrow k$ and $f : k \rightarrow y$ from input data paired with concepts and targets: (x, k, y). However, learning a model of this type presents some challenges: 1) Paired user profile data of the form $(D_u, \mathcal{P}_u)$ for training g is often hard to obtain. 2) Since we aim to allow edit actions such as addition, deletion, or renaming of concepts in $\mathcal{P}_u$, g must support these actions and allow interactions to influence downstream predictions. 3) Given that strong text recommendations [8, 50] rely on rich user document features, models g and f should leverage neural network features of D_u to generate recommendations.

Our proposed approach (Figure 1), LACE, represents a CBM with two components: *profile construction*: $g : D_u \rightarrow \mathcal{P}_u$, and *ranking* $f : \mathcal{P}_u \rightarrow R_u$. To tackle the challenges outlined above, we present two key novelties in profile construction: i) A retrieval enhanced concept bottleneck: We formulate g as a *retrieval function*, retrieving concepts from a global concept inventory $\mathcal{K}$ to construct a profile

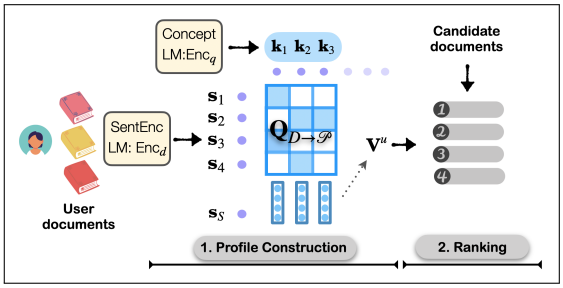

Figure 2: Profile concepts in LACE, obtained using retrieval, serve as *keys* used to compute *personalized concept values* from user documents; these are used for ranking.

$\mathcal{P}_u$ with pre-trained LM encoders. This formulation allows users to make edits to an induced concept bottleneck, with encoders used for concept retrieval also used to encode user edits. Further, use of pre-trained LM encoders allows the construction of $\mathcal{P}_u$ without labeled data $(D_u, \mathcal{P}_u)$. ii) Concept personalization: To leverage features of user documents in representing the profile concepts, each concept is represented with a personalized concept value, $\mathbf{V}_i$ computed as a function of the concept and user documents. Since the personalized concept value is a function of the concept, any user edits to the concept also update the personalized concept value. For this, we leverage Optimal Transport (OT), a method for computing assignments and distances between sets of vectors [51]. Here OT is used to make a sparse assignment of user documents to profile concepts. The assigned document content is then used to compute the personalized concept values. The concepts and their personalized values may also be viewed as keys and values, resulting in a *concept-value bottleneck* model. These personalized concept values $\mathbf{V}^u = \{\mathbf{V_i}\}_{i=1}^{P}$ represent a multi-vector user representation that is used for ranking. For ranking, we follow recent work [48] and represent candidate documents as multi-vectors $\mathbf{S}^d$ computed from their sentences and generate recommendations by using OT once again to compute distances between sets of user and document vectors: $\mathbf{V}^u$ and $\mathbf{S}^d$. Next, we briefly review Optimal Transport.

3.1 Background on Optimal Transport

The optimal transport problem may be seen as a way to compute a minimum cost alignment between sets of points given the pairwise distances between them. Specifically, given the set of points, $\mathbf{S}_p \in \mathbb{R}^{m\times d}$ and $\mathbf{S}_{p'} \in \mathbb{R}^{n\times d}$, and distributions $\mathbf{x}_p$ and $\mathbf{x}_{p'}$ according to which the set of points is distributed. The OT problem involves computation of a soft assignment, the *transport plan* $\mathbf{Q}$, which converts $\mathbf{x}_p$ into $\mathbf{x}_{p'}$ by transporting probability mass from $\mathbf{x}_p$ to $\mathbf{x}_{p'}$ while minimizing an aggregate cost $\mathcal{W}$ of moving mass between the points. $\mathcal{W}$ is computed from pairwise costs $\mathbf{C}$. Further, $\mathbf{Q}$ is constrained such that its columns and rows marginalize respectively to $\mathbf{x}_p$ and $\mathbf{x}_{p'}$. Therefore, computation of $\mathbf{Q}$ takes the form of a constrained linear optimization problem:

$$\mathcal{W} = \min_{\mathbf{Q}'\in\mathcal{S}} \langle \mathbf{C}, \mathbf{Q}' \rangle \tag{1}$$

Where $\mathcal{W}$ refers to the Wasserstein or Earth Movers Distance and $\mathbf{Q}$ minimizes Eq (1), and the feasible set $\mathcal{S} = \{\mathbf{Q}' \in \mathbb{R}_+^{m\times n} | \mathbf{Q}'\mathbf{1}_n = \mathbf{x}_p, \mathbf{Q}'^T\mathbf{1}_m = \mathbf{x}_{p'}\}$. In our work costs $\mathbf{C}$ are pairwise L2 distances between $\mathbf{S}_p$ and $\mathbf{S}_{p'}$. For computing the solution above, Cuturi [16] introduced an entropy regularized variant of Eq (1). This can be used with end-to-end differentiation and allows GPU computation. We leverage this formulation in our work. We do this at two stages of our model: in profile construction, we use the OT plan $\mathbf{Q}$, and for ranking, we use the minimum cost Wasserstein distance $\mathcal{W}$. In our implementations, we use the `geomloss` package for solving OT problems with entropy regularizer $\lambda = 20$ and uniform $\mathbf{x}_p, \mathbf{x}_{p'}$

3.2 Model Description

Retrieval enhanced concept bottleneck. Given the user documents D (we drop user subscripts for brevity), we leverage pre-trained language model encoders to retrieve concepts from a concept inventory $\mathcal{K}$ to describe the user documents. This design allows interaction from users who may add concepts not present in the user profile or $\mathcal{K}$, and rename (remove-then-add) existing concepts in $\mathcal{P}$ (Figure 3). Therefore the function, $g : D \to \mathcal{P}$ is factored into a document and concept encoders: $\texttt{Enc}_d$ and $\texttt{Enc}_q$. Specifically, we represent user documents $D = \{s_i\}_{i=1}^{S}$ with sentences due to their ability to capture finer-grained information in documents [48]. Sentence vectors $\mathbf{S}_D$ and concept vectors $\mathbf{K}$, are obtained from $\texttt{Enc}_d$ and $\texttt{Enc}_q$. To construct the profile concepts $\mathcal{P} = \{k_1 \dots k_P\}$ the $\texttt{top-}P$ concepts are retrieved as follows:

$$\mathcal{P} = \texttt{top-}P(\mathbf{S}_D, \mathbf{K}) \tag{2}$$

$$\texttt{dist}(\mathbf{S}_D, \mathbf{k}_i) = \texttt{min}_j \|\mathbf{k}_i - \mathbf{s}_j\| \tag{3}$$

The distance for individual concepts $\mathbf{k}_i$ is computed as the minimum L2 distance to the sentences. Further, $\mathbf{K} \in \mathbb{R}^{|\mathcal{K}|\times E}$ and $\mathbf{S}_D \in \mathbb{R}^{S\times E}$, where E represents the embedding dimension. This set of retrieved concepts for a user is revised during training as $\mathbf{S}_D$ and $\mathbf{K}$ are updated. In practice, $\mathcal{K}$ may consist of a large number of concepts, so to update $\mathcal{P}$ during training a smaller set of pre-fetched concepts $\mathcal{K}_f$ are used for construction of $\mathcal{P}$ in Eq (2).

Personalized concept values. Now given a users profile concepts $\mathcal{P}$ and their embeddings from $\texttt{Enc}_q$ as $\mathbf{K}_\mathcal{P}$. The representations of the same concepts across different users will be identical. However, stronger personalization performance can be achieved if user content influences concept representations. To achieve this, we pair each profile concept with a *personalized concept value*: $\mathbf{V}_i$. Specifically, given $\mathbf{K}_\mathcal{P}$ and sentence vectors $\mathbf{S}_D$ we compute a soft matching $\mathbf{Q}_{D\to\mathcal{P}}$ of sentences in D to profile concepts $\mathcal{P}$. We leverage the optimal transport procedure (§3.1) to compute this matching. Computation of $\mathbf{V}_i$ involves computing an assignment weighted average of sentences:

$$\mathbf{V}_{i=\{1\dots P\}} = \frac{1}{\sum_{j=1}^{S} \mathbf{Q}_{ji}} \sum_{j=1}^{S} \mathbf{Q}_{ji} \cdot \mathbf{S}_j \tag{4}$$

Here we drop subscripts for $\mathbf{Q}$ and $S > P$. Note that computation of $\mathbf{Q}$ involves the use of profile concept vectors $\mathbf{K}_\mathcal{P}$ and sentence vectors $\mathbf{S}_D$ through the pairwise cost $\mathbf{C}$ in Eq (1).

These values $\mathbf{V}^u$ represent concept semantics grounded in user content allowing strong personalization performance. Further, OT computes sparse assignments $\mathbf{Q}$ [61], ensuring that sentences are only assigned to a small number of relevant concepts. Therefore, the concepts partition the sentences into soft clusters described by their concept. This enables users to specify positive or negative

preferences for specific concepts, which includes or excludes topical clusters of sentences in generating recommendations. Further, user edits to the text of concepts influence their embeddings, which in turn influence Q, $\mathbf{V}^u$, and R_u - allowing edits to reflect in recommendations. In this sense, the profile concepts may be seen as *keys* paired with personalized concept *values*. Finally, since $\mathbf{Q}$ remains differentiable, it allows gradient descent based training.

Candidate Ranking. Both user and candidate document representations, $\mathbf{V}^u \in \mathbb{R}^{P \times E}$ and $\mathbf{S}^d \in \mathbb{R}^{L \times E}$ are multi-vector representations. To compute a score w_{ud}, to rank the documents $d \in C$, we seek to align user interests with document content optimally. Further, we expect only a subset of user interests to match the document. Therefore, we score $\mathbf{S}^d$ against the top $|\mathcal{T}|$ elements of $\mathbf{V}^u$, i.e $\mathbf{V}^u_{\mathcal{T}} = \mathbf{V}^u[\mathcal{T}, :]$ where $|\mathcal{T}| < P$. $\mathcal{T}$ is obtained according to the minimum L2 distances of $\mathbf{V}^u[i, :]$ and $\mathbf{S}^d$. To compute the distance between multi-vector representations of the user and document, we leverage optimal transport once more [48]. Having computed pairwise costs $\mathbf{C}_w$ between $\mathbf{V}^u_{\mathcal{T}}$ and $\mathbf{S}^d$ and a minimum cost alignment $\mathbf{Q}_w$, the distance $w_{ud} = \langle \mathbf{C}_w, \mathbf{Q}_\mathbf{w} \rangle$, is used for ranking.

Document Encoder. Our approach leverages sentence representation of user documents D_u and candidate items $d \in \mathcal{D}$. Our work uses pre-trained transformer language model encoders for $\texttt{Enc}_d$. Given document d, we obtain contextual sentence representations $\mathbf{s}_i \in \mathbf{S}_d$ by averaging contextualized word-piece embeddings from $\texttt{Enc}_d$. In experiments, we use Sentence-Bert [54] for web text and the Aspire for scientific text [48]. Both models are strong 110M parameter BERT models pre-trained on document-similarity tasks.[2]

Concept Encoder. Control over LACE is achieved through profile concepts. These concepts are encoded with $\texttt{Enc}_q$. This requires $\texttt{Enc}_q$ to capture the semantics of concepts to ensure intuitive interaction with users. Further, vectors obtained from $\texttt{Enc}_q$ must be aligned with those obtained from the sentence encoders $\texttt{Enc}_d$. In this sense, $\texttt{Enc}_q$ may be considered a "query encoder". For web-text datasets, $\texttt{Enc}_q$ uses Sentence-Bert, making it identical to $\texttt{Enc}_d$. For scientific datasets, we pre-train a concept encoder using the contrastive Inverse Cloze Task (ICT) objective of Lee et al. [39] given the lack of performant short text encoders for scientific text. For ICT pre-training, we use 1M concept-document context pairs. Concepts were extracted using the unsupervised concept extraction method of King et al. [32] from the S2ORC corpus [42].

3.3 Training and Inference

Training. Training our model involves fine-tuning the parameters of the sentence and concept encoders: $\texttt{Enc}_d$, $\texttt{Enc}_q$. The primary objective, $\mathcal{L}_{rec}$, updates the parameters of $\texttt{Enc}_d$ from recommendation interactions. Specifically, given a users documents $D_u = \{d_i\}_{i=1}^{|D_u|}$, we treat each document d_i in turn, as a positive document d^+ for obtaining candidate sentence vectors $\mathbf{S}^{d^+}$ with the other documents $D^i_u = D_u \setminus \{d_{ui}\}$ used for computing profile values $\mathbf{V}^{u^+}$. Giving us the loss: $\mathcal{L}_{rec} = \sum_{u \in \mathcal{U}} \sum_{i=1}^{|D_u|} \max[w^i_{ud^+} - w^i_{ud^-} + \delta, 0]$. Here, w^i_{uc} denotes the distance between the user profile and candidate document. Negative document d^- is randomly sampled from the interactions of a different user u', and margin $\delta = 1$. Second, we continue to train $\texttt{Enc}_q$ on an ICT objective [39] used to pre-train this encoder to ensure that $\texttt{Enc}_q$ remains updated as $\texttt{Enc}_d$ is trained. Not updating $\texttt{Enc}_q$ with $\mathcal{L}_{rec}$ follows from the intuition that this encoder presents a way for users to interact with the model via edits. Therefore, it only captures the semantics of concepts and concept-sentence matches. Consequently, $\texttt{Enc}_q$ is continually trained on the ICT loss for the user documents D_u using a random half of pre-fetched concepts $\mathcal{K}_f$. Note that though we fine-tune our model, our semi-parametric model with pre-trained encoders allows zero-shot prediction (see §4). Finally, our ICT objective may be seen as a search objective with LACE trained as a joint search and recommendation model [73]. However, we focus on the recommendation task here.

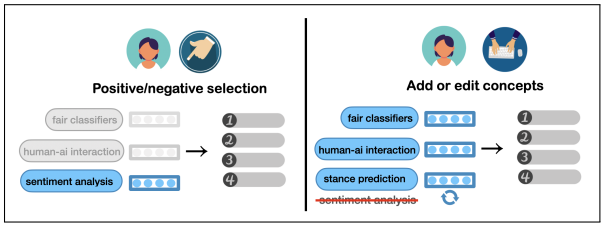

Figure 3: Users may interact with their concept-based profiles through positive/negative selection of concepts or edit concepts directly through addition, deletion, or renaming.

Recommendations. Making recommendations with LACE involves computing ranked documents R_u from candidate documents $\mathcal{D}$. Since our approach relies on a set of dense vectors to represent users and candidate documents with $\mathbf{V}^u$ and $\mathbf{S}^d$, these can be computed as per §3.2 and cached. Ranking involves computation of the Wasserstein distance w_{ud} – recent work has explored approximate nearest neighbor (ANN) methods for Wasserstein distances [4]. This paves the way for interactive large-scale recommendations with LACE - an important element of our desiderata (§2, D2). While we indicate the potential of ANN, we leave exploration of this to future work. We opt for a simpler approach - computing w_{ud} for all candidates $\mathcal{D}$ or opting to use LACE as a re-ranker where fast and performant first-stage rankers are available.

Interactions. Our model allows two forms of interaction through the profile concepts $\mathcal{P}_u$. Users may specify a positive or negative preference for profile concepts or edit the concepts through additions, deletions, and renaming to account for variations and errors (Figure 3). These are achieved as follows. 1. Positive/negative selection. After construction of profile concepts $\mathcal{P}_u$ and the corresponding values $\mathbf{V}^u$, users may positively or negatively select elements of $\mathbf{V}^u$ to generate recommendations. E.g., for a user with "sentiment analysis" in their profile, positive selection is akin to saying: "I want only sentiment analysis" and a negative selection: 'I don't want sentiment analysis". These amount to the recommendations computed from a topical subset of D_u. Positive selection results in the positively selected values, $\mathbf{V}^u[p, :]$ being used for computing R_u. Similarly, negative selection results in a compliment of the selections $\mathbf{V}^u[\overline{n}, :]$ being used for computing R_u. 2. Profile edits. Users may also directly change the text of concepts in $\mathcal{P}_u$ triggering recomputation of $\mathbf{V}^u$ i.e a reorganization of documents in D_u. Profile edits may span two types: deletion, addition, or modification of concepts *consistent* with the interests represented in D_u, and addition

[2]HF Transformers: sentence-transformers/bert-base-nli-mean-tokens, allenai/aspire-contextualsentence-multim-compsci

of new interests not captured in D_u. While adding new interests is an important interaction [5], it represents a problem more similar to search, different from recommendation. More generally, we expect profile interactions to take many forms and to be informed by subsequent model results, much like query reformulations [12, 29]. A full understanding of this requires thoroughly examining user behaviors [25]. However, our user study (§5) explores the efficacy of these interactions for improving recommendations.

4 RECOMMENDATION EVALUATION

We conduct an offline evaluation of LACE on three recommendation tasks: scientific paper recommendation, TED talk recommendation, and scientific reviewer-paper matching for peer review. This presents a task where papers suitable and of interest to reviewers must be recommended for review [59].

4.1 Experimental Setup

Datasets. For paper recommendation, we use the two public datasets CiteULike-A (Sparsity: 99.78%) and CiteULike-T (Sparsity: 99.93%) [63, 64]. Here, user's past history D_u consists of scientific papers (title and abstract) that the user saved in their personal "libraries" on the CiteULike platform, collected between 2004-2013. For TED talk recommendation (Sparsity: 99.70%) [50], we use a public dataset of users and their saved talks (title and description) which form D_u – referred to as TEDREC in §4.2. For reviewer-paper matching, we obtain three private datasets of reviewer-paper assignments from the ICLR 2019, ICLR 2020, and UAI 2019 conferences in collaboration with the OpenReview peer-review platform. Here, users represent expert reviewers, their authored papers (title and abstracts) represent D_u, and to-be-reviewed papers represent candidate documents $\mathcal{D}$. The relevance of candidate documents is captured in two ways: bids and assignments. Bids are made by reviewers on a paper to express interest in reviewing the paper, and assignments are made by conference organizers indicating the suitability to review a paper.

Implementation Details. Next, we describe important dataset-specific and model details and include other hyperparameters in our code release. The concepts in $\mathcal{K}$ used for user profiles must be able to describe documents and be interpretable to users - we use different inventories per dataset. For the TED Talk dataset (web-text), we use the inventory of categories in the dataset, $|\mathcal{K}| = 200$. For Openreview datasets (scientific text), we use user-contributed concepts in the dataset, $|\mathcal{K}| = 8000$. For the CiteULike datasets (scientific text), we extract scientific concepts using the unsupervised method of King et al. [32] from a corpus of 2.1 million computer science and biomedical papers in the S2ORC corpus [42] giving $|\mathcal{K}| = 116$k. Next, as the encoders in our model are updated during training, we update $\mathcal{P}_u$ from a pre-fetched set of concepts $\mathcal{K}_f$ by retaining 50% of these for constructing $\mathcal{P}_u$ (P of §3.2). This results in a variable length profile per user. To build $\mathcal{K}_f$ we retrieve a single concept per sentence from $\mathcal{K}$ for the documents in D_u. For web text, we build $\mathcal{K}_f$ with a Sentence-Bert model. For scientific text, we use TFIDF followed by reranking with our pre-trained encoders and retain the top concept. To compute profile-document distances for ranking, we use 20% of the user profile (i.e. $|\mathcal{T}|$ of §3.2). The choice of $\mathcal{K}_f$ and $\mathcal{T}$ were made from development set performance.

Evaluation Setup. We evaluate models in warm-start, cold-start, and zero-shot recommendation setups. We evaluate the paper and TED talk recommendation tasks in all three setups and paper reviewer matching in the zero-shot setup since it represents the natural application setup. Specifically, (1) *Warm-start*: Evaluates a method's ability to recommend items seen in training. For every user, we randomly sample 20% of a user's items and treat these as test items to be retrieved, treating the remaining items *per user* as a training set. (2) *Cold-start*: Evaluates a method's ability to recommend items unseen in training – a persistent challenge in recommender systems. Here, a test set is created by randomly sampling 20% of *all* items in the dataset and ensuring that these are unseen for any user. Models are trained on the remaining 80% of the data. (3) *Zero-shot*: This setup is identical to that of cold-start recommendation but is one where no training on interaction data is permitted and represents a task setup explored in more recent work [60]. This is apt in recommendation applications with privacy commitments against use of interaction data as in reviewer-paper matching. More generally reviewer-paper matching presents a natural cold-start setup where candidate items are unobserved during training. It also represents a significant domain shift where authored papers are likely to follow a different distribution than to-be-reviewed papers requiring zero-shot generalization at inference time. Model development was performed on a randomly sampled development set of 10% of users. Following prior work, CiteULike items are chosen such that they are saved by at least 5 users [8, 63], TEDREC excluded users with fewer than 12 interactions [50], and OpenReview used all items. We report performance in NDCG@20 and recall@20 in the interest of space, noting that result trends hold at rank 5.

Baselines. In the following experiments, we aim to benchmark the recommendation performance of LACE against various classes of established, scalable, and well-performing baselines spanning matrix factorization, content-based, and hybrid models – often found to outperform more complex architectures [19]. Note, however, that all the baselines cannot be applied in all three evaluation setups. Popular: A non-personalized baseline recommending items by popularity among users. BPR: Bayesian Personalized Ranking represents a strong matrix factorization method [55]. ALS: Alternating Least Squares regression represents a matrix factorization method allowing positive interactions to be weighted over negative ones. Popular, BPR, and ALS only work for our warm-start setup. Hybrid: This method of Bansal et al. [8] presents a strong hybrid recommendation method where users are represented by learned latent vectors, and items are represented with a neural network encoder - intended for cold start recommendation. We train this approach with dataset-specific pre-trained transformer language models. We freeze the LM encoder during training. It can only work for our warm- and cold-start setups. NeuKNN: This represents a transformer LM-based item nearest neighbor method making a recommendation by ranking candidates $\mathcal{D}$ based on the minimum L2 distance to documents in user interactions D_u. For training, we fine-tune these models for pairwise user document similarity. Zero-shot performance relies only on pre-trained parameters. NeuKNN presents a controllable method for making recommendations by excluding items in user interactions – in our user study of §5.2, this serves as a baseline. For Hybrid and NeuKNN models, we use specter [14] for scientific text datasets and Sentence-Bert [54] for

Table 1: Offline evaluation for warm-start setups.

	CiteULike-A		CiteULike-T		TEDREC	
Warm-start	NDCG@20	R@20	NDCG@20	R@20	NDCG@20	R@20
[1]Popular	0.64	1.25	1.68	3.04	7.34	12.10
[2]ALS	16.82	25.55	16.58	28.58	6.49	11.07
[3]BPR	12.90	20.34	13.49	24.14	4.80	8.57
[4]Hybrid$_{SP/SB}$	10.37	17.46	6.22	12.05	1.58	2.55
[5]NeuKNN$_{SP/SB}$	5.79	17.45	5.95	15.19	1.57	5.22
LACE	8.86	20.20$^{\times 3}$	8.36	19.27	3.69	7.78
LACE $_{rrALS}$	**24.86**	**44.92**	**19.74**	**35.63**	**12.50**	**27.27**

Table 2: Offline evaluation for cold-start and zero-shot setups.

	CiteULike-A		CiteULike-T		TEDREC	
Cold-start	NDCG@20	R@20	NDCG@20	R@20	NDCG@20	R@20
[1]Hybrid$_{SP}$	22.93	31.64	13.61	23.45	7.32	11.51
[2]NeuKNN$_{SP}$	26.31	37.36	18.38	28.75	17.49	25.69
LACE	**29.39**	**39.72**	**21.20**	**32.11**	**17.93**$^{\times 2}$	**27.06**$^{\times 2}$
Zero-shot						
[1]NeuKNN$_{SP}$	21.25	30.46	15.21	24.07	16.30	22.77
LACE	**27.47**	**38.86**	**18.76**	**29.15**	**17.41**	**24.86**

TED talk recommendation (Hybrid$_{SP/SB}$, NeuKNN$_{SP/SB}$). Finally, in warm-start, we use LACE to re-rank the top 100 results for an established approach, ALS – denoted LACE $_{rrALS}$.

4.2 Experimental Results

Tables 1, 2, and 3 present our empirical results across datasets and evaluation setups. Here, metrics are in percentage, bold indicates the best metric, and statistical significance of LACE models was measured against all baselines with two-sided t-tests at $p < 0.05$ with Bonferroni corrections. Superscripts ($^{\times\#}$) indicate *non* significant results in Table 1, 2, and 3. Unmarked results are significant.

Warm-start. In Table 1, we first examine the performance of the non-personalized baseline, Popular. In both CiteULike-A/T, it underperforms all other approaches. In TEDREC, however, it sees strong performance – indicating a strong bias for popular talks in viewer preferences. Next, given the sparsity of these datasets, matrix factorization approaches ALS and BPR show strong performance compared to approaches leveraging content: Hybrid and NeuKNN. This gap is larger in the more sparse CiteULike-T, matching prior understanding [20]. ALS and BPR also leverage the popularity signal more effectively in TEDREC. Next, while LACE underperforms non-controllable matrix factorization approaches, we see it matches or outperforms approaches leveraging item content. This may be attributed to LACE learning *personalized* representations of user content through its profile values and aggregating the strength of multiple user items in computing user-candidate item similarity. Finally, LACE as a re-ranker for ALS, LACE $_{rrALS}$, outperforms all other approaches with large percent improvements over best baselines: 47-75% in CiteULike-A, 19-25% in CiteULike-T, and 70-125% in TEDREC. This strategy leverages the benefits of matrix factorization approaches while retaining the benefits of controllability and a content-based model for cold-start and zero-shot recommendation presented by LACE. Re-ranking also presents a path to scaling LACE and adopting it into existing matrix factorization systems.

Table 3: Offline evaluation on the Openreview platform in a zero-shot setup – the realistic setup for this task.

	ICLR-2020		ICLR-2019		UAI-2019	
Bids	NDCG@20	R@20	NDCG@20	R@20	NDCG@20	R@20
[1]NeuKNN$_{SP}$	15.52	13.22	15.21	11.91	22.98	22.76
LACE	**18.02**	**15.74**	**18.33**	**14.81**	**28.05**	**28.17**
Assignments						
[1]NeuKNN$_{SP}$	7.52	12.85	6.51	12.58	17.21	28.16
LACE	**9.13**	**15.31**	**11.88**	**17.72**	**20.09**$^{\times 1}$	**33.13**

Cold-start. Table 2 presents results in a setup where test set items are never seen during training. This precludes comparison to Popular, ALS, and BPR. Here, across datasets, a content-based approach NeuKNN sees stronger performance than a Hybrid method indicating the value of content-based representations in the cold-start setup. In the more dense TEDREC dataset, NeuKNN$_{SB}$ sees stronger performance with LACE matching or slightly outperforming it. This matches prior understanding with item nearest neighbor methods seeing a stronger performance in denser datasets [23]. In the sparser CiteULike-A/T datasets, we see LACE improve upon the best baselines by 5-11% and 11-15% respectively. This indicates the potential of LACE for application in the challenging cold-start setup while providing the benefits of interactive control.

Zero-shot. Tables 2 and 3 present results in the challenging zero-shot setup where recommendations must be made without any training data – when presented only with user items D_u. Here we only compare against NeuKNN$_{SP/SB}$ a model which can be applied only with its pre-trained weights. In Table 2, we see trends similar to the cold-start setup. In CiteULike-A/T, we see gains of 27-29% and 21-23% respectively and 7-9% in TEDREC. Next, consider Table 3, presenting results on reviewer-paper matching datasets of OpenReview with two measures of relevance: Bids and Assignments. Across datasets, we observe that LACE improves on NeuKNN$_{SP}$ by 16-24% on bids and 17-82% on assignments. These datasets also represent a difference in distribution between user items (authored papers) and candidate items (to-be-reviewed papers) – the strong performance of LACE also indicates its robustness to these shifts. Finally, NeuKNN$_{SP}$ forms part of the system used for reviewer-paper matching on OpenReview. This performance of LACE also indicates its potential for adoption in the important peer-review application of reviewer-paper matching.

Ablation Study. In table 4 we ablate elements of LACE and important baselines to demonstrate the design trade-offs involved. We report performance in zero-shot and cold-start setups on two datasets, given the ability of our models to be applied in these settings and the interest of space.

Minus CV bottleneck. Recall that LACE represents documents with sentences and computes personalized concept values, which aggregate sentences for computing user-item scores. A lack of this concept-value bottleneck results in a model which uses sentence vectors of user documents (S_D of §3.2) directly for computing user-item scores. In CiteULike-A, we see LACE outperform this model (I) in the zero-shot setup and show smaller gains with training as in cold-start. This indicates the benefit provided by the inductive bias of the concept-value bottleneck, which may be overcome with training data. In TEDREC, we see similar performance for both models,

Table 4: Ablations indicating trade-offs in model design.

		Cold-start		Zero-shot	
	CiteULike-A	NDCG@20	R@20	NDCG@20	R@20
	LACE	29.39	39.72	27.47	38.86
	NeuKNN$_{SP}$	26.31	37.36	21.25	30.46
I	LACE−CV bottleneck	28.10	38.21	24.30	35.22
II	LACE−concept values	12.66	19.85	03.90	06.63
III	NeuKNN+more pretraining	26.28	37.39	24.89	35.38
	TEDREC				
	LACE	17.93	27.06	17.41	24.86
	NeuKNN$_{SB}$	17.49	25.69	16.30	22.77
I	LACE−CV bottleneck	17.78	27.02	17.85	24.79
II	LACE−concept values	09.85	16.18	08.80	14.53
III	NeuKNN+more pretraining	17.45	25.89	16.09	21.40

indicating the value of learning dataset-specific patterns. Recall, however, that this simplified model (I) does not offer controllability beyond that offered by NeuKNN.

Minus concept values. Next, we consider a model with a concept-based profile as in LACE but not using the personalized concept values of LACE. Instead, this uses embeddings of profile concepts $\mathcal{P}_u$ to compute user-item scores. This approach offers controllability similar to LACE. However, we see (II) this approach consistently underperforms LACE indicating the value provided by the personalized concept values of LACE. Note that this may also be seen as a tag-based recommender as in Balog et al. [7] – showing personalized concept values to be a more effective use of interacted documents beyond determination of a tags alone.

More pretraining. Finally, to examine the benefits of extensive text similarity pre-training for ItemKNN models, we examine the performance of NeuKNN variants with more extensive pre-training.[3]. For CiteULike-A, we see (III) improved performance in the zero-shot setup compared to NeuKNN$_{SP}$ with gains disappearing upon training as seen in cold-start. In TEDREC, we see more pre-training not to benefit performance, indicating the importance of learning dataset-specific patterns. Broadly, we also see patterns similar to Table 2 and performance comparable or lower than LACE.

5 INTERACTION EVALUATION

To evaluate the efficacy of LACE to control recommendations, we first validate some of the interactions provided by LACE through a series of simulations (§5.1). Then we conduct a task-oriented lab evaluation of users interacting with LACE and use the resulting data to evaluate the interactive aspects of LACE (§5.2).

5.1 Simulation Evaluation

Before our lab evaluation, we use simulations to evaluate simpler interactions in LACE: validating the positive and negative selection (§3.3) of individual concepts to perform as expected and synonymous edits (e.g., replacing "passage retrieval" with "document retrieval") to individual profile concepts not to cause large changes in the recommendations received. To measure the efficacy of positive and negative selection, we measure Concept Recall@20 (CR@20) - testing for the presence of a concept in the recommended documents compared to all documents before and after a tuning interaction - a metric also used in prior work [49]. An effective model should increase CR@20 for positive selection and decrease it for negative selection. We measure robustness through Recall@20 with interactions data, expecting a robust model not to cause large changes after a synonymous edit. Given its cleaner text, we conduct our simulations with user data in CiteULike-A. We begin by selecting the 14 most frequent concepts in user profiles[4], and randomly select 20 users with the concept in their profile. The resulting 280 concept user pairs are used in simulations. For each concept-user pair, we positive/negative select the personalized concept value for the concept in the user's profile, followed by generating recommendations. To simulate synonymous edits, we use a separate Sentence-Bert [54] encoder to encode and generate nearest neighbors for the 14 frequent concepts and select one of the top 20 concepts as a replacement for a user's profile. Table 5 depicts the results of our simulations. Positive and negative selection cause CR@20 to increase and decrease significantly (two-sided t-test, p=0.05). We also see synonymous edits not to cause significant changes to R@20. These indicate the effectiveness of simple interactions with LACE.

Table 5: Simulation evaluation of LACE for positive/negative selection and robustness to synonymous profile edits.

		Positive	Negative	Robustness
		CR@20 ↑	CR@20 ↓	R@20 =
LACE	[1]Initial	36.67	36.67	42.23
	Tuned	49.75[1]	25.94[1]	42.14

5.2 Task Oriented User-Study

Next, we conduct a task-oriented within-subjects lab evaluation of LACE with 20 users consisting of computer science researchers. Here, users interacted with two models LACE (system name, Maple) and NeuKNN (system name, Otter) to receive research paper recommendations, saved the papers they found interesting, and tuned the recommendations to their liking through interactions with the systems. Through the data collected, we aim to answer the following research questions: LACE Controllability, RQ1: Does LACE allow users to improve the recommendations they receive? LACE vs. NeuKNN, RQ2: Does LACE allow users to improve their recommendations more effectively than NeuKNN? Besides answering these RQs, we also report realistic usage patterns with the data gathered in our user study.

System Description. To conduct our study, we developed two interactive recommendation systems for making scientific paper recommendations - Otter and Maple. For both systems, candidate documents $\mathcal{D}$ consisted of 100k computer science papers from the S2ORC corpus [42]. Of this, 50k were the most highly cited computer science papers to ensure familiarity, and the other half was sampled randomly. Choosing popular items is also common in prior work [7]. Otter used a NeuKNN model and served as our baseline system. Maple used LACE. NeuKNN used a SOTA document bi-encoder model for document similarity [48] - we denote it as NeuKNN$_{ASP}$. LACE was implemented similarly to Section 3;

[3]HF Transformers, CiteULike-A: allenai/aspire-biencoder-compsci-spec, TEDREC: sentence-transformers/all-mpnet-base-v2

[4]Selected to ensure familiarity to the author running the semi-automatic simulation.

however, a Sentence-Bert model was used to retrieve profile concepts and optimal transport was implemented using the POT library [21]. Our system used a re-ranking strategy for LACE. Retrieving 500 documents from $\mathcal{D}$ for each $d \in D_u$ using NeuKNN$_{\text{Asp}}$ and re-ranking these documents with LACE. Both systems ran on 2 CPUs and 16 GB RAM. They used identical interfaces, only varying in their recommendation tuning methods.

Study Participants. Participants for our study were recruited through university mailing lists and announcements made on social media. To ensure that users had formed research interests and could identify papers of interest to them, we asked respondents to confirm that they were involved in a research project and briefly describe their research in our sign-up form. Of the respondents, 20 were selected as participants/users for our study on a first-come-first-serve basis while ensuring they were involved in research. Participants received $25 gift cards for hour-long participation. All study procedures were approved by the university IRB. As a proxy for their expertise, participants noted authorship for research papers: 1-5 research papers (14/20), none (5/20), and 6-10 (1/20).

User Study Description. Our within-subjects study consisted of three main phases, (A) preference elicitation, (B) evaluation of an initial list of recommendations, and finally, (C) multiple iterations of edits to a user profile followed by an examination of the recommended list to improve the initial recommendations. While (A) was performed offline through a web form, (B) and (C) were performed in a 1-hour study session over Zoom.

Preference Elicitation. In our offline preference elicitation, users were instructed to submit 2 distinct sets of 4-5 papers or 2 authors of interest to any of their prior, current, or likely future research interests. The submitted information was used to construct a seed set of 20-25 papers representing user documents (D_u) per system. To build D_u, we expanded the user-submitted papers by randomly sampling the references cited in each submitted paper or by gathering the 20 most recent papers by the submitted author. These methods are common for discovering relevant papers [3]. Data was gathered through the Semantic Scholar API.

We avoided gathering extensive item ratings to keep a low burden on participants. Further, having participants submit papers of interest to their research ensured that they were experts in these areas and could evaluate recommendations relevant to their research interests. Additionally, a semi-automatic method for gathering D_u also ensured that while many papers in D_u were topically relevant to users, some were undesirable - necessitating tuning. Finally, our study also used distinct sets of seed papers D_u per system; this ensured that users remained engaged in examining recommendations from both systems. However, this meant that comparisons between both systems could only be made in aggregate across all users.

Study Procedure. Next, in an hour-long session conducted via Zoom, users used the Maple and Otter systems for 20-25 minutes each or until they decided to stop. They did not know the proposed vs baseline systems, and the order of system use was random to prevent fatigue or learning effects privileging a system.

In Maple, users first skimmed the seed papers D_u to familiarize themselves with their contents. Next, they examined the inferred profile concepts $\mathcal{P}_u$. Here, users removed concepts if they were redundant, nonsensical, or did not represent the seed papers or added concepts if there were aspects of the seed papers which were missing in $\mathcal{P}_u$. These were used to compute personalized concept values $\mathbf{V}^u$ and produce an initial list of 30 recommendations R_u^0. Users were told to examine the recommended papers and save the ones they wanted to read in more detail. To ground their interest and mirror preference elicitation, users were encouraged to consider papers relevant to prior, current, or future research interests. Following examination of R_u^0, users were free to find as many more interesting papers as they could by interacting with the profile, examining the recommendations, and saving the ones they found interesting. Users could make positive or negative selections from $\mathcal{P}_u$ to refine their recommendations by focusing on specific concepts or excluding some concepts, respectively. Alternatively, they could edit concepts in $\mathcal{P}_u$ if there were aspects of D_u they wanted to focus on or to accomplish other intents.

Table 6: User study evaluation of LACE compared against the baseline NeuKNN$_{\text{Asp}}$ for improving recommendations.

		MRR	NDCG@5	NDCG@20
NeuKNN$_{\text{Asp}}$	[1]Initial	66.97	48.30	69.16
	Tuned	85.38	67.54[1]	82.03[1]
	+ Gain	18.42	19.24	12.88
LACE	[a]Initial	70.76	50.93	71.68
	Tuned	90.00[a]	74.83[a]	86.16[a]
	+ Gain	19.24	23.90	14.48

The study procedure for Otter mirrored that of Maple. However, users did not make concept corrections in Otter (NeuKNN). Further, to refine their recommendations, users could only exclude papers in D_u and recompute their recommendations (i.e negative/positive selection) – the only interaction possible with NeuKNN. We gathered tuning actions (additions, deletions of concepts or seed items), recommended lists, and saved papers in both systems. In our subsequent analysis, user saves are treated as binary relevance measures.

User-study Results. Our user study allows us to answer RQ1 and RQ2, examining if LACE allows users to improve their recommendations and if it is more effective than a baseline NeuKNN. We answer both questions through ranking metrics: MRR, NDCG@5, NDCG@20 in Table 6. We omit recall metrics since they cannot be computed from our study, and superscripts indicate statistical significance at p=0.05 with a paired t-test. Note that, Table 6 shows higher metrics than §4.2 due to the fully judged recommendation lists obtained from users in the user study, unlike the incomplete relevance labels of implicit feedback datasets.

LACE Controllability. Comparing Initial vs. Tuned performance in Table 6, users saw statistically significant gains of 20-47% through interactions with LACE. Therefore we answer RQ1 in the affirmative - LACE is effectively able to improve the quality of recommendations users receive. We also note that NeuKNN$_{\text{Asp}}$ also saw gains of 18-40% from tuning. In using LACE and NeuKNN$_{\text{Asp}}$, users made 2.65 and 2.20 tuning iterations respectively. Both systems were used for the same duration or until users choose to stop.

LACE vs NeuKNN. To compare the two systems, we examine both models' Initial, Tuned, and Gain performance. In Table 6, we note that LACE outperforms NeuKNN$_{\text{Asp}}$ at the Initial and Tuned stages by 4-6% and 5-10%, respectively. We also note slightly larger Gained metrics for LACE. However, these were not statistically

significant. Therefore, our results indicate that LACE sees improved performance before and after tuning compared to NeuKNN$_{\text{Asp}}$. However, larger-scale studies with diversity in users, and larger candidate document sets are necessary to establish significance.

Characterizing Usage of LACE. Next, we characterize usage with LACE as implemented in Maple. Based on the changes users made to $\mathcal{P}_u$ for redundancy and correctness we found the initially inferred $\mathcal{P}_u$ to have a precision of 74.25% on average. Here, users deleted 5.15 concepts and added 1.25 concepts. Fewer additions indicate that the inferred profiles had sufficient coverage of D_u, but suffered from redundant or incorrect concepts. In tuning their recommendations, users made 7.2 positive/negative selections and 0.68 additions to their initially corrected profile, indicating a preference for selection operations over edits to $\mathcal{P}_u$.

6 RELATED WORK

Next, we discuss the rich body of prior work on which we build.

Interacting with Recommenders. A line of work has explored interfaces to control recommenders and their influence on users. This line of work has explored user profile-based interaction, with profiles constructed automatically [5, 24, 47] or from user input [53]. Here, automatic methods often construct profiles using supervised methods for keyphrase or entity extraction, enrich the extractions via linking to existing knowledge bases, and then, use them to compute recommendations. This line of work has also explored other forms of control, ranging from a selection between different algorithms, applying keyword filters to a generated list of recommendations [26, 31, 33], and changes to algorithms themselves [27]. Different from our work which contributes a performant interactive recommender this work has used existing methods and focused on studying the rich ways in which recommenders influence users.

Critiquable Recommenders. Critiquable recommenders allow control over recommendations in one of three ways, at present: 1) *one-time user feedback* on recommended item explanations followed by *retraining* whole or parts of a model [22, 38]. 2) *Conversational feedback* via item keyphrases or explanations with a latent user representation updated incrementally given a critique [1, 2, 40, 45, 67, 69]. 3) *Feedback via item attributes* which endow latent user or item dimensions with information of item attributes [49, 65]. This allows user feedback to influence user or item representations, which are then reflected in recommendations. Each of these bears differences to LACE. Our approach does not require retraining as in one-time user feedback methods. Next, while methods for one-time or conversational feedback via explanations control recommendations by interaction with *individual items*, LACE allows control over sets of items, a more intuitive and efficient structure for expressing preferences [7, 11]. Finally, current methods for conversational and item attribute feedback rely on observing keyphrase usage of users or item attributes for training, not allowing expansion at test time – LACE allows test expansion of keyphrases/attributes to user-specified keyphrases, offering greater flexibility.

Another notable aspect of the work in critiquable recommendations is their use of variational autoencoders to capture user preferences via a single latent vector and model components to update this vector from feedback [1, 46, 49, 65, 69]. The concept-value bottleneck of LACE models *multiple* user interests explicitly as human readable concepts which can be directly interacted with – not requiring modeling for aligning latent dimensions to human interpretable ones or updating them with feedback.

Richer User and Item Modeling. As in LACE, prior work in information filtering and recommendation has developed content-based recommenders [8, 43, 63] allowing performant recommendation in cold-start and zero-shot setups. Further, prior work has also modeled the multiple interests of users in collaborative filtering models via multiple *latent* prototypes[9, 66, 70]. Our work primarily differs from these in building interactive and transparent user profiles to control content-based recommenders.

User Profile Construction. A line of work has sought to build user profiles for a range of applications leveraging approaches in matrix factorization [10], learning to rank [56], and information extraction [41, 72]. This line of work often attempts to build general-purpose user profiles while leveraging labeled data such as tagging behavior of users [10], profile attribute values extracted from social networks [41], and user-assigned document tags [56, 72]. While this line of work leverages supervised data for constructing profiles we contribute a method for inducing user profiles in the absence of labeled data *and* influencing a downstream recommender.

7 CONCLUSION AND FUTURE WORK

Our paper introduces a novel retrieval-enhanced concept-value bottleneck model, LACE, for constructing a human editable user profile and making performant text recommendations. We demonstrate strong performance in 3 recommendation tasks and 6 datasets in offline evaluations. Then we validate the controllability of LACE through simulated edits and a task-oriented user study. We demonstrate that users can make significant improvements to their recommendations through interaction with LACE.

LACE presents several opportunities for future work. The concept-value user representation may be used for controllable personalization in other applications, e.g. search [62] and text generation [17, 57], perhaps by augmenting transformer language models with a rich and compact personalized memory [68]. Further, richer structured user data in the form of personal knowledge graphs may motivate more structured profile representations and accompanying interactive learning and inference algorithms [6]. Further, the intuitive profile edit interactions supported in LACE call for the design of interactive recommenders leveraging this strength. These may then be studied and evaluated in larger-scale online evaluation spanning impactful applications, such as peer-review and text recommendation – presenting benefits to end users and providing a rich canvas for future research spanning multiple communities.

ACKNOWLEDGMENTS

We thank anonymous reviewers for their invaluable feedback. This work was partly supported by the Center for Intelligent Information Retrieval, NSF grants IIS-1922090 and 2143434, the Office of Naval Research contract number N000142212688, and the Chan Zuckerberg Initiative under the project Scientific Knowledge Base Construction. Any opinions, findings and conclusions or recommendations expressed in this material are those of the authors and do not necessarily reflect those of the sponsors.

REFERENCES

[1] Diego Antognini and Boi Faltings. 2021. Fast Multi-Step Critiquing for VAE-Based Recommender Systems. In *Proceedings of the 15th ACM Conference on Recommender Systems* (Amsterdam, Netherlands) *(RecSys '21)*. Association for Computing Machinery, New York, NY, USA, 209–219. https://doi.org/10.1145/3460231.3474249

[2] Diego Antognini, Claudiu Musat, and Boi Faltings. 2021. Interacting with Explanations through Critiquing. In *Proceedings of the Thirtieth International Joint Conference on Artificial Intelligence, IJCAI-21*, Zhi-Hua Zhou (Ed.). International Joint Conferences on Artificial Intelligence Organization, 515–521. https://doi.org/10.24963/ijcai.2021/72 Main Track.

[3] Kumaripaba Athukorala, Eve Hoggan, Anu Lehtiö, Tuukka Ruotsalo, and Giulio Jacucci. 2013. Information-seeking behaviors of computer scientists: Challenges for electronic literature search tools. *Proceedings of the American Society for Information Science and Technology* 50, 1 (2013), 1–11. https://doi.org/10.1002/meet.14505001041 arXiv:https://asistdl.onlinelibrary.wiley.com/doi/pdf/10.1002/meet.14505001041

[4] Arturs Backurs, Yihe Dong, Piotr Indyk, Ilya Razenshteyn, and Tal Wagner. 2020. Scalable Nearest Neighbor Search for Optimal Transport. In *ICML*.

[5] Fedor Bakalov, Birgitta König-Ries, Andreas Nauerz, and Martin Welsch. 2010. IntrospectiveViews: An interface for scrutinizing semantic user models. In *UMAP*.

[6] Krisztian Balog and Tom Kenter. 2019. Personal Knowledge Graphs: A Research Agenda. In *Proceedings of the 2019 ACM SIGIR International Conference on Theory of Information Retrieval* (Santa Clara, CA, USA) *(ICTIR '19)*. Association for Computing Machinery, New York, NY, USA, 217–220. https://doi.org/10.1145/3341981.3344241

[7] Krisztian Balog, Filip Radlinski, and Shushan Arakelyan. 2019. Transparent, scrutable and explainable user models for personalized recommendation. In *SIGIR*.

[8] Trapit Bansal, David Belanger, and Andrew McCallum. 2016. Ask the GRU: Multi-Task Learning for Deep Text Recommendations. In *RecSys*.

[9] Oren Barkan, Roy Hirsch, Ori Katz, Avi Caciularu, and Noam Koenigstein. 2021. Anchor-Based Collaborative Filtering. In *CIKM*.

[10] Cheng Cao, Hancheng Ge, Haokai Lu, Xia Hu, and James Caverlee. 2017. What Are You Known For? Learning User Topical Profiles with Implicit and Explicit Footprints. In *SIGIR*.

[11] Shuo Chang, F. Maxwell Harper, and Loren Terveen. 2015. Using Groups of Items for Preference Elicitation in Recommender Systems. In *CSCW*.

[12] Jia Chen, Jiaxin Mao, Yiqun Liu, Fan Zhang, Min Zhang, and Shaoping Ma. 2021. Towards a Better Understanding of Query Reformulation Behavior in Web Search. In *Proceedings of the Web Conference 2021* (Ljubljana, Slovenia) *(WWW '21)*. Association for Computing Machinery, New York, NY, USA, 743–755. https://doi.org/10.1145/3442381.3450127

[13] Yanda Chen, Chen Zhao, Zhou Yu, Kathleen McKeown, and He He. 2022. On the Relation between Sensitivity and Accuracy in In-context Learning. https://doi.org/10.48550/ARXIV.2209.07661

[14] Arman Cohan, Sergey Feldman, Iz Beltagy, Doug Downey, and Daniel S Weld. 2020. Specter: Document-level representation learning using citation-informed transformers. In *ACL*.

[15] W. Bruce Croft. 2019. The Importance of Interaction for Information Retrieval. In *SIGIR*.

[16] Marco Cuturi. 2013. Sinkhorn distances: Lightspeed computation of optimal transport. *NIPS* (2013).

[17] Shiran Dudy, Steven Bedrick, and Bonnie Webber. 2021. Refocusing on Relevance: Personalization in NLG. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*. Association for Computational Linguistics, Online and Punta Cana, Dominican Republic, 5190–5202. https://doi.org/10.18653/v1/2021.emnlp-main.421

[18] Malin Eiband, Sarah Theres Völkel, Daniel Buschek, Sophia Cook, and Heinrich Hussmann. 2019. When People and Algorithms Meet: User-Reported Problems in Intelligent Everyday Applications. In *IUI*.

[19] Maurizio Ferrari Dacrema, Paolo Cremonesi, and Dietmar Jannach. 2019. Are We Really Making Much Progress? A Worrying Analysis of Recent Neural Recommendation Approaches. In *Proceedings of the 13th ACM Conference on Recommender Systems* (Copenhagen, Denmark) *(RecSys '19)*. Association for Computing Machinery, New York, NY, USA, 101–109. https://doi.org/10.1145/3298689.3347058

[20] Maurizio Ferrari Dacrema, Paolo Cremonesi, and Dietmar Jannach. 2019. Are We Really Making Much Progress? A Worrying Analysis of Recent Neural Recommendation Approaches. In *Proceedings of the 13th ACM Conference on Recommender Systems* (Copenhagen, Denmark) *(RecSys '19)*. Association for Computing Machinery, New York, NY, USA, 101–109. https://doi.org/10.1145/3298689.3347058

[21] Rémi Flamary, Nicolas Courty, Alexandre Gramfort, Mokhtar Z. Alaya, Aurélie Boisbunon, Stanislas Chambon, Laetitia Chapel, Adrien Corenflos, Kilian Fatras, Nemo Fournier, Léo Gautheron, Nathalie T.H. Gayraud, Hicham Janati, Alain Rakotomamonjy, Ievgen Redko, Antoine Rolet, Antony Schutz, Vivien Seguy, Danica J. Sutherland, Romain Tavenard, Alexander Tong, and Titouan Vayer. 2021. POT: Python Optimal Transport. *Journal of Machine Learning Research* 22, 78 (2021), 1–8. http://jmlr.org/papers/v22/20-451.html

[22] Azin Ghazimatin, Soumajit Pramanik, Rishiraj Saha Roy, and Gerhard Weikum. 2021. ELIXIR: Learning from User Feedback on Explanations To Improve Recommender Models. In *Web Conference*.

[23] Miha Grčar, Dunja Mladenič, Blaž Fortuna, and Marko Grobelnik. 2005. Data sparsity issues in the collaborative filtering framework. In *International workshop on knowledge discovery on the web*.

[24] M Guesmi, M Chatti, Y Sun, S Zumor, F Ji, A Muslim, L Vorgerd, and SA Joarder. 2021. Open, scrutable and explainable interest models for transparent recommendation. In *IUI Workshops*.

[25] Manish Gupta, Rui Li, Zhijun Yin, and Jiawei Han. 2010. Survey on Social Tagging Techniques. *SIGKDD Explor. Newsl.* (2010).

[26] Jaron Harambam, Dimitrios Bountouridis, Mykola Makhortykh, and Joris van Hoboken. 2019. Designing for the Better by Taking Users into Account: A Qualitative Evaluation of User Control Mechanisms in (News) Recommender Systems. In *RecSys*.

[27] F Maxwell Harper, Funing Xu, Harmanpreet Kaur, Kyle Condiff, Shuo Chang, and Loren Terveen. 2015. Putting users in control of their recommendations. In *RecSys*.

[28] Dietmar Jannach, Sidra Naveed, and Michael Jugovac. 2016. User control in recommender systems: Overview and interaction challenges. In *International Conference on Electronic Commerce and Web Technologies*.

[29] Jiepu Jiang, Daqing He, and James Allan. 2014. Searching, Browsing, and Clicking in a Search Session: Changes in User Behavior by Task and over Time. In *Proceedings of the 37th International ACM SIGIR Conference on Research Development in Information Retrieval* (Gold Coast, Queensland, Australia) *(SIGIR '14)*. Association for Computing Machinery, New York, NY, USA, 607–616. https://doi.org/10.1145/2600428.2609633

[30] Yucheng Jin, Bruno Cardoso, and Katrien Verbert. 2017. How do different levels of user control affect cognitive load and acceptance of recommendations?. In *Workshop on Interfaces and Human Decision Making for Recommender Systems at RecSys*.

[31] Yucheng Jin, Nava Tintarev, and Katrien Verbert. 2018. Effects of Personal Characteristics on Music Recommender Systems with Different Levels of Controllability. In *RecSys*.

[32] Daniel King, Doug Downey, and Daniel S. Weld. 2020. High-Precision Extraction of Emerging Concepts from Scientific Literature. In *SIGIR*.

[33] Bart P. Knijnenburg, Niels J.M. Reijmer, and Martijn C. Willemsen. 2011. Each to His Own: How Different Users Call for Different Interaction Methods in Recommender Systems. In *RecSys*.

[34] Ajith Kodakateri Pudhiyaveetil, Susan Gauch, Hiep Luong, and Josh Eno. 2009. Conceptual Recommender System for CiteSeerX. In *RecSys*.

[35] Pang Wei Koh, Thao Nguyen, Yew Siang Tang, Stephen Mussmann, Emma Pierson, Been Kim, and Percy Liang. 2020. Concept Bottleneck Models. In *ICML*.

[36] Joseph Konstan and Loren Terveen. 2021. Human-centered recommender systems: Origins, advances, challenges, and opportunities. *AI Magazine* (2021).

[37] Maya Krishnan. 2020. Against interpretability: a critical examination of the interpretability problem in machine learning. *Philosophy & Technology* 33, 3 (2020), 487–502.

[38] Benjamin Charles Germain Lee, Doug Downey, Kyle Lo, and Daniel S Weld. 2020. LIMEADE: A General Framework for Explanation-Based Human Tuning of Opaque Machine Learners. *arXiv preprint arXiv:2003.04315* (2020).

[39] Kenton Lee, Ming-Wei Chang, and Kristina Toutanova. 2019. Latent Retrieval for Weakly Supervised Open Domain Question Answering. In *ACL*.

[40] Hanze Li, Scott Sanner, Kai Luo, and Ga Wu. 2020. A Ranking Optimization Approach to Latent Linear Critiquing for Conversational Recommender Systems. In *Proceedings of the 14th ACM Conference on Recommender Systems* (Virtual Event, Brazil) *(RecSys '20)*. Association for Computing Machinery, New York, NY, USA, 13–22. https://doi.org/10.1145/3383313.3412240

[41] Jiwei Li, Alan Ritter, and Eduard Hovy. 2014. Weakly Supervised User Profile Extraction from Twitter. In *ACL*.

[42] Kyle Lo, Lucy Lu Wang, Mark Neumann, Rodney Kinney, and Daniel Weld. 2020. S2ORC: The Semantic Scholar Open Research Corpus. In *ACL*.

[43] Pasquale Lops, Marco de Gemmis, and Giovanni Semeraro. 2011. Content-based recommender systems: State of the art and trends. *Recommender systems handbook* (2011), 73–105.

[44] Max Losch, Mario Fritz, and Bernt Schiele. 2019. Interpretability beyond classification output: Semantic bottleneck networks. *preprint arXiv:1907.10882* (2019).

[45] Kai Luo, Scott Sanner, Ga Wu, Hanze Li, and Hojin Yang. 2020. Latent Linear Critiquing for Conversational Recommender Systems. In *The Web Conference*.

[46] Kai Luo, Hojin Yang, Ga Wu, and Scott Sanner. 2020. Deep Critiquing for VAE-Based Recommender Systems. In *Proceedings of the 43rd International ACM SIGIR Conference on Research and Development in Information Retrieval* (Virtual Event, China) *(SIGIR '20)*. Association for Computing Machinery, New York, NY, USA, 1269–1278. https://doi.org/10.1145/3397271.3401091

[47] Cataldo Musto, Fedelucio Narducci, Marco Polignano, Marco de Gemmis, Pasquale Lops, and Giovanni Semeraro. 2020. Towards Queryable User Profiles: Introducing Conversational Agents in a Platform for Holistic User Modeling. In *Adjunct Publication of UMAP*.

[48] Sheshera Mysore, Arman Cohan, and Tom Hope. 2022. Multi-Vector Models with Textual Guidance for Fine-Grained Scientific Document Similarity. *NAACL*.
[49] Preksha Nema, Alexandros Karatzoglou, and Filip Radlinski. 2021. Disentangling Preference Representations for Recommendation Critiquing with ß-VAE. In *CIKM*.
[50] Nikolaos Pappas and Andrei Popescu-Belis. 2013. Combining content with user preferences for TED lecture recommendation. In *2013 11th International Workshop on Content-Based Multimedia Indexing (CBMI)*. IEEE, 47–52.
[51] Gabriel Peyré, Marco Cuturi, et al. 2019. Computational optimal transport: With applications to data science. *Foundations and Trends in Machine Learning* (2019).
[52] Filip Radlinski, Krisztian Balog, Fernando Diaz, Lucas Gill Dixon, and Ben Wedin. 2022. On Natural Language User Profiles for Transparent and Scrutable Recommendation. In *SIGIR*.
[53] Behnam Rahdari, Peter Brusilovsky, and Alireza Javadian Sabet. 2021. Connecting Students with Research Advisors Through User-Controlled Recommendation. In *RecSys*.
[54] Nils Reimers and Iryna Gurevych. 2019. Sentence-BERT: Sentence Embeddings using Siamese BERT-Networks. In *EMNLP*.
[55] Steffen Rendle, Christoph Freudenthaler, Zeno Gantner, and Lars Schmidt-Thieme. 2009. BPR: Bayesian Personalized Ranking from Implicit Feedback. In *UAI*.
[56] Isac S. Ribeiro, Rodrygo L.T. Santos, Marcos A. Gonçalves, and Alberto H.F. Laender. 2015. On Tag Recommendation for Expertise Profiling: A Case Study in the Scientific Domain. In *WSDM*.
[57] Alireza Salemi, Sheshera Mysore, Michael Bendersky, and Hamed Zamani. 2023. LaMP: When Large Language Models Meet Personalization. arXiv:2304.11406
[58] Tobias Schnabel, Saleema Amershi, Paul N. Bennett, Peter Bailey, and Thorsten Joachims. 2020. The Impact of More Transparent Interfaces on Behavior in Personalized Recommendation. In *SIGIR*.
[59] Nihar B. Shah. 2022. Challenges, Experiments, and Computational Solutions in Peer Review. *Commun. ACM* 65, 6 (may 2022), 76–87. https://doi.org/10.1145/3528086
[60] Damien Sileo, Wout Vossen, and Robbe Raymaekers. 2022. Zero-Shot Recommendation as Language Modeling. In *ECIR*.
[61] Kyle Swanson, Lili Yu, and Tao Lei. 2020. Rationalizing Text Matching: Learning Sparse Alignments via Optimal Transport. In *ACL*.
[62] Jaime Teevan, Susan T. Dumais, and Eric Horvitz. 2005. Personalizing Search via Automated Analysis of Interests and Activities. In *Proceedings of the 28th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval* (Salvador, Brazil) *(SIGIR '05)*. Association for Computing Machinery, New York, NY, USA, 449–456. https://doi.org/10.1145/1076034.1076111
[63] Chong Wang and David M. Blei. 2011. Collaborative Topic Modeling for Recommending Scientific Articles. In *KDD*.
[64] Hao Wang, Binyi Chen, and Wu-Jun Li. 2013. Collaborative Topic Regression with Social Regularization for Tag Recommendation. In *IJCAI*.
[65] Haonan Wang, Chang Zhou, Carl Yang, Hongxia Yang, and Jingrui He. 2021. Controllable Gradient Item Retrieval. In *Web Conference*.
[66] Jason Weston, Ron J. Weiss, and Hector Yee. 2013. Nonlinear Latent Factorization by Embedding Multiple User Interests. In *RecSys*.
[67] Ga Wu, Kai Luo, Scott Sanner, and Harold Soh. 2019. Deep Language-Based Critiquing for Recommender Systems. In *Proceedings of the 13th ACM Conference on Recommender Systems* (Copenhagen, Denmark) *(RecSys '19)*. Association for Computing Machinery, New York, NY, USA, 137–145. https://doi.org/10.1145/3298689.3347009
[68] Yuhuai Wu, Markus Norman Rabe, DeLesley Hutchins, and Christian Szegedy. 2022. Memorizing Transformers. In *International Conference on Learning Representations*. https://openreview.net/forum?id=TrjbxzRcnf-
[69] Hojin Yang, Tianshu Shen, and Scott Sanner. 2021. Bayesian Critiquing with Keyphrase Activation Vectors for VAE-Based Recommender Systems. In *SIGIR*.
[70] Longqi Yang, Tobias Schnabel, Paul N. Bennett, and Susan Dumais. 2021. Local Factor Models for Large-Scale Inductive Recommendation. In *Proceedings of the 15th ACM Conference on Recommender Systems* (Amsterdam, Netherlands) *(RecSys '21)*. Association for Computing Machinery, New York, NY, USA, 252–262. https://doi.org/10.1145/3460231.3474276
[71] Longqi Yang, Michael Sobolev, Yu Wang, Jenny Chen, Drew Dunne, Christina Tsangouri, Nicola Dell, Mor Naaman, and Deborah Estrin. 2019. How Intention Informed Recommendations Modulate Choices: A Field Study of Spoken Word Content. In *WWW*.
[72] ChingMan Au Yeung, Nicholas Gibbins, and Nigel Shadbolt. 2008. A Study of User Profile Generation from Folksonomies. In *Workshop on Social Web and Knowledge Management @ WWW*.
[73] Hamed Zamani and W. Bruce Croft. 2020. Learning a Joint Search and Recommendation Model from User-Item Interactions. In *WSDM*.
[74] Erheng Zhong, Nathan Liu, Yue Shi, and Suju Rajan. 2015. Building Discriminative User Profiles for Large-Scale Content Recommendation. In *KDD*.