Pulling the Right Levers: Steerable Collaborative Autoencoders with Sparse Autoencoders

Martin Spišák[1,3,*], Vojtěch Vančura[2,3], Rodrigo Alves[2,4], Petr Škoda[3] and Ladislav Peška[3,*]

[1]TopK, San Francisco, California, United States
[2]Recombee, Prague, Czechia
[3]Faculty of Mathematics and Physics, Charles University, Prague, Czechia
[4]Faculty of Information Technology, Czech Technical University in Prague, Czechia

Abstract

Sparse autoencoders (SAEs) have recently emerged as powerful tools for introspection in large language models. SAEs can uncover interpretable features at different levels of granularity and enable targeted steering of the generation process by selectively activating specific neurons in their latent activations.

In this work, we introduce *PuRL*, a pipeline for SAE application in collaborative autoencoders (CFAEs), a widely adopted class of recommendation algorithms, which aims to extract similarly interpretable features from representations learned purely from interactional data. In extensive experiments, we first demonstrate that SAE-based reconstruction can preserve most of the downstream recommendation accuracy while also exposing previously unrecognized failure modes in some recommendation architectures. We further demonstrate that many learned SAE neurons correspond to semantically meaningful concepts, even without any metadata supervision. Leveraging these neurons as interpretable "control levers", we show that recommendations can be steered toward desired segments in a controlled manner, supported by quantitative evaluation and qualitative analysis. Finally, we provide an interactive demo showcasing the proposed steering pipeline at https://steerable-collaborative-filtering.streamlit.app/.

Keywords

Sparse autoencoders, Interpretable recommenders, User control

1. Introduction

Collaborative-filtering (CF) techniques have been dominating the recommender systems' research since the late 1990s, following their success on early benchmarks such as the MovieLens dataset [1] and the Netflix Prize competition [2]. While the dawn of deep learning partially shifted the focus towards approaches incorporating multiple input modalities [3] or sequential processing [4], pure CF techniques remain fundamental, especially for the initial stages of the recommendation pipeline, i.e., candidate retrieval.

Among the plethora of CF algorithms, **collaborative autoencoders** (CFAEs) [5, 6, 7, 8, 9] belong to the most widely used approaches for this task. CFAEs are recommendation methods designed to encode sparse user–item interaction data into a dense, lower-dimensional latent space and then reconstruct it, enabling the extraction of information crucial for matching users with items. By focusing on the most relevant features of user behavior and item characteristics during reconstruction, these models reveal latent patterns and hidden relationships that are not immediately apparent in the raw data. This versatility, combined with strong recommendation performance, has made them a popular choice among researchers and practitioners for addressing challenges such as data sparsity and noise [10, 11], making these models a surging trend in the recommendation industry [7, 12, 13, 14].

However, despite their flexibility and robust performance, understanding the underlying mechanisms and characteristics hidden in the latent structure of CFAEs remains a very challenging task [15, 16, 17].

IntRS'26: Joint Workshop on Interfaces and Human Decision Making for Recommender Systems, September 28, 2026, Minneapolis
*Corresponding authors.
martin@topk.io (M. Spišák); vojtech.vancura@recombee.com (V. Vančura); rodrigo.alves@fit.cvut.cz (R. Alves); petr.skoda@matfyz.cuni.cz (P. Škoda); ladislav.peska@matfyz.cuni.cz (L. Peška)
0009-0006-7763-5575 (M. Spišák); 0000-0003-2638-9969 (V. Vančura); 0000-0001-7458-5281 (R. Alves); 0000-0002-2732-9370 (P. Škoda); 0000-0001-8082-4509 (L. Peška)

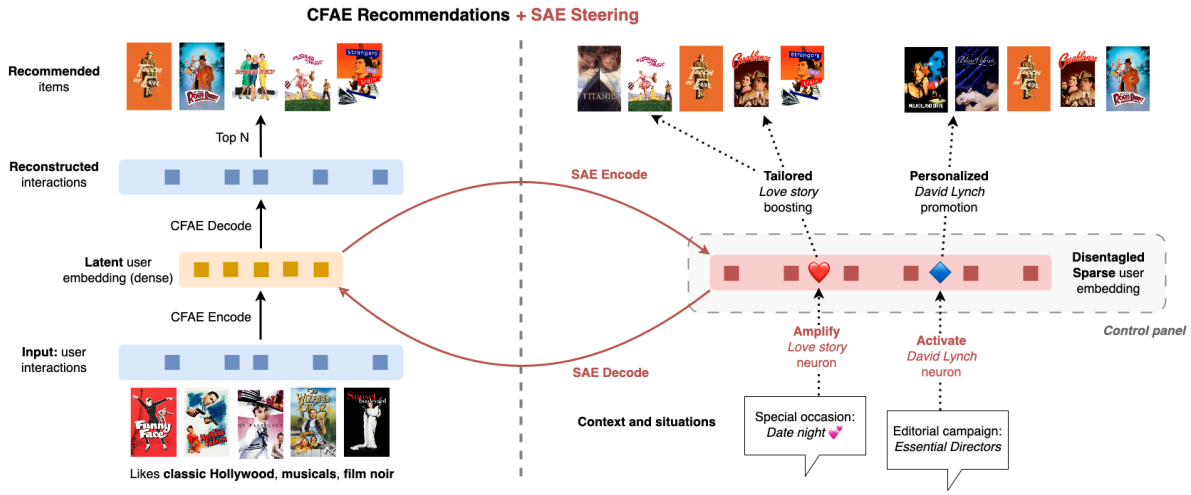

Figure 1: Overview of the *PuRL* approach. A sparse autoencoder (SAE) is applied to dense user embeddings inside collaborative filtering autoencoders (CFAEs), disentangling them into a set of interpretable features. These features receive labels based on their activation patterns and are provided to the users or editors as levers of a "control panel". By **Pu**lling the **R**ight **L**evers, i.e., selectively amplifying activations of particular neurons, recommendations can be steered toward context- or intent-specific preferences.

The abstract encoder-decoder mechanism not only results in opaque model behavior, making it difficult to track how recommendations are derived, but also limits the capacity of different agents (e.g., users, editors, providers) to influence or steer the recommendation process. This lack of transparency is inherent to other CF techniques based on latent representations, and poses significant challenges when diagnosing issues, fine-tuning performance, ensuring fairness, adhering to ethical standards, or granting control over recommendation outcomes [18]. Even shallow architectures, which might appear straightforward at first glance, can exhibit unexpected behavior in real-world situations [19].

The problem of understandability and interpretability of models' internal mechanisms is not limited to CF, but rather inherent to many machine learning application areas. For instance, notable improvements have been recently made in the field of large language models (LLMs), using a different type of autoencoders - a **sparse autoencoder (SAE)**.

In particular, SAEs are used to identify highly interpretable, monosemantic features (i.e., a dictionary of concepts)[1] in the residual streams of language models [20, 21, 22, 23, 24]. This approach is remarkable for its elegance: SAEs are shallow neural networks trained in a self-supervised manner to reconstruct activation vectors at a given network location under sparsity-inducing constraints. These constraints encourage each activation to be represented using only a small number of latent features, forcing the model to disentangle their meanings. Owing to this architectural simplicity, SAEs are highly approachable in practice. In this research, we investigate the possibility of using **SAEs** as a *reliable* interface for interpreting and steering the outputs of collaborative filtering algorithms, specifically **CFAEs**. We illustrate this approach in Figure 1. On the left, a standard CFAE encodes a user's (sparse) interaction vector into a dense latent embedding, which is then decoded, producing recommendation scores for all candidate items and generating Top-N recommendations. This process alone offers limited options for external influence, apart from modifying the input interaction vector directly. Therefore, as shown on the right of Figure 1, we propose inserting an SAE between the CFAE's encoder and decoder. This SAE "hook" maps the dense latent user representation to a higher-dimensional, sparsely activated user code – effectively exposing a control panel of interpretable levers (neurons), each corresponding to a distinct concept. For example, a neuron may be associated with *love stories* (depicted as a red heart), or *David Lynch* (blue diamond). [2] To steer the recommendations, we may suppress or amplify one or

[1]We understand the term *concept* as a latent yet meaningful property of an item.

[2]Interestingly, our experiments (Section 4.3) show that meaningful neuron–concept correspondence emerges through self-supervised SAE training on purely interaction-based data – that is, even without access to item metadata.

more levers (e.g., boosting the *love story* neuron in a *date night* context). The modified sparse code then passes through the SAE decoder and subsequently the CFAE decoder to generate recommendations *steered* toward the desired theme while maintaining original user preferences as much as possible. By exposing and manipulating these monosemantic controls, our method enables fine-grained, human-understandable steering of recommendations. Moreover, different agents can initiate these semantic boosts: for example, (1) a user might increase the *love story* lever for a romantic evening via a graphical interface, while (2) an editor could selectively boost the *David Lynch* lever (even if it was not part of the user's original interests) to craft personalized curated recommendations for special occasions such as sales or aniversaries. Ultimately, our approach provides a key interface for seamless, personalized control over recommendations through semantic-level interventions rather than manual item selection.

Although previous work has addressed the steering of general recommendation models [25, 26, 27] as well as interpretability of CFAE-like models [28, 19, 29, 30, 31, 32], none of these approaches has applied SAEs to generate interpretable user embeddings while *simultaneously* steering recommendations. Moreover, no previous study has investigated the nuanced characteristics of user–item interaction reconstruction quality in CFAEs, nor has any work systematically examined the sparse dictionary of SAEs in relation to fine-grained structured metadata for recommender systems. To address these research gaps, our main contributions are twofold:

- We propose *PuRL* (**Pu**lling the **R**ight **L**evers), a modular pipeline for steerable collaborative filtering autoencoders. *PuRL* augments a trained CFAE with a sparse autoencoder to learn disentangled user representations, automatically associates latent neurons with semantic concepts, and finally discloses the resulting "control panel" for steering purposes.
- We provide the first comprehensive empirical study of SAE-based steering in collaborative autoencoders. Across multiple CFAE architectures, datasets, and SAE variants, we show when sparse reconstruction preserves recommendation quality, identify previously unrecognized failure modes of certain architectures, demonstrate that meaningful semantic concepts emerge without metadata supervision, and confirm that these concepts enable effective recommendation steering through both quantitative and qualitative evaluation. Finally, an interactive demonstration of the complete pipeline (https://steerable-collaborative-filtering.streamlit.app/) is released to facilitate inspection and future research.

1.1. Background and Notations

For a set of users $\mathcal{U} = \{1, 2, \dots, m\}$ and a set of items $\mathcal{I} = \{1, 2, \dots, n\}$, let $X \in \{0, 1\}^{m \times n}$ be a partially-observed matrix of binary interactions (e.g., clicks, purchases), where the rows represent users and the columns represent items, i.e. x_{ij} denotes the presence or absence of an interaction between user-i and item-j. Typically, the matrix X is very sparse, meaning only a small fraction of user–item interactions have been observed. Moreover, let $x_i \in \{0, 1\}^n$ denote the i-th row vector of the matrix X. Using this notation, the goal of a recommender system based on collaborative filtering is to learn a function $f_\theta : \mathcal{U} \times \mathcal{I} \to \mathbb{R}$ (parametrized by θ) that predicts a relevance score $r(u, i) = f_\theta(u, i, X)$ for user $u \in \mathcal{U}$ and item $i \in \mathcal{I}$ using the interaction matrix X.

The main aim of our approach is to realize user control over recommender systems by letting users steer recommendations towards more desirable results. Let $\mathcal{S} \subseteq \mathcal{I}$ be a *segment* of items that share a common *semantic concept*,[3] formally represented by an inclusion function $g(i, \mathcal{S})$, where 0 denote item i does not belong to the segment and 1 denote a perfect membership. Then, we can define steering toward a target segment $\mathcal{S}$ as a controlled transformation of the user–item relevance function f_θ such that the predicted relevance is adjusted to favor items from the target segment $\mathcal{S}$ while maintaining consistency with the original user preference model. Formally, steering induces a new relevance estimator $f_{\theta,\mathcal{S},\alpha}(u, i) = \Phi\big(f_\theta(u, i), g(i, \mathcal{S}), \alpha\big)$, where $g(i, \mathcal{S})$ quantifies the relevance alignment of item i

[3] E.g., movies labeled with a particular tag, or sharing a similar plot twist.

to the segment $\mathcal{S}$, and $\Phi(\cdot)$ is a steering operator which combines the base relevance and the steering alignment, controlled by intensity parameter α.

Our methodology employs an autoencoder architecture. A core optimization problem for an autoencoder can be defined as

$$\min_{\mathscr{E},\mathscr{D}} \sum_{i=1}^{m} \ell\big(x_i, \sigma(x_i)\big) \qquad \text{with } \sigma(x) = \big(\mathscr{D} \circ \mathscr{E}\big)(x),$$

where $\mathscr{E}$ and $\mathscr{D}$ denote, respectively, the encoder and decoder functions, $f \circ g$ denotes the composition of functions f and g, and $\ell(x, y)$ is a reconstruction loss that measures the discrepancy between vectors x and y. Since we embed SAE within the CFAE in the proposed pipeline, the nested architecture can be described as

$$\sigma_{c,s}(x) = \big(\mathscr{D}_c \circ \mathscr{D}_s \circ \mathscr{E}_s \circ \mathscr{E}_c\big)(x), \tag{1}$$

where $\mathscr{E}_c$ and $\mathscr{D}_c$ represent encoder and decoder of the "outer" CFAE, while $\mathscr{E}_s$ and $\mathscr{D}_s$ represent the "inner" SAE.

2. Related Work

Collaborative Autoencoders. Together with two-tower models [33], neighborhood-based approaches [34], and graph neural networks [35], CFAEs are among the most popular choices in production recommender systems. CFAEs are especially valued for the initial retrieval phase of the recommendation generation due to their efficient representation learning under sparse implicit feedback and high scalability [36]. In our work, we focus on two popular and complementary classes of CFAEs: *linear* and *variational*.

Linear autoencoders [6, 7, 29, 14, 37] are *shallow* models favored for their straightforward, easy-to-implement, and generally scalable design. While the most popular representative of this class is probably the EASE algorithm [7], its $O(n^2)$ model size and fused encoder and decoder limit its applicability in our scenarios. Therefore, instead of EASE, we used its low-rank approximation, ELSA [29], which effectively mitigates both issues while retaining highly competitive performance.

Variational autoencoders (VAE) [8, 9, 38, 39] belong to a family of *deep, generative* models and are known for their ability to model complex data distributions through probabilistic latent representations, offering enhanced flexibility and expressiveness compared to linear models. Unlike classical autoencoders that project inputs to a single point in the bottleneck latent space, VAE representations $z_i = \mathscr{E}_c(x_i)$ are transformed via a function to produce a probability $\pi(y)$ distribution over the items, from which the click history x_i is assumed to have been generated. In this study, we experimented with MultVAE [8], one of the best-known VAE representatives.

Sparse Autoencoders. Sparse autoencoders (SAEs) [21, 40] are neural network architectures trained to reconstruct their input while promoting sparsity in high-dimensional[4] latent activations. Unlike traditional autoencoders, which compress input into a lower-dimensional bottleneck layer, SAEs map their input to a more expressive latent space but constrain the activations by allowing only a few nonzero entries. There are several ways to impose such an objective, e.g., by regularizing the size of the learned latent representation [21], or by imposing a rigid constraint on the number of active latents (using, e.g., Top-k [40] or batch Top-k [41] activation functions). The resulting representations often tend to be more interpretable, with individual neurons representing distinct, monosemantic concepts. Promisingly, recent works have demonstrated the effectiveness of SAEs in disentangling representations from highly expressive models, including state-of-the-art LLMs [23, 24].

Control and Transparency in Recommender Systems. While the prevalent recommendation paradigm is to supply users with recommendations directly with no user intervention, numerous works suggest that users benefit (e.g., through better recommendation acceptance) when they have

[4]The hidden layer dimension is often several times higher than the input dimension.

some degree of control over the system [42, 25, 26, 27, 43, 44, 45]. For example, Parra and Brusilovsky [44] proposed an interactive interface that allowed users to specify weights for fusing individual recommending methods. Liang and Willemsen [45] let users control the trade-off between personalized and representative music recommendations while exploring a new genre. Similarly, Millecamp et al. [46] developed a UI for adjusting parameters like danceability for music recommenders. Nevertheless, these approaches relied on post-processing to incorporate user control, while our proposal is capable of addressing it within the model itself. Closer to our work is the TEASER model proposed by De Pauw et al. [47]. TEASER is a hybrid interactive recommendation model based on EDLAE [37], a linear CFAE similar to one of our backbones. Nevertheless, TEASER directly represents items through their metadata, essentially using a fixed content-based decoder, while only learning the encoder network. As such, the metadata-neuron mappings are hardcoded and can be directly used as "control levers". On the other hand, such a rigid item representation may result in lower recommendation accuracy. In contrast, our "control levers" emerge organically within an SAE embedded into a CFAE and reflect solely the properties implicitly available in the interaction data alone; we use metadata merely to label the neurons. As such, *PuRL* may deliver better recommendation accuracy with sufficient user control over existing metadata, while also allowing steering towards concepts revealed only in interactional data. While user control over recommendations has already received notable attention, *editorial control* is also critically important in industry settings, despite being less explored to date [48, 49]. Existing approaches typically rely on simple item-level boosting or intervene during post-processing of recommendations [50, 49], which limits the flexibility of intervention due to dependence on the initial item rankings. Instead, *PuRL* paves the way for fine-grained in-process intervention already at the candidate generation stage.

Finally, within the realm of interpretable recommender systems [51], *PuRL* qualifies as a hybrid approach, combining interaction data with item metadata to provide interpretable insights. Crucially, unlike most existing methods that focus on post hoc recommendation justifications [52, 53, 54, 55, 56, 57], our technique, similarly to [28, 58], enables inspection of the model's internal mechanics and thus contributes to the recommendation transparency as well.

Using SAEs in CF techniques. We are not aware of any directly related approaches aiming to use SAEs to enable steering of CFAE-based recommendations. While some proposals for using metadata for CFAE interpretation exist [47], none have used SAEs for that purpose. Another line of work uses SAEs to enforce some high-level objectives of recommender systems, such as popularity debiasing [59] or calibration [60]. Nevertheless, they did not delve into finer-grained neuron labeling needed for broad-purpose user or editorial steering. In fact, to the best of our knowledge, Wang et al. [58] and Klenitskiy et al. [61] are the closest works that use SAEs to interpret the recommendation models. Nevertheless, they focused on a different recommendation task (sequential recommendation) and evaluated their approach on transformer-based architectures such as SASRec [4], which are highly similar to those typically studied in prior SAE works. Moreover, although Wang et al. [58] briefly mention the possibility of recommendation steering, this aspect is not systematically explored or evaluated, whereas Klenitskiy et al. only focused on coarse-grained genre data. In contrast, our work focuses on establishing the foundations for fine-grained recommendation steering by using SAEs within CFAEs. We demonstrate that not all CFAE architectures are equally robust to sparse reconstruction and that the choice of reconstruction loss and a particular steering mechanism can have both positive and detrimental effects, depending on the selected backbone recommender. This corrects the over-optimistic conclusions about SAE incorporation into arbitrary recommender architectures [58].

3. Methodology

To enable the steering of CFAE-based recommenders, we propose a four-stage pipelined approach denoted as *PuRL* (**Pu**lling the **R**ight **L**evers). *PuRL* works as follows: First, the CFAE model is trained as usual. Then, we nest an "SAE hook" to reconstruct dense CFAE embeddings using a wider, yet sparsely activated, layer. Next, we create a mapping between available semantic concepts and individual neurons on the SAE's hidden layer. Finally, the labeled neurons are supplied as a "control board" to interested

parties, enabling targeted in-process steering of recommendations by adjusting the neurons' activation patterns.

Note that, in principle, each stage is independent of the composition of the others. This makes *PuRL* highly modular, where individual variants of each stage can be deployed in a plug-and-play fashion. Therefore, in this foundational work, we intentionally focused on evaluating relatively simple variants for each stage, thereby providing a straightforward, scalable, highly reproducible, yet competitive baseline for future research.

3.1. Network Architecture

The network architecture overview is depicted in Figure 1. In particular, we propose a nested autoencoder architecture, where a CFAE serves as the outer layer, while an SAE serves as the inner layer. The resulting network can be seen as a function $\sigma_{c,s}(x) = \big(\mathscr{D}_c \circ \mathscr{D}_s \circ \mathscr{E}_s \circ \mathscr{E}_c\big)(x)$, where x represent a vector of user's interactions. Naturally, while in general $\mathscr{E}_c : \mathbb{R}^n \to \mathbb{R}^r$ and $\mathscr{E}_s : \mathbb{R}^p \to \mathbb{R}^d$, we implicitly assume $p \equiv r$ when we nest an SAE within a CFAE.

CFAEs. As representatives for the CFAE, we selected well-known ELSA [29] and MultVAE [8] algorithms. In the case of ELSA, which optimizes $\min_A \|X - X(AA^\top - I)\|_F^2 \quad \text{s.t.} \quad \|a_i\|_2 = 1$, we consider its first step as the encoder, i.e., $\mathscr{E}_c(x) = A^T x$, while the remainder as decoder, i.e., $(\mathscr{D}_c \circ \mathscr{E}_c)(x) = A\mathscr{E}_c(x) - x$. For MultVAE, the embeddings are taken from the mean head of the encoder network.

SAEs. As representatives for SAE, we experimented with a basic ReLU SAE [21] (which, for the sake of clarity, we refer to as *Basic SAE*) and a k-sparse autoencoder, *TopK SAE* [40].

For an input vector $y_i \in \mathbb{R}^p$, *Basic SAE* constructs d-dimensional ReLU SAE encoder $\mathscr{E}_s$ and decoder $\mathscr{D}_s$ as:

$$\mathscr{E}_s(y_i) = \mathrm{ReLU}\big(W_{\mathscr{E}_s}(y_i - b_{\mathscr{D}_s}) + b_{\mathscr{E}_s}\big) \text{ and} \tag{2}$$

$$\mathscr{D}_s(\mathscr{E}_s(y_i)) = W_{\mathscr{D}_s}\mathscr{E}_s(y_i) + b_{\mathscr{D}_s}, \tag{3}$$

with learnable parameters $W_{\mathscr{E}_s} \in \mathbb{R}^{d\times p}$, $b_{\mathscr{E}_s} \in \mathbb{R}^d$, $W_{\mathscr{D}_s} \in \mathbb{R}^{p\times d}$, and $b_{\mathscr{D}_s} \in \mathbb{R}^p$. During both training and inference, input vectors are standardized before being passed through the SAE and de-standardized on the output.

TopK SAE differs from the Basic SAE only in the encoder, which is defined as $\mathscr{E}_s(y_i) = \mathrm{TopK}\big(W_{\mathscr{E}_s}(y_i - b_{\mathscr{D}_s}), k\big)$, where $\mathrm{TopK}(z,k)$ is an activation function that zeros out all but the k largest positive values of vector z. Following [23], we include a ReLU activation before applying the TopK function, forcing all activations to be non-negative. To align the two architectures, we incorporate an encoder bias term (similar to Basic SAE), resulting in:

$$\mathscr{E}_s(y_i) = \mathrm{TopK}\Big(\mathrm{ReLU}\big(W_{\mathscr{E}_s}(y_i - b_{\mathscr{D}_s}) + b_{\mathscr{E}_s}\big), k\Big). \tag{4}$$

This formulation makes the model equivalent to Basic SAE, with the only difference being the addition of the TopK activation.

Both SAE variants are optimized w.r.t. the following loss:

$$\min_{\mathscr{E}_s,\mathscr{D}_s} \sum_{i=1}^{m} \ell_s(y_i) = \sum_{i=1}^{m} \|y_i - \big(\mathscr{D}_s \circ \mathscr{E}_s\big)(y_i)\|_2^2 + \lambda\|\mathscr{E}_s(y_i)\|_1, \tag{5}$$

where λ is an L_1-norm regularization hyperparameter. Note that while λ serves as the primary controlling parameter for sparsity in Basic SAE, in TopK SAE it mainly prevents large spikes in activations and plays only a secondary role in enforcing sparsity. For the sake of simplicity, and because the SAEs we train are not very wide, we do not incorporate auxiliary losses, as described in [23].

3.2. Concept-Neuron Mapping

The next step in the *PuRL* pipeline is to map the activation of individual neurons in the sparse layer to semantically meaningful concepts and to analyze the quality of this mapping. In other words, we

need to identify what "levers" emerged from our model. This requires evidence of a reasonably narrow mapping between concepts and neurons in both directions.

To label the "levers", we employ a simple yet effective method that uses the available item metadata, e.g., user-created or curated textual tags as collected for the ML-25M and MSD datasets. Let $\mathcal{T}$ denote the set of all such tags. We first estimate the joint tag-item distribution, $\hat{p}(t \in \mathcal{T}, i \in \mathcal{I})$, by normalizing the sparse tag assignment count matrix.[5] Next, to link SAE neurons with entities in the data, we compute sparse representations for every item by passing its one-hot encoding through the encoder part of the network, i.e., $z_i = (\mathscr{E}_s \circ \mathscr{E}_c)(\texttt{onehot}(i))$. Multiplying the empirical tag-item distribution matrix by the item-neuron sparse activations produces a tag–activation matrix $\mathscr{M} : \mathbb{R}^{|\mathcal{T}| \times d}$, where d is the dimensionality of the SAEs' hidden layer.

From the $\mathscr{M}$ matrix, we derive TF-IDF scores in two orientations: $M_{t \to n}$, treats tags as terms and neurons as documents, while $M_{n \to t}$ treats neurons as terms and tags as documents. Note that this duality carries important semantical nuances. In particular, $\text{argmax}_n M_{t \to n}$ identifies, for each tag, the neuron whose firing is **most unique** to that tag, while $\text{argmax}_t M_{t \to n}$ selects, for each neuron, the tag that **best characterizes** its overall activity. Conversely, $\text{argmax}_t M_{n \to t}$ finds, for each neuron, the tag that elicits its **most distinctive** response, and $\text{argmax}_n M_{n \to t}$ finds, for each tag, the neuron that **most representatively** encodes it.

3.3. Recommendation Steering

Finally, the established concept–neuron mappings can be made available to interested parties as labeled "levers" to steer the recommendations to a desired segment of the catalog.

To realize the steering towards a segment $\mathcal{S}$ in the nested autoencoder architecture, we start with a sparse vector $z_u = (\mathscr{E}_s \circ \mathscr{E}_c)(x_u)$, representing a user x_u and a specific neuron j that best correspond to the desired segment based on the concept-neuron mapping. To achieve comparability, we first normalize z_u to unit sum, obtaining $\bar{z}_u$, and then construct a steered representation by a convex combination of the original profile and the targeted concept emphasis: $z_u^{\mathcal{S}} = (1 - \alpha) \times \bar{z}_u + \alpha \times \texttt{onehot}(j)$ for $\alpha \in [0, 1]$. The steered representation $z_u^{\mathcal{S}}$ is then passed through the network's decoder to obtain recommendations.

Finally, note that SAE's sparse layer dimension is typically larger than the number of tags, and the interactional data may capture finer concepts than those explicitly stated in the metadata. As such, it is safe to assume that multiple neurons may reflect the target segment $\mathcal{S}$. To adhere to that, instead of $\texttt{onehot}(j)$, we can use a (possibly weighted) list of several best neurons as the steering vector.

4. Experiments

We systematically evaluated all stages of the *PURL* pipeline. First, we investigate the robustness of CFAE latent representations to disentanglement and sparse reconstruction, analyze the behavior of different SAEs, and explore the attainable trade-off between the sparsity and downstream recommendation quality. Second, we study the "levers" emerging during self-supervised, interaction-only training and evaluate the quality of mapping to concepts available from metadata. Finally, we demonstrate how the discovered concept-neuron mapping can be leveraged to steer recommendation outcomes.[6]

4.1. Preliminaries

Datasets. We conduct experiments on two well-known recommendation datasets, namely MovieLens 25M (ML-25M) [1] and Million Song Dataset (MSD) [62], and follow the same pre-processing steps as described in [8]. For ML-25M, we first binarized the ratings by treating $r_{u,i} \geq 4$ as positive interactions (with a value of 1) and discarding the rest. We then remove users with fewer than 5 interactions. For MSD, we binarize play counts and consider them as implicit feedback. Next, we remove items with

[5] In different contexts, one may utilize any other prior information specifying how relevant a particular tag is for an item.

[6] The source codes, reproduction instructions, additional results, and a live demo are available at https://github.com/matospiso/Disentangling-user-embeddings-using-SAE/.

Table 1
Accuracy of ELSA and MultVAE on ML-25M and MSD datasets. d denotes the dimension of the bottleneck layer.

Dataset	Model	d	Recall@20	nDCG@20
ML-25M	ELSA	512	0.390 ± 0.002	0.353 ± 0.002
	ELSA	1024	0.396 ± 0.002	0.360 ± 0.002
	ELSA	2048	0.397 ± 0.002	0.363 ± 0.002
	MultVAE	256	0.375 ± 0.002	0.333 ± 0.002
	MultVAE	512	0.383 ± 0.002	0.340 ± 0.002
	MultVAE	1024	0.377 ± 0.002	0.334 ± 0.002
MSD	ELSA	512	0.245 ± 0.001	0.239 ± 0.001
	ELSA	1024	0.275 ± 0.001	0.269 ± 0.001
	ELSA	2048	0.298 ± 0.001	0.293 ± 0.001
	MultVAE	256	0.211 ± 0.001	0.203 ± 0.001
	MultVAE	512	0.241 ± 0.001	0.232 ± 0.001
	MultVAE	1024	0.252 ± 0.001	0.242 ± 0.001

fewer than 200 interactions and users with fewer than 20 interactions. Finally, for the presentation purposes, TMDB API (https://developer.themoviedb.org/) has been used to collect additional data about ML-25M items.

We split each dataset into training, validation, and test sets, employing the principle of strong generalization – ensuring that users are disjoint between the splits. We reserve 10% of users for the test set and an additional 10% for validation, leaving 128,621 and 457,084 users for training on ML-25M and MSD, respectively.

Training and Evaluating CFAEs. For each dataset, we train ELSA and MultVAE in three sizes: "small", "medium", and "large". In particular, for ELSA, we use embedding dimensions $d \in \{512, 1024, 2048\}$. For MultVAE, we select a neural architecture $[d_{\text{input}} \to 3d \to d]$, where d_{input} is the number of items and $d \in \{256, 512, 1024\}$, with a tanh activation. The final layer has two heads – one for the mean and one for the variance. No activation is applied to these heads. The decoder mirrors the encoder structure and applies a softmax activation to the output. This architecture largely follows the suggestions from the original paper [8]. We train all variants using the Adam optimizer [63] with $\alpha = 3 \cdot 10^{-4}$ for ELSA and 10^{-3} for MultVAE, $\beta_1 = 0.9$, and $\beta_2 = 0.99$. For MultVAE, we anneal the β parameter by 10^{-6} after each step. We train both models using a batch size of 1024 for up to 25 epochs. We employ early stopping for ELSA after 10 epochs without improvement in validation loss.

To contextualize the results discussed in the next subsection, we evaluate the CFAEs in terms of Recall@20 and nDCG@20 on the held-out sets. The evaluation targets are randomly selected as 20% of each user's interactions, with the remainder serving as inference input. We report test set results for all variants in Table 1. Although the peak performance of CFAEs is not the primary focus of our experiments, the results for ELSA are on par with those reported in [29]. For MultVAE, our results were approximately 5% lower than those of [8], which we attribute to a simpler annealing strategy and a less extensive hyperparameter search.

Training SAEs. We trained 12 variants of Basic SAE and 12 variants of TopK SAE for every combination of dataset, CFAE model, and its size, resulting in 288 sparse autoencoders in total. For both SAE architectures, we varied the width-to-input dimension ratios $\{2, 4, 8\}$, while for Basic SAE, we tuned the L_1 coefficient, and for TopK SAE, we tuned the k hyperparameter. Each SAE was trained on the embeddings of the training user interactions generated by the encoder $\mathscr{E}_c$ of its parent CFAE model. SAEs are optimized using Adam [63] with $\alpha = 3 \cdot 10^{-4}$, $\beta_1 = 0.9$, and $\beta_2 = 0.99$.Training proceeds with a batch size of 1024 for up to 250 epochs, with early stopping triggered after 50 epochs without improvement in validation loss.

TEASER. For the TEASER baseline, we used a gradient-based reimplementation of the original model, aiming for a practical baseline under the same protocol as the other recommenders. We used batch size

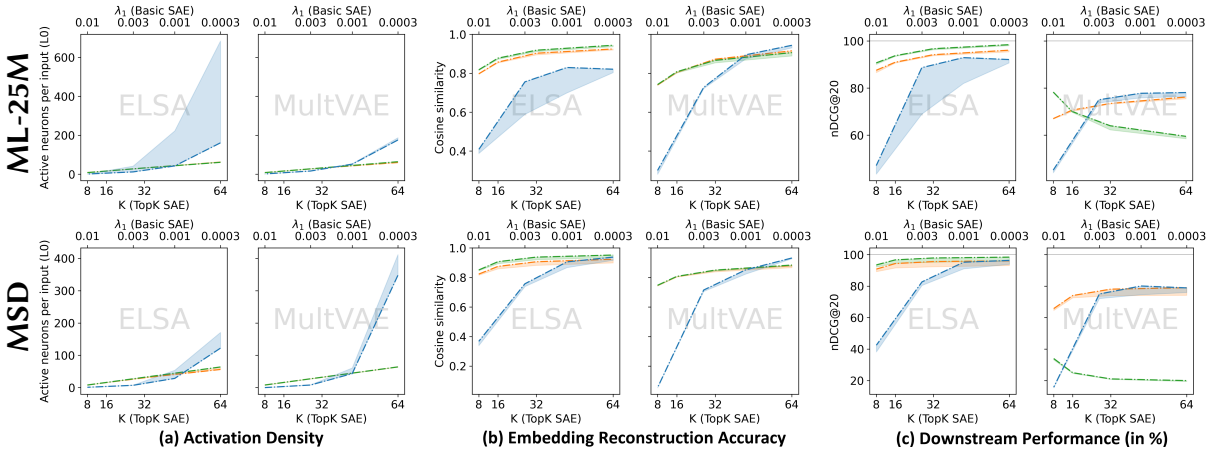

Figure 2: Results of SAE reconstruction on 1024-dimensional CFAEs w.r.t. varying sparsity-inducing hyperparameters (k, λ_1). **Main highlights:** (1) **TopK SAE** (orange) is simple to use and achieves a superior sparsity–accuracy trade-off compared to **Basic SAE** (blue), which is very sensitive to hyperparameter selection. (2) TopK SAEs reconstruct cosine-like embeddings (ELSA) accurately, while variational embeddings (MultVAE) are difficult to reconstruct without sacrificing performance. (3) Replacing the L_2 reconstruction loss in TopK SAE with **cosine similarity loss** (green) further improves the results for ELSA, but breaks the downstream performance of MultVAE.

1024, learning rate 10^{-4}, $\lambda_1 = 10^{-3}$, $\lambda_2 = 10^{-2}$, and Gaussian initialization with standard deviation 0.01. Unlike the SAE-based approach, TEASER is steered directly in metadata space: given a user profile $p_u = x_u E$, we form $p'_u = p_u + f_u$, where f_u is a sparse steering vector on the selected tag dimension, and decode p'_u with the fixed metadata decoder. TEASER therefore requires no concept-to-neuron mapping, since its control dimensions are predefined by the metadata vocabulary itself.

4.2. Reliability of Sparse Reconstruction

Before applying sparse neurons for steering, we first need to establish whether SAE-based reconstruction can be reliably used at inference time without prohibitively degrading downstream performance. If sparse reconstruction is unstable or lossy, it risks introducing harmful performance regressions in practical deployments. As we later show, while reconstruction is generally robust, certain types of embeddings remain brittle and susceptible to degradation. The reconstruction quality was tested using (i) the cosine similarity between the input and the reconstructed output; (ii) the downstream performance (Recall@20 and nDCG@20) relative to the performance of the unmodified CFAE, and (iii) the average number of neurons firing per input (denoted as L_0).

Figure 2 presents the results for SAEs based on 1024-dimensional ELSA and MultVAE backbones.[7] In general, CFAEs with nested SAEs can retain high performance relative to their original counterparts, though robustness varies across SAE and CFAE architectures. In particular, TopK SAEs consistently achieved a more favorable sparsity−-accuracy trade-off compared to Basic SAEs. The Basic SAE's λ_1 sparsity controlling hyperparameter is difficult to tune and highly sensitive; even small misconfigurations can significantly degrade either sparsity or reconstruction quality. In contrast, TopK SAEs explicitly constrain the number of active neurons, making them easier to configure and more effective overall.

For CFAEs that use cosine-like embeddings, TopK SAEs can reconstruct the latent space with minimal degradation in downstream performance. For example, on ML-25M, 1024-dimensional ELSA embeddings could be reconstructed using $k = 32$ neurons, while preserving 95% of the original Recall@20 and 94% of nDCG@20. Our ablation study further showed that replacing the L_2 reconstruction loss (see Eq. (5)) with a negative cosine similarity loss,

[7] Results for other backbone variants are available from the repository.

Table 2
Tags with highest (black) and lowest (grey) KL divergence from average activation distributions over all tags on the ML-25M dataset. A corresponding table for the MSD dataset is available in the repository.

MultVAE + L2	H	D_{KL}	**ELSA + L2**	H	D_{KL}	**ELSA + Cosine**	H	D_{KL}
james bond	2.06	14.30	quentin tarantino	4.65	14.60	quentin tarantino	4.90	14.34
quentin tarantino	2.86	14.24	wes anderson	4.09	14.25	coen brothers	5.21	14.20
studio ghibli	1.67	14.16	stanley kubrick	4.50	14.20	james bond	5.18	14.20
boring	5.44	5.76	boring	7.33	3.15	bd-r	8.22	2.65

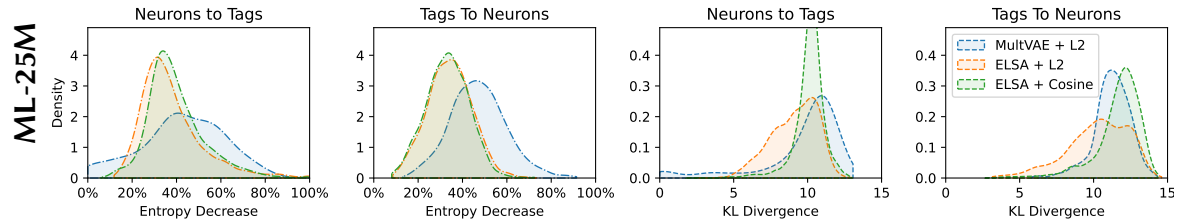

Figure 3: KL divergence and entropy decrease of tags and neurons. For "tags to neurons" direction, relative entropy decrease is defined as $(H - H_t)/H$, where H_t is the entropy of the tag's distribution of neuron activations, while H is the entropy of the average distribution over all tags. The other direction is defined analogously.

$$1 - \left(\frac{y_i}{\|y_i\|}\right)^T \left(\frac{\tilde{y}_i}{\|\tilde{y}_i\|}\right), \quad \text{where} \quad \tilde{y}_i = (\mathscr{D}_s \circ \mathscr{E}_s)(y_i), \tag{6}$$

improved the sparsity–accuracy trade-off for cosine-like embeddings, as used by ELSA. In the previously mentioned $k = 32$ case, this loss yielded 97% of ELSA's Recall@20 and nDCG@20. However, in the case of MultVAE, this loss no longer ensures a small Euclidean distance between variational embeddings – which represent samples from a probability distribution – and their reconstructions. As a result, downstream performance degraded significantly.

Overall, our results suggest that CFAE models optimized for cosine similarity between embeddings are particularly well-suited for the *PURL* pipeline. We hypothesize that this generalizes to other non-autoencoder CF architectures (e.g., matrix factorization or graph neural networks) that rely on embedding angles rather than distances for inference. We encourage future work to explore this broader applicability.

4.3. Interpretability of SAE Neurons

Now we need to determine whether a correspondence has emerged between sparse neurons and semantic concepts – that is, to identify which "levers" can be used to steer the recommendations. In this experiment, we utilized the TF-IDF based concept-neuron mapping as described in Section 3.2 on top of ML-25M and MSD tags. We preprocessed the dataset by removing tags that appear fewer than 100 times to reduce noise, and retained only those associated with items present in the interaction data. We analyzed three nested autoencoder variants as follows. We used 1024-dimensional ELSA and MultVAE on top of which we trained a TopK SAE with an L_2 reconstruction loss, an 8× scaling factor (resulting in an 8,192-dimensional sparse layer), $k = 16$, and $\lambda_1 = 0.0003$. Additionally, we used another TopK SAE with the same configurations but using cosine reconstruction loss on top of the ELSA backbone. In the results, the three experimental configurations were denoted as `ELSA+L2`, `MultVAE+L2`, and `ELSA+Cosine`, respectively.

To quantify the selectivity of neuron activation for individual tags, we analyzed the tag-to-neuron score matrix $M_{t\to n}$ using two distributional metrics. For each tag (i.e., row of $M_{t\to n}$, interpreted as a probability distribution via L_1 normalization denoted by $\tilde{\cdot}$), we computed (1) the entropy H, which measures the spreading of activation across neurons, and (2) the Kullback-Leibler divergence D_{KL},

Figure 4: Effects of concept steering via *PuRL* pipeline for a particular user.

which compares each tag τ_i's activation distribution $\tilde{S}_{t\to n}[\tau_i]$ to the average distribution across all tags $\mathbb{E}_\tau[M_{t\to n}[\tau]]$; formally $D_{\mathrm{KL}}\left(\mathbb{E}_\tau\left[\tilde{S}_{t\to n}[\tau]\right] \,\|\, \tilde{S}_{t\to n}[\tau_i]\right)$.

Figure 3 presents the distributions of D_{KL} and relative entropy decrease across tags for all three model variants, while Table 2 lists tags with the most and least specific activation patterns for the ML-25M dataset. We found that a large portion of tags exhibit significantly lower entropy than the per-model averages.[8] Similarly, many tags showed highly divergent activation patterns (e.g., 88% of `MultVAE+L2`, 59% of `ELSA+L2`, and 88% of `ELSA+Cosine` entries have $D_{\mathrm{KL}} > 10$ on the ML-25M dataset). That is, most tags induce rather unique activation patterns highly different from the average, which may be used for subsequent steering.

To further analyze the activation patterns, we focused on which tags induce the most/least specific activations. On ML-25M, the highest D_{KL} typically correspond to individuals associated with movies (i.e., actors, characters, writers, or directors), while tags with the lowest divergence tend to be vague and less informative about the movies themselves, e.g., *BD-R* (a type of optical media) or "*boring*". Similar patterns were also observed in MSD, where tags targeting compact subcultures (e.g., *Reggaeton*, *Kiwi*, *Turkish*, or *Celtic punk*) exhibited high D_{KL}, while general concepts like *favorites* or *rock* were on the other end of the spectrum.

We applied an analogous analysis in the opposite direction – from neurons to tags – using $M_{n\to t}$. As shown in Figure 3, many neurons exhibit large D_{KL} and relative entropy decrease, indicating sparse, targeted responses to a small subset of tags. Comparing the models, `MultVAE+L2` achieved the highest relative entropy decrease across both directions, followed by `ELSA+Cosine` and then `ELSA+L2`. For D_{KL}, `MultVAE+L2` and `ELSA+Cosine` showed similar selectivity, with `ELSA+L2` slightly trailing behind.

In summary, despite using only simple metadata for concept identification, we have shown that a significant portion of the learned neurons appears to be activated by compact, well-defined, and semantically meaningful features – revealing a "control board" with sufficient granularity, interpretability, and fidelity.

4.4. Practical Steering of Recommendations

We conclude our experimental study by evaluating whether the learned control panel of sparse neurons enables effective intervention in recommendation behavior. We begin with a qualitative evaluation depicted in Figure 4.[9] Here, we employed the `ELSA+L2` nested autoencoder and $\mathrm{argmax}_n M_{n\to t}$ concept-neuron mapping to steer the recommendations of a particular user toward several item segments in the movie domain. The user shown in the Figure originally interacted with 33 movies, and the default list of recommendations (denoted as *[no boost]*) reflects their general interests in classic Hollywood films, musicals, and film noir. Boosting the characteristic neurons corresponding to selected concepts (with $\alpha = 0.15$) shifts the recommendations toward the desired themes while preserving the user's original preferences. For instance, amplifying neurons associated with the "*Children*" and "*David Lynch*" tags results in the inclusion of *Toy Story* and *Mulholland Drive*, respectively. Notably, steering behaves as a **blend** of the original profile and the targeted concept. For example, in addition to the popular *Titanic*, amplifying the "*Love story*" neuron also promotes *The Sound of Music* and *Casablanca*, reflecting a preference for romantic films within classic Hollywood and musical genres.

[8] $H = 6.3$, 8.3, and 9.0 for `MultVAE+L2`, `ELSA+L2`, and `ELSA+Cosine`, respectively, on the ML-25M dataset.

[9] To facilitate further exploration, additional qualitative examples are made available through an interactive demo at https://steerable-collaborative-filtering.streamlit.app/.

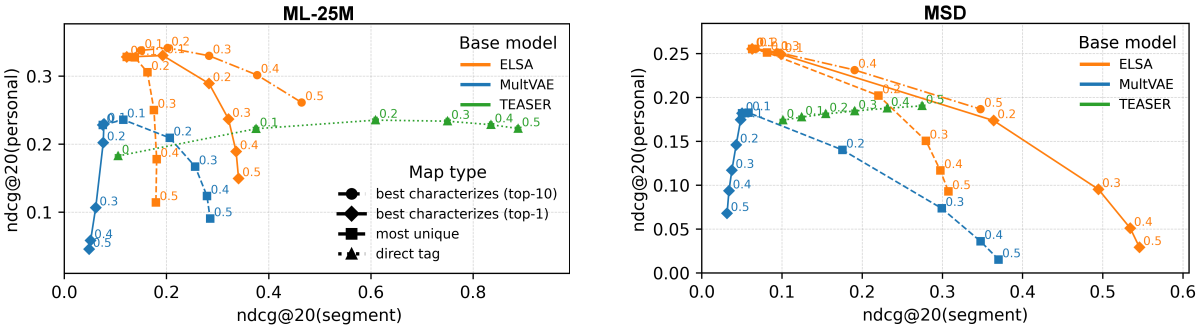

Figure 5: Results of the steering procedure for varying intensities α and concept-neuron mappings.

Next, we evaluated whether such a blend of original and segment-wise preferences can be achieved in general. For each test user, we selected the segment $\mathcal{S}_u$,[10] whose size in the hold-out set minus the size in the input set was the largest, i.e., the most under-represented segment of the user's interests. This creates an opportunity for recommendation steering to compensate for this mismatch. After identifying $\mathcal{S}_u$, we applied the steering toward this segment, using the *PuRL* with gradually increasing α. To map segments to neurons, we considered both $\text{argmax}_n M_{t\to n}$ (the most *unique* neuron) and $\text{argmax}_n M_{n\to t}$ (*best characterizing* neuron). To put our methods in context, we also depict the results of the TEASER baseline.

Figure 5 depicts the trade-off between recommendation accuracy w.r.t. the user's hold-out set (denoted as *nDCG@20(personal)*) and recommendation accuracy w.r.t. items from steered segment $\mathcal{S}_u$ (*nDCG@20(segment)*) for varying steering magnitude α. Overall, thanks to its direct metadata mappings, TEASER can deliver good steering performance. Nevertheless, the price for that is an inferior recommendation accuracy w.r.t. the user's general preferences (nDCG@20(personal)). *PuRL* pipeline provides much better steering-accuracy tradeoff as long as the desired steering magnitude is not too large. As for particular variants, `ELSA+L2` clearly outperformed `MultVAE+L2` on both datasets, while for ELSA, steering based on the *best characterizing* neuron outperformed that based on the *most unique* neuron. In contrast, for `MultVAE+L2`, steering through *best characterizing* neurons fails entirely, potentially due to substantial overlap in the $\text{argmax}_n M_{n\to t}$ mappings across tags. We hypothesize this is related to the lack of reconstruction robustness discussed in Section 4.2. Finally, we checked the impact of using *top-k* best neurons instead of top-1. This change improved the accuracy-steering tradeoff across all tested approaches. For clarity, Figure 5 depicts only the results of ELSA, with the top-10 best characterizing neurons, which represent the best of the evaluated variants. We further focused on the effect of TF-IDF weighting of neurons, but this did not provide further improvements.

Overall, our experiments confirmed that steering based on SAE, such as the *PuRL* pipeline, is a favorable option in certain situations. However, in contrast to [58, 61], we demonstrated that *not all CF architectures* are equally amenable to such modifications: different architectures may require variations in the subsequent pipeline, and, in some cases, steering must be applied with caution.

5. Conclusions and Future Work

In this work, we explored the applicability of SAEs to provide an interpretable and steerable layer on top of the CFAE. We proposed the *PuRL* pipeline instantiating this idea and evaluated its impact on (1) the downstream performance, (2) embeddings interpretability, and (3) steerability of recommendations by boosting specific concepts. Overall, the pipeline proved effective: TopK SAEs preserved almost all downstream accuracy for cosine-like CFAE embeddings; even very simple processing of structured metadata can yield interpretable concept-to-neuron mappings, which can be used to effectively and

[10]Here, we focused on simple segments defined through assigned tags: an item $i \in$ segment $\mathcal{S}_t$ induced by a tag t if t has been assigned to i.

efficiently steer the final recommendations. We also indicated several options and design choices that we advise against, as they may lead to unstable or inferior performance.

Nevertheless, our current work has several limitations. We evaluated relatively simple SAE variants, leaving room for improved accuracy via auxiliary losses (e.g., [23]) or more advanced SAE architectures (e.g., [41, 64]). Moreover, we only experimented with simple segment definitions, concept-neuron mappings, and steering vector definitions. Applying large language models to full-text item descriptions could reveal finer neuron–item relationships beyond existing metadata, and examining neuron activations on multi-item inputs may surface additional useful inter-item features. Finally, we hypothesized that *PuRL* pipeline should be robust for further RS backbones, which - just like in ELSA - are based on cosine similarity of embeddings. Therefore, we plan to investigate such RS, e.g., matrix factorization or graph neural networks, in the future.

Acknowledgments

This paper has been supported by the Czech Science Foundation (GAČR) project 25-16785S.

Declaration on Generative AI

During the preparation of this work, the authors used ChatGPT and Grammarly for grammar and spelling checks and text polishing. After using these tools, the authors reviewed and edited the content as needed and take full responsibility for the publication's content.

References

[1] F. M. Harper, J. A. Konstan, The movielens datasets: History and context, ACM Trans. Interact. Intell. Syst. 5 (2015). URL: https://doi.org/10.1145/2827872. doi:10.1145/2827872.

[2] Y. Koren, S. Rendle, R. Bell, Advances in Collaborative Filtering, Springer US, New York, NY, 2022, pp. 91–142. URL: https://doi.org/10.1007/978-1-0716-2197-4_3. doi:10.1007/978-1-0716-2197-4_3.

[3] C. Ganhör, M. Moscati, A. Hausberger, S. Nawaz, M. Schedl, A multimodal single-branch embedding network for recommendation in cold-start and missing modality scenarios, in: Proceedings of the 18th ACM Conference on Recommender Systems, RecSys '24, Association for Computing Machinery, New York, NY, USA, 2024, p. 380–390. URL: https://doi.org/10.1145/3640457.3688138. doi:10.1145/3640457.3688138.

[4] W.-C. Kang, J. McAuley, Self-attentive sequential recommendation, in: 2018 IEEE International Conference on Data Mining (ICDM), 2018, pp. 197–206. doi:10.1109/ICDM.2018.00035.

[5] S. Sedhain, A. K. Menon, S. Sanner, L. Xie, Autorec: Autoencoders meet collaborative filtering, in: Proceedings of the 24th international conference on World Wide Web, 2015, pp. 111–112.

[6] X. Ning, G. Karypis, Slim: Sparse linear methods for top-n recommender systems, in: Proceedings of the 2011 IEEE 11th International Conference on Data Mining, ICDM '11, IEEE Computer Society, USA, 2011, p. 497–506. URL: https://doi.org/10.1109/ICDM.2011.134. doi:10.1109/ICDM.2011.134.

[7] H. Steck, Embarrassingly shallow autoencoders for sparse data, in: The World Wide Web Conference, WWW '19, Association for Computing Machinery, New York, NY, USA, 2019, p. 3251–3257. URL: https://doi.org/10.1145/3308558.3313710. doi:10.1145/3308558.3313710.

[8] D. Liang, R. G. Krishnan, M. D. Hoffman, T. Jebara, Variational autoencoders for collaborative filtering, in: Proceedings of the 2018 World Wide Web Conference, WWW '18, International World Wide Web Conferences Steering Committee, Republic and Canton of Geneva, CHE, 2018, p. 689–698. URL: https://doi.org/10.1145/3178876.3186150. doi:10.1145/3178876.3186150.

[9] I. Shenbin, A. Alekseev, E. Tutubalina, V. Malykh, S. I. Nikolenko, Recvae: A new variational autoencoder for top-n recommendations with implicit feedback, in: Proceedings of the 13th International Conference on Web Search and Data Mining, WSDM '20, Association for Computing

Machinery, New York, NY, USA, 2020, p. 528–536. URL: https://doi.org/10.1145/3336191.3371831. doi:10.1145/3336191.3371831.

[10] P. Li, S. A. M. Noah, H. M. Sarim, A survey on deep neural networks in collaborative filtering recommendation systems, arXiv preprint arXiv:2412.01378 (2024).

[11] V. Vančura, Scalable and explainable linear shallow autoencoders for collaborative filtering from industrial perspective, in: Proceedings of the 31st ACM Conference on User Modeling, Adaptation and Personalization, 2023, pp. 290–295.

[12] O. Jeunen, J. Van Balen, B. Goethals, Embarrassingly shallow auto-encoders for dynamic collaborative filtering, User Modeling and User-Adapted Interaction 32 (2022) 509–541.

[13] V. Vančura, P. Kordík, M. Straka, beeformer: Bridging the gap between semantic and interaction similarity in recommender systems, in: Proceedings of the 18th ACM Conference on Recommender Systems, 2024, pp. 1102–1107.

[14] M. Spišák, R. Bartyzal, A. Hoskovec, L. Peska, M. Tůma, Scalable approximate nonsymmetric autoencoder for collaborative filtering, in: Proceedings of the 17th ACM Conference on Recommender Systems, RecSys '23, Association for Computing Machinery, New York, NY, USA, 2023, p. 763–770. URL: https://doi.org/10.1145/3604915.3608827. doi:10.1145/3604915.3608827.

[15] F. Xu, H. Uszkoreit, Y. Du, W. Fan, D. Zhao, J. Zhu, Explainable ai: A brief survey on history, research areas, approaches and challenges, in: Natural language processing and Chinese computing: 8th cCF international conference, NLPCC 2019, dunhuang, China, October 9–14, 2019, proceedings, part II 8, Springer, 2019, pp. 563–574.

[16] L. Longo, R. Goebel, F. Lecue, P. Kieseberg, A. Holzinger, Explainable artificial intelligence: Concepts, applications, research challenges and visions, in: International cross-domain conference for machine learning and knowledge extraction, Springer, 2020, pp. 1–16.

[17] A. Abusitta, M. Q. Li, B. C. Fung, Survey on explainable ai: Techniques, challenges and open issues, Expert Systems with Applications 255 (2024) 124710.

[18] A. D. Selbst, J. Powles, Meaningful information and the right to explanation, International Data Privacy Law 7 (2017) 233–242. URL: https://doi.org/10.1093/idpl/ipx022. doi:10.1093/idpl/ipx022. arXiv:https://academic.oup.com/idpl/article-pdf/7/4/233/22923065/ipx022.pdf.

[19] M. Spišák, R. Bartyzal, A. Hoskovec, L. Peška, On interpretability of linear autoencoders, in: Proceedings of the 18th ACM Conference on Recommender Systems, RecSys '24, Association for Computing Machinery, New York, NY, USA, 2024, p. 975–980. URL: https://doi.org/10.1145/3640457.3688179. doi:10.1145/3640457.3688179.

[20] H. Cunningham, A. Ewart, L. Riggs, R. Huben, L. Sharkey, Sparse autoencoders find highly interpretable features in language models, 2023. URL: https://arxiv.org/abs/2309.08600. arXiv:2309.08600.

[21] T. Bricken, A. Templeton, J. Batson, B. Chen, A. Jermyn, T. Conerly, N. Turner, C. Anil, C. Denison, A. Askell, et al., Towards monosemanticity: Decomposing language models with dictionary learning, Transformer Circuits Thread 2 (2023).

[22] L. Bereska, E. Gavves, Mechanistic interpretability for ai safety – a review, 2024. URL: https://arxiv.org/abs/2404.14082. arXiv:2404.14082.

[23] L. Gao, T. D. la Tour, H. Tillman, G. Goh, R. Troll, A. Radford, I. Sutskever, J. Leike, J. Wu, Scaling and evaluating sparse autoencoders, 2024. URL: https://arxiv.org/abs/2406.04093. arXiv:2406.04093.

[24] A. Templeton, T. Conerly, J. Marcus, J. Lindsey, T. Bricken, B. Chen, A. Pearce, C. Citro, E. Ameisen, A. Jones, et al., Scaling monosemanticity: Extracting interpretable features from claude 3 sonnet. transformer circuits thread, 2024.

[25] S. Bostandjiev, J. O'Donovan, T. Höllerer, Tasteweights: a visual interactive hybrid recommender system, in: Proceedings of the Sixth ACM Conference on Recommender Systems, RecSys '12, Association for Computing Machinery, New York, NY, USA, 2012, p. 35–42. URL: https://doi.org/10.1145/2365952.2365964. doi:10.1145/2365952.2365964.

[26] F. M. Harper, F. Xu, H. Kaur, K. Condiff, S. Chang, L. Terveen, Putting users in control of their recommendations, in: Proceedings of the 9th ACM Conference on Recommender Systems, RecSys '15, Association for Computing Machinery, New York, NY, USA, 2015, p. 3–10. URL:

https://doi.org/10.1145/2792838.2800179. doi:10.1145/2792838.2800179.

[27] B. P. Knijnenburg, S. Bostandjiev, J. O'Donovan, A. Kobsa, Inspectability and control in social recommenders, in: Proceedings of the Sixth ACM Conference on Recommender Systems, RecSys '12, Association for Computing Machinery, New York, NY, USA, 2012, p. 43–50. URL: https://doi.org/10.1145/2365952.2365966. doi:10.1145/2365952.2365966.

[28] B. Epstein, R. Meir, Generalization bounds for unsupervised and semi-supervised learning with autoencoders, arXiv preprint arXiv:1902.01449 (2019).

[29] V. Vančura, R. Alves, P. Kasalický, P. Kordík, Scalable linear shallow autoencoder for collaborative filtering, in: Proceedings of the 16th ACM Conference on Recommender Systems, RecSys '22, Association for Computing Machinery, New York, NY, USA, 2022, p. 604–609. URL: https://doi.org/10.1145/3523227.3551482. doi:10.1145/3523227.3551482.

[30] M. Refinetti, S. Goldt, The dynamics of representation learning in shallow, non-linear autoencoders, in: International Conference on Machine Learning, PMLR, 2022, pp. 18499–18519.

[31] Z. Guo, G. Li, J. Li, C. Wang, S. Shi, Dualvae: Dual disentangled variational autoencoder for recommendation, in: Proceedings of the 2024 SIAM International Conference on Data Mining (SDM), SIAM, 2024, pp. 571–579.

[32] J. Šafařík, V. Vančura, P. Kordík, Repsys: Framework for interactive evaluation of recommender systems, in: Proceedings of the 16th ACM Conference on Recommender Systems, 2022, pp. 636–639.

[33] P. Covington, J. Adams, E. Sargin, Deep neural networks for youtube recommendations, in: Proceedings of the 10th ACM Conference on Recommender Systems, RecSys '16, Association for Computing Machinery, New York, NY, USA, 2016, p. 191–198. URL: https://doi.org/10.1145/2959100.2959190. doi:10.1145/2959100.2959190.

[34] A. N. Nikolakopoulos, X. Ning, C. Desrosiers, G. Karypis, Trust Your Neighbors: A Comprehensive Survey of Neighborhood-Based Methods for Recommender Systems, Springer US, New York, NY, 2022, pp. 39–89. URL: https://doi.org/10.1007/978-1-0716-2197-4_2. doi:10.1007/978-1-0716-2197-4_2.

[35] C. Gao, Y. Zheng, N. Li, Y. Li, Y. Qin, J. Piao, Y. Quan, J. Chang, D. Jin, X. He, Y. Li, A survey of graph neural networks for recommender systems: Challenges, methods, and directions, ACM Trans. Recomm. Syst. 1 (2023). URL: https://doi.org/10.1145/3568022. doi:10.1145/3568022.

[36] V. Vančura, P. Kasalickỳ, R. Alves, P. Kordík, Evaluating linear shallow autoencoders on large scale datasets, ACM Transactions on Recommender Systems (2025).

[37] H. Steck, Autoencoders that don't overfit towards the identity, in: Proceedings of the 34th International Conference on Neural Information Processing Systems, NIPS '20, Curran Associates Inc., Red Hook, NY, USA, 2020.

[38] S. Liang, Z. Pan, w. liu, J. Yin, M. de Rijke, A survey on variational autoencoders in recommender systems, ACM Comput. Surv. 56 (2024). URL: https://doi.org/10.1145/3663364. doi:10.1145/3663364.

[39] D. Kim, B. Suh, Enhancing vaes for collaborative filtering: flexible priors & gating mechanisms, in: Proceedings of the 13th ACM Conference on Recommender Systems, RecSys '19, Association for Computing Machinery, New York, NY, USA, 2019, p. 403–407. URL: https://doi.org/10.1145/3298689.3347015. doi:10.1145/3298689.3347015.

[40] A. Makhzani, B. Frey, k-sparse autoencoders, 2014. URL: https://arxiv.org/abs/1312.5663. arXiv:1312.5663.

[41] B. Bussmann, P. Leask, N. Nanda, Batchtopk sparse autoencoders, 2024. URL: https://arxiv.org/abs/2412.06410. arXiv:2412.06410.

[42] J. O'Donovan, B. Smyth, B. Gretarsson, S. Bostandjiev, T. Höllerer, Peerchooser: visual interactive recommendation, in: Proceedings of the SIGCHI Conference on Human Factors in Computing Systems, CHI '08, Association for Computing Machinery, New York, NY, USA, 2008, p. 1085–1088. URL: https://doi.org/10.1145/1357054.1357222. doi:10.1145/1357054.1357222.

[43] R. Sun, A. Akella, R. Kong, M. Zhou, J. A. Konstan, Interactive content diversity and user exploration in online movie recommenders: A field experiment, International Journal of Human–Com-

puter Interaction 0 (2023) 1–15. URL: https://doi.org/10.1080/10447318.2023.2262796. doi:10.1080/10447318.2023.2262796. arXiv:https://doi.org/10.1080/10447318.2023.2262796.

[44] D. Parra, P. Brusilovsky, User-controllable personalization: A case study with setfusion, International Journal of Human-Computer Studies 78 (2015) 43–67. URL: https://www.sciencedirect.com/science/article/pii/S1071581915000208. doi:https://doi.org/10.1016/j.ijhcs.2015.01.007.

[45] Y. Liang, M. C. Willemsen, Exploring the longitudinal effects of nudging on users' music genre exploration behavior and listening preferences, in: Proceedings of the 16th ACM Conference on Recommender Systems, RecSys '22, Association for Computing Machinery, New York, NY, USA, 2022, p. 3–13. URL: https://doi.org/10.1145/3523227.3546772. doi:10.1145/3523227.3546772.

[46] M. Millecamp, N. N. Htun, Y. Jin, K. Verbert, Controlling spotify recommendations: Effects of personal characteristics on music recommender user interfaces, in: Proceedings of the 26th Conference on User Modeling, Adaptation and Personalization, UMAP '18, Association for Computing Machinery, New York, NY, USA, 2018, p. 101–109. URL: https://doi.org/10.1145/3209219.3209223. doi:10.1145/3209219.3209223.

[47] J. D. Pauw, K. Ruymbeek, B. Goethals, Modelling users with item metadata for explainable and interactive recommendation, 2022. URL: https://arxiv.org/abs/2207.00350. arXiv:2207.00350.

[48] J. Kruse, K. Lindskow, S. Kalloori, M. Polignano, C. Pomo, A. Srivastava, A. Uppal, M. R. Andersen, J. Frellsen, Eb-nerd a large-scale dataset for news recommendation, in: Proceedings of the Recommender Systems Challenge 2024, RecSysChallenge '24, Association for Computing Machinery, New York, NY, USA, 2024, p. 1–11. URL: https://doi.org/10.1145/3687151.3687152. doi:10.1145/3687151.3687152.

[49] F. Lu, A. Dumitrache, D. Graus, Beyond optimizing for clicks: Incorporating editorial values in news recommendation, in: Proceedings of the 28th ACM Conference on User Modeling, Adaptation and Personalization, UMAP '20, Association for Computing Machinery, New York, NY, USA, 2020, p. 145–153. URL: https://doi.org/10.1145/3340631.3394864. doi:10.1145/3340631.3394864.

[50] Z. Gao, T. Shen, Z. Mai, M. R. Bouadjenek, I. Waller, A. Anderson, R. Bodkin, S. Sanner, Mitigating the filter bubble while maintaining relevance: Targeted diversification with vae-based recommender systems, in: Proceedings of the 45th International ACM SIGIR Conference on Research and Development in Information Retrieval, SIGIR '22, Association for Computing Machinery, New York, NY, USA, 2022, p. 2524–2531. URL: https://doi.org/10.1145/3477495.3531890. doi:10.1145/3477495.3531890.

[51] Y. Zhang, X. Chen, Explainable recommendation: A survey and new perspectives, Found. Trends Inf. Retr. 14 (2020) 1–101. URL: https://doi.org/10.1561/1500000066. doi:10.1561/1500000066.

[52] M. Guesmi, M. A. Chatti, S. Joarder, Q. U. Ain, C. Siepmann, H. Ghanbarzadeh, R. Alatrash, Justification vs. transparency: Why and how visual explanations in a scientific literature recommender system, 2023. URL: https://arxiv.org/abs/2305.17034. arXiv:2305.17034.

[53] N. Park, A. Kan, C. Faloutsos, X. L. Dong, J-recs: Principled and scalable recommendation justification, 2020. URL: https://arxiv.org/abs/2011.05928. arXiv:2011.05928.

[54] K. Balog, F. Radlinski, A. Petrov, Measuring the impact of explanation bias: A study of natural language justifications for recommender systems, in: Extended Abstracts of the 2023 CHI Conference on Human Factors in Computing Systems, CHI '23, ACM, 2023, p. 1–8. URL: http://dx.doi.org/10.1145/3544549.3585748. doi:10.1145/3544549.3585748.

[55] K. Muhammad, A. Lawlor, B. Smyth, On the use of opinionated explanations to rank and justify recommendations, in: The Twenty-Ninth International Flairs Conference, 2016.

[56] P. Symeonidis, A. Nanopoulos, Y. Manolopoulos, Providing justifications in recommender systems, IEEE Transactions on Systems, Man, and Cybernetics - Part A: Systems and Humans 38 (2008) 1262–1272. doi:10.1109/TSMCA.2008.2003969.

[57] N. Mauro, Z. F. Hu, L. Ardissono, Justification of recommender systems results: a service-based approach, User Modeling and User-Adapted Interaction 33 (2023) 643–685. URL: https://doi.org/10.1007/s11257-022-09345-8. doi:10.1007/s11257-022-09345-8.

[58] J. Wang, X. Zhang, W. Ma, M. Zhang, Interpret the internal states of recommendation model with sparse autoencoder, 2024. URL: https://arxiv.org/abs/2411.06112. arXiv:2411.06112.

[59] P. Ahmadov, M. Mansoury, Opening the black box: Interpretable remedies for popularity bias in recommender systems, in: Proceedings of the Nineteenth ACM Conference on Recommender Systems, RecSys '25, Association for Computing Machinery, New York, NY, USA, 2025, p. 1246–1250. URL: https://doi.org/10.1145/3705328.3759310. doi:10.1145/3705328.3759310.

[60] P. Dokoupil, L. Boratto, L. Peska, Calissta: Calibrated candidate retrieval using sparse autoencoders and top-k aggregation, in: Proceedings of the 34th ACM Conference on User Modeling, Adaptation and Personalization, UMAP '26, Association for Computing Machinery, New York, NY, USA, 2026, p. 345–350. URL: https://doi.org/10.1145/3774935.3806154. doi:10.1145/3774935.3806154.

[61] A. Klenitskiy, K. Polev, D. Denisova, A. Vasilev, D. Simakov, G. Gusev, Sparse autoencoders for sequential recommendation models: Interpretation and flexible control, 2025. URL: https://arxiv.org/abs/2507.12202. arXiv:2507.12202.

[62] T. Bertin-Mahieux, D. P. W. Ellis, B. Whitman, P. Lamere, The million song dataset, in: A. Klapuri, C. Leider (Eds.), Proceedings of the 12th International Society for Music Information Retrieval Conference, ISMIR 2011, Miami, Florida, USA, October 24-28, 2011, University of Miami, Miami, FL, USA, 2011, pp. 591–596. URL: http://ismir2011.ismir.net/papers/OS6-1.pdf.

[63] D. P. Kingma, J. Ba, Adam: A method for stochastic optimization, 2017. URL: https://arxiv.org/abs/1412.6980. arXiv:1412.6980.

[64] B. Bussmann, N. Nabeshima, A. Karvonen, N. Nanda, Learning multi-level features with matryoshka sparse autoencoders, in: A. Singh, M. Fazel, D. Hsu, S. Lacoste-Julien, F. Berkenkamp, T. Maharaj, K. Wagstaff, J. Zhu (Eds.), Proceedings of the 42nd International Conference on Machine Learning, volume 267 of *Proceedings of Machine Learning Research*, PMLR, 2025, pp. 6077–6101. URL: https://proceedings.mlr.press/v267/bussmann25a.html.