

# Discovering Personalized Semantics for Soft Attributes in Recommender Systems using Concept Activation Vectors\*

CHRISTINA GÖPFERT, Bielefeld University, Germany

YINLAM CHOW<sup>†</sup>, Google Research, USA

CHIH-WEI HSU, Google Research, USA

IVAN VENDROV<sup>‡</sup>, Omni Labs, USA

TYLER LU<sup>†</sup>, Talka, Inc., USA

DEEPAK RAMACHANDRAN, Google Research, USA

CRAIG BOUTILIER, Google Research, USA

Interactive *recommender systems* (RSs) have emerged as a promising paradigm to overcome the limitations of the primitive user feedback used by traditional RSs (e.g., clicks, item consumption, ratings). **They allow users to express intent, preferences, constraints, and contexts in a richer fashion, often using natural language; yet more research is needed to find the most effective ways to use this feedback.** One challenge is **inferring a user’s semantic intent from the open-ended terms (attributes or tags) used to describe a desired item, and using it to refine recommendation results.** Leveraging *concept activation vectors* (CAVs) [22], a recently developed approach for model interpretability in machine learning, we develop a framework to learn a representation that captures the semantics of such attributes and connects them to user preferences and behaviors in RSs. One novel feature of our approach is its ability to distinguish objective and *subjective* attributes (both subjectivity of *degree* and of *sense*), and associate *different senses* of subjective attributes with different users. We demonstrate on both synthetic and real-world datasets that our CAV representation not only accurately interprets users’ subjective semantics, but can also be used to improve recommendations through *interactive item critiquing*.

## 1 INTRODUCTION

The ubiquity of *recommender systems* (RSs) in mediating the discovery and consumption of content, products and services has driven user demands of RSs, and increased the expectation that RSs should more deeply understand their needs and preferences. *Conversational recommenders* [2] have emerged as a promising paradigm to meet such expectations—they can overcome the primitive user feedback admitted by traditional RSs (e.g., simple queries, clicks, item consumption, ratings, purchases), and allow users to express their intent, preferences, constraints and contexts in a richer fashion through the use of natural-language-based interaction (e.g., faceted search, or more open-ended dialogue). However, interpreting such interactions requires grounding the intended semantics of the user with respect to the RS’s underlying model of user preferences. For example, if a user expresses a desire for a “funny” movie, this must be translated into an actionable representation of her preferences over the target movie corpus.

When the set of item attributes is well-defined and known *a priori* (e.g., as with the item attributes in a product catalog), existing techniques such as *faceted search* [25, 49] or *example critiquing* [12, 14] can be used directly. But

\*This is an extended version of a paper to appear at WWW-22: The Web Conference 2022, April 2022, Lyon, France.

<sup>†</sup>Contact author.

<sup>‡</sup>Work conducted while at Google Research.

Authors’ addresses: Christina Göpfert, Bielefeld University, Universitätsstr. 25, Bielefeld, Germany, 33613, chgopfert@gmail.com; Yinlam Chow, Google Research, 1600 Amphitheatre Parkway, Mountain View, CA, USA, 94043, yinlamchow@google.com; Chih-wei Hsu, Google Research, 1600 Amphitheatre Parkway, Mountain View, CA, USA, 94043, cwhsu@google.com; Ivan Vendrov, Omni Labs, 821 Folsom St., San Francisco, CA, USA, 94107, ivendrov@gmail.com; Tyler Lu, Talka, Inc., San Francisco, CA, USA, tyler.lu@gmail.com; Deepak Ramachandran, Google Research, 1600 Amphitheatre Parkway, Mountain View, CA, USA, 94043, ramachandrand@google.com; Craig Boutilier, Google Research, 1600 Amphitheatre Parkway, Mountain View, CA, USA, 94043, cboutilier@google.com.

often item attributes are *soft* [1]: there is no definitive “ground truth” source associating such soft attributes with items; the attributes themselves may have imprecise interpretations; and they may be *subjective* in nature (i.e., different users may interpret them differently). For instance, in *collaborative filtering* (CF) tasks such as movie recommendation, side information about movie attributes like ‘funny,’ ‘thought-provoking,’ or ‘violent’ is often available, but it is often ancillary, derived from sparse, noisy user comments, reviews, or tags); and, users may disagree on which movies they consider to be ‘violent’ (or which are ‘too violent’ for them). Using such attributes for item search or recommendation is thus challenging, since which items exhibit such attributes usually has to be predicted in the absence of a ground truth, and these predictions may need to depend on a specific user’s usage or semantics.

Recent work has attempted to *jointly* learn the semantics of soft attributes with user preferences [28, 36, 51]. In this work, we adopt a different perspective: we treat the recommendation task as primary, using standard CF models for RSs; and we *infer the semantics of soft attributes using the representation learned by the RS model itself* [15, 43]. Our approach has four key advantages: (1) Model capacity is directed to predicting user-item preferences without side information, which often does not improve RS performance. (2) The RS model can easily accommodate new attributes without retraining should new sources of tags, keywords or phrases emerge. (3) It offers a means to test whether specific soft attributes are *relevant* to predicting user preferences, and to focus attention on attributes most relevant to capturing a user’s intent (e.g., when explaining recommendations, eliciting preferences, or suggesting critiques). (4) One can learn soft attribute/tag semantics with relatively small amounts of labelled data, in the spirit of pre-training and few-shot learning.

Concretely, we assume we are given: (i) a CF-style model (e.g., probabilistic matrix factorization or dual encoder) which embeds items and users in a latent space based on user-item ratings; and (ii) a (small) set of *tags* (i.e., soft attribute labels) provided by a *subset of users* for a *subset of items*. We develop methods that associate with each item the degree to which it exhibits a soft attribute, thus determining that attribute’s semantics. We do this by applying *concept activation vectors* (CAVs) [22]—a recent method developed for ML interpretability—to the CF model to detect whether it *learned a representation of the attribute*. The projection of this CAV in embedding space provides a (local) *directional semantics* for the attribute that can then be applied to items (and users). Moreover, the technique can be used to identify the *subjective nature* of an attribute, specifically, whether different users have different meanings (or tag *senses*) in mind when using that tag. Such a *personalized semantics* for subjective attributes can be vital to the sound interpretation of a user’s true intent when trying to assess her preferences.

Our key contributions are as follows: (i) We propose a novel framework using CAVs to identify the semantics of soft attributes relative to preference prediction or behavioral models in RSs *without requiring co-training* of semantics and preference models. (ii) We develop methods to distinguish *objective* and *subjective* attributes (both subjectivity of *degree* and of *sense*) and associate different senses of subjective attributes with different users. (iii) We investigate a simple method that leverages this semantics to elicit preferences via *example critiquing*. Experiments on both synthetic and real-world data show the efficacy of our methods. The sequel is organized as follows. In Sec. 2 we outline our problem setup and discuss related work. In Sec. 3, we use CAVs to identify the semantics of (objective) soft attributes by leveraging CF user/item representations (linear and nonlinear), while in Sec. 4 we extend the approach to handle subjective attributes and identify a user’s interpretation for different tags. In Sec. 5 we illustrate how CAVs can be used to support example critiquing using such soft attributes. We test our methods on both synthetic data and the MovieLens20M dataset [18]. Additional material elaborating on some of the details summarized in these sections can be found in the appendix.

## 2 PROBLEM FORMULATION

We first outline our problem formulation and data assumptions, then briefly discuss related work.

**User-item Ratings.** We assume a standard collaborative filtering (CF) task: users  $\mathcal{U}$  offer ratings of items  $\mathcal{I}$ , with  $r_{u,i} \in \mathcal{R}$  denoting the rating of user  $u \in \mathcal{U}$  for item  $i \in \mathcal{I}$ , where  $\mathcal{R}$  is the set of possible ratings (e.g., 1–5 stars). Let  $n = |\mathcal{U}|$ ,  $m = |\mathcal{I}|$ , and  $\mathbf{R}$  denote the  $m \times n$  (usually sparse) ratings matrix, with  $r_{u,i} = 0$  denoting that user  $u$  has not rated item  $i$ . Let  $R = \{(u, i) : r_{u,i} \neq 0\}$  be the set of user-item pairs for which a rating has been given.

**Preference Predictions.** We assume a CF method has been applied to  $\mathbf{R}$  to construct *user and item embeddings*,  $\phi_U : \mathcal{U} \mapsto \mathbb{R}^d$  and  $\phi_I : \mathcal{I} \mapsto \mathbb{R}^d$ , respectively, such that the model’s predicted (expected) rating is  $\hat{r}_{i,u} = \phi_U(u)^\top \phi_I(i)$ . We let  $X \subseteq \mathbb{R}^d$  generically denote the embedding space. Suitable methods include matrix factorization [44] or certain forms of neural CF [3, 52]. For concreteness, we assume a *two-tower model* (or *dual encoder*) in which users and items are passed through separate (but co-trained) deep neural nets (DNNs),  $N_U$  and  $N_I$ , to produce their respective vector embeddings  $\phi_U(u)$  and  $\phi_I(i)$ , which are combined via dot product to predict user-item affinity  $\hat{r}_{i,u}$  [52, 53]. We can view  $\phi_I(i)$  as a (learned) *latent feature vector* characterizing item  $i$  and  $\phi_U(u)$  as parameterizing user  $u$ ’s estimated *utility (or preference) function* over these features. By construction, user utility is linear with respect to these latent item features (a limitation we discuss in Sec. 4.2). While we focus on this specific two-tower architecture, our methods apply directly to matrix factorization models and should extend to more general neural CF approaches [19].

**Soft Attributes & Tags.** CF methods are often used to predict user-item affinity in *content* RSs (e.g., for movies, music, news, etc.) because *user rating or consumption behavior* is generally far more predictive of user preferences than *hard attributes*, i.e., typical “objective,” known item features (e.g., genre, artist, director) [24]. Despite this, users often *describe* items using *soft attributes* [1], features that are not part of an agreed-upon, formal item specification. For example, movies might be described using terms like ‘funny,’ ‘thought-provoking,’ ‘violent,’ ‘cheesy,’ etc. We call such terms *tags* rather than attributes, since they are neither applied universally to all items, nor are they applied by all users. Moreover, as we discuss below, these tags may be applied *subjectively* in the sense that users may disagree on the items to which, or the degree to which, these tags apply. While some RSs allow users to specify tags directly (see below), tags may also be extracted from user descriptions, reviews or other data sources.

A number of RSs support user-supplied tags (see, e.g., the MovieLens dataset [18] we use below). For simplicity, we assume a set of  $k$  canonical tags  $\mathcal{T}$  that users may adopt to describe items.<sup>1</sup> We also assume that tags are used “propositionally” (a user chooses to apply a tag or not) though the underlying attributes to which they refer may be ordinal or cardinal (e.g., a tag ‘violent’ may refer to some degree of ‘violence’).<sup>2</sup> *Tag data* comprises a  $m \times n \times k$  tensor  $\mathbf{T}$  where  $t_{u,i,g} = 1$  if user  $u$  applies tag  $g$  to item  $i$ , and 0 otherwise. Let  $T = \{(u, i, g) : t_{u,i,g} = 1\}$  and  $T_g = \{(u, i) : t_{u,i,g} = 1\}$  be the set of user-item pairs for which tag  $g$  has been applied. Tags are usually strictly sparser than ratings, so we assume  $T_g \subseteq R$  for all  $g \leq k$ . Note that  $u$  may apply multiple tags to the same item. Let  $T_u \subseteq \mathcal{I}$  be the set of items tagged by  $u$  (using any tag),  $T_{u,g}$  be those tagged by  $u$  with  $g$  specifically, and  $T_{u,\bar{g}} = T_u \setminus T_{u,g}$  be those tagged by  $u$  but not with  $g$ . Our tag-data setup is similar to that used in work on tag recommenders [15, 17, 28].

**Elicitation & Critiquing.** CF models, in isolation, are ill-suited to RSs that aim to naturally interact with users to refine knowledge of their preferences. A CF-based RS can actively elicit new ratings or support “more like this” statements at the *item level* [9, 54], but the embedding representation of items, since it is not generally interpretable, does not lend

<sup>1</sup>We set aside the question of aggregating potentially diverse tags into a set of covering prototypes.

<sup>2</sup>Our techniques can be extended in a straightforward way to Boolean (positive and negative application), ordinal or cardinal tags.

itself to such *attribute-based interaction*. Tags can help alleviate this limitation, and offer a compelling mechanism to help users navigate item space. A number of preference elicitation and example critiquing methods have been developed that use *hard attributes* (see Related Work below). The challenge in content RSs is that tags typically correspond to *soft attributes* and are often *subjective* in nature. For example, if a user critiques a movie recommendation by asking for something “more thought-provoking” or “less violent,” standard techniques for adjusting the RS’s model of the user’s preference cannot be applied unless we have a concrete semantics that relates the corresponding tag to specific items in a way that better reflects her preferences.

**Concept Activation Vectors.** Research on interpretable representations tries to overcome the fact that modern ML models—and deep neural networks (DNNs) in particular—usually learn complex, non-transparent representations of concepts [22, 45]. The *testing CAVs* (TCAV) framework [22] is one such mechanism that tries to find a correspondence between the “state” of a model (e.g., input features, DNN activation patterns) and human-interpretable concepts. For instance, suppose a DNN has been trained to classify animals in images. Using only a small set of images with positive and negative examples of some concept (e.g., “objects with stripes”), TCAV is used to test whether the DNN has learned a representation of that concept in the form of a vector of activation (CAV) that correlates with its presence. Moreover, using the derivative of the classifiers output with respect to the CAV’s direction, it measures how important that concept is to the classifier’s predictions (e.g., how sensitive a “zebra” classification is to the presence of stripes in an image). In our setting, we use CAVs to translate between latent item representations learned by a CF model and the soft attributes that users adopt to describe items and preferences (see below and Appendix A.1).

For RSs, we use CAVs to translate between the latent item representations learned by a CF model and the soft attributes users adopt to describe items and preferences. We detail the adaptation of key CAV concepts to RSs in Section 3 below.

**Related Work.** A number of methods exist for finding the semantics of tags and attributes in RSs using tag data [17, 28] or reviews [31]. While some learn semantics jointly with ratings prediction, others build attribute models “on top of” a latent factor model built for ratings prediction as we do here. Most related to our approach is that of Gantner et al. [17], who learn semantics for tags or explicit item features as a linear combination of latent features using  $k$ -nearest neighbors or linear regression, where the latent feature are those of a BPR model [42] (though the technique applies more broadly). Cohen et al. [15] extend this model to incorporate uncertainty and active exploration. This work is proposed as a means for solving the cold-start problem. Our work differs primarily in its ability to handle nonlinear representations and subjectivity, and its focus on conversational and critiquing RSs. More generally, the broader literature on tag recommendation bears some connection to our work [26, 27, 29, 43] by modeling the relationship between users, items and tags.

One of our main motivations is to make soft and subjective attributes interpretable for use in critiquing [14], faceted search [49], and preference elicitation for RSs. These areas include a variety of techniques. We do not survey these here since we do not propose novel elicitation or critiquing methods—our focus is instead on how to incorporate soft, subjective attributes into such methods, which we exemplify using a fairly “vanilla” critiquing model. However, Radlinski et al. [39] develop a methodology (and dataset) for connecting the preferences of users with their use of soft attributes in conversational RSs. While little work in elicitation for RSs addresses subjectivity, one exception is [7, 8] who consider “definitional” subjectivity defined using personalized logical concepts, which is a very different notion from ours.

Subjectivity has been studied in natural language processing and psycholinguistics. Welch et al. [50] show that training personalized embeddings for certain classes of words (e.g., adverbs, social words, words relating to cognitive processes) increases language modeling performance. Their methodology could be used for a more granular analysis of which words and phrases are most subjective, which can then be used as a prior for analyzing subjectivity in recommender settings. Prototype theory [37] is a popular methodology where subjectivity is related to the similarity of an item to an idealized exemplar. Geometrically, this could be considered analogous to a “ball” semantics of subjectivity while CAVs encapsulate a “line” semantics.

### 3 FINDING RELEVANT SOFT ATTRIBUTES

We develop a method for identifying the semantics of *relevant soft attributes* with respect to the item embedding representation learned by our CF method. Assume a CF model  $\Phi = (\phi_U, \phi_I)$  trained on ratings data  $\mathbf{R}$  with additional tag data  $\mathbf{T}$ . As above, we’ll often assume that  $\phi_U$  and  $\phi_I$  are realized by DNNs  $N_U$  and  $N_I$ , respectively. We use CAVs [22] to discover whether the CF model has learned an implicit representation of a soft attribute corresponding to a given tag. If so, that representation can be used to support example critiquing, preference elicitation or item-space navigation (Sec. 5). Critically, *we do not use the tag data* when training the CF model—this is akin to work that builds attribute models on top of embeddings to address the cold-start problem [15, 43] (and contrasts with models that jointly train attribute models [28, 51]). Our hypothesis is that *if a tag is useful for understanding user preferences across a broad swath of the population, the CF model will have learned a representation of the corresponding soft attribute*. The converse is that if no such representation (or CAV) is uncovered, this soft attribute is of limited use for users expressing their preferences.

Our approach has a number of advantages: (1) the RS model can be developed/trained/used without a pre-commitment to a specific attribute vocabulary—new attributes can be added as needed without retraining the RS model itself; (2) RS model capacity is focused on the core task of preference prediction and recommendation; and (3) our method can be used to assess the relevance and importance of specific attributes for preference elicitation or critiquing. In this section, we set aside subjectivity, deferring its treatment to Sec. 4 (though some implicit *subjectivity of degree* is accounted for here). We explicate our concepts by first assuming that soft attributes are linear in user/item embedding space (Section 3.1), then turn to nonlinear representations (Section 3.2). We evaluate our methods empirically in Section 3.3.

**Mapping CAV Notions to RSs.** Before formalizing our approach, we briefly map (and adapt) key CAV concepts to our recommender setting by drawing an analogy with the image classification setting used to explicate CAVs by Kim et al. [22] (and briefly described above). Our DNN CF model  $\Phi = (\phi_U, \phi_I)$  is trained on user-item ratings, similar to a multi-class image classifier trained on labeled images. A soft attribute or tag  $g$  (say, ‘violent’) is analogous to some image feature (e.g., ‘stripes’), and our aim is to use CAVs to determine if the *item network*  $N_I$  has learned some representation of this tag. Just as in the image setting, where some small set of positive (with stripes) and negative (non-striped) images is used to attempt to identify a CAV, we use a *small* set of positive (tagged) and negative (untagged) items for the same purposes, though we must account for both variability and inconsistency in the tags applied by different users.<sup>3</sup> We refer to Appendix A.1 for a concise list of key concepts and notations. Fig. 1 offers a graphical depiction our use of CAVs for RSs (including in example critiquing, see Sec. 5).

<sup>3</sup>While not our aim in this work, we can also use CAVs to test the sensitivity of various predictions to the presence of this soft attribute: by analogy with testing the sensitivity of a ‘zebra’ classification to the presence of stripes, we can test the sensitivity of a user’s item rating to the item’s (degree of), say, violence. In the CF setting however, this sensitivity will differ across the user population.

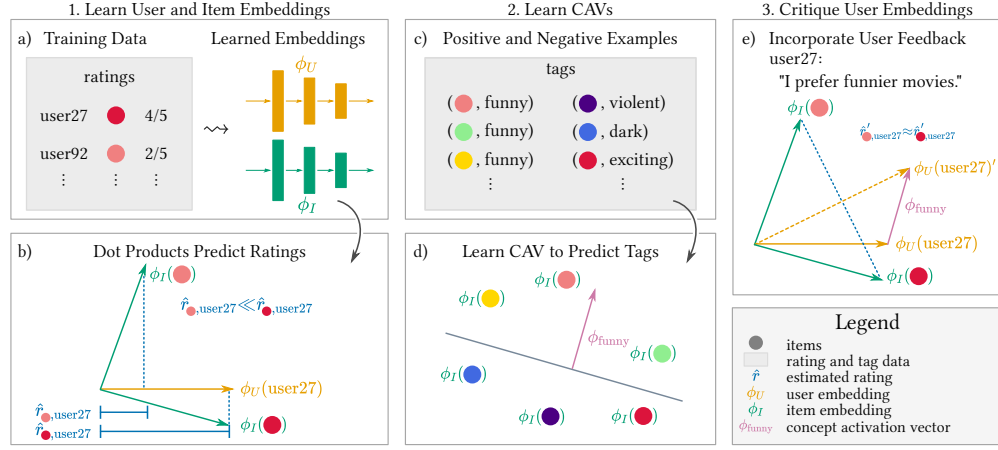


Fig. 1. An overview of the CF, CAV learning and critiquing setup used in our work. a) Learn user  $\phi_U$  and item embeddings  $\phi_I$  from ratings data. b) User-item affinity  $\hat{r}_{i,u}$  is predicted using dot products of user-item embeddings. c) To learn a CAV for the concept ‘funny,’ gather positive and negative examples, e.g., from tag data, and use them to d) learn a CAV in item-embedding space. (We explore several methods for CAV training.) e) In one use case, we use the CAV to update a user’s embedding given her item-attribute feedback. Figure inspired by [22].

### 3.1 Linear Attributes

We adapt CAVs to test whether a CF model has learned a representation of a soft attribute corresponding to a tag. We first illustrate our approach by testing whether the *embedding space itself* contains a *linear* representation of the tag’s underlying attribute (i.e., an attribute that is linear in item embedding features  $\phi_I$ ). We generalize this linear model (whose weaknesses we detail below) in Sec. 3.2. Given CF model  $\Phi$ , each  $i \in \mathcal{I}$  is represented by its embedding  $\phi_I(i) \in X$ . For any user  $u$ , the representations  $\phi_I(i)$  of items  $i \in T_{u,g}$  to which she has applied tag  $g$  are treated as positive examples of the underlying concept (say, violent movies), while  $\phi_I(i)$ ,  $i \in T_{u,\bar{g}}$  are used as negative examples.<sup>4</sup> Taking examples over all  $u \in \mathcal{U}$ , we train a linear classifier in one of two ways.

Our first model assumes each  $u$  uses  $g$  in roughly the same way, with positive instances given by the multi-set  $\cup_{\mathcal{U}}\{\phi_I(i) : i \in T_{u,g}\}$ , and negatives by  $\cup_{\mathcal{U}}\{\phi_I(i) : i \in T_{u,\bar{g}}\}$ . Since positive tag examples are often sparse, we use *negative sampling* to manage class imbalance [33, 52]. Let  $D_g$  be the induced “global” (cross-user) data set. We train a logistic regressor  $\phi_g$ , where  $P(g(i); \phi_g) = \sigma(\phi_g^\top \phi_I(i))$  is the predicted probability that  $i$  “satisfies”  $g$ , using (regularized) logistic loss (and labels  $y \in \{+1, -1\}$ ):

$$\mathcal{L}(\phi_g; D_g) = \sum_{(i,y) \in D_g} \log(1 + e^{-y\phi_g^\top \phi_I(i)}) + \frac{\lambda}{2} \phi_g^\top \phi_g. \quad (1)$$

We refer to this method as binary logistic regression for CAVs.

If two users disagree on the application of tag  $g$  to some item, this global classifier treats it as label noise. An alternative explanation for such a discrepancy is that they agree on the “direction” of  $g$ , but disagree on the “degree” to which item  $i$  exhibits  $g$ ’s underlying soft attribute. For example, two users may agree on which movie, for *any* pair of movies, is more violent, but have different *thresholds* or tolerances when applying the tag (e.g., disagree on “how

<sup>4</sup>These negatives are “implicit,” but plausible, since  $u$  has otherwise tagged these items. Explicit negatives are more informative, and what follows can be generalized easily to this case. Potential bias that may arise since tags may not be “missing at random” [30], but we defer this issue to future work.



violent is violent”)—both may agree movie  $i$  is more violent than movie  $j$ , but one may consider neither to reach the threshold needed to label it truly ‘violent,’ while the other may consider  $i$  and  $j$  to both warrant that label. Our second model accounts for this by treating each  $u$  as generating *pairwise comparisons*  $D_u = \{\phi_I(i) >_g \phi_I(j) : i \in T_{u,g}, j \in T_{u,\bar{g}}\}$ , drawn from an underlying ranking. We use a *per-user* pairwise ranking loss to generate a *regressor* over  $X$  specifying the *degree* to which items exhibit the soft attribute:

$$\mathcal{L}(\phi_g; (D_u)_{u \in \mathcal{U}}) = \sum_u \sum_{\substack{i \in T_{u,g} \\ j \in T_{u,\bar{g}}}} \log(1 + e^{-\phi_g^\top (\phi_I(j) - \phi_I(i))}) + \frac{\lambda}{2} \phi_g^\top \phi_g. \quad (2)$$

This logistic pairwise loss is the same as that used in RankNet [10], hence we refer to this method as RankNet for CAVs.

<sup>5</sup> The regressor  $\phi_g$  obtained serves as our CAV. Notice that  $\phi_g$  is linear in the learned item embedding features  $\phi_I$ .

Given a CAV  $\phi_g$ , the degree to which an item  $i$  satisfies the induced attribute is given by the score  $\phi_g(i) = \phi_g^\top \phi_I(i)$ . The *quality*  $Q(\phi_g; D)$  of a CAV on dataset  $D$  is the fraction of the tag applications that it orders “correctly,” i.e., if  $i \in T_{u,g}, j \in T_{u,\bar{g}}$ , then  $\phi_g(i) \geq \phi_g(j)$ . This is closely related to the pairwise loss above and is a key metric we use to evaluate CAVs, and can be used as a measure of a CAV’s “usefulness” for, say preference elicitation, as we discuss below.

### 3.2 Nonlinear Attributes

A limitation of the linear approach is that if a CAV for tag  $g$  is linear in the latent embedding space  $X \subseteq \mathbb{R}^d$ , every user’s utility for  $g$  is also linear in  $X$ . For example, if the CAV for the tag ‘violent’ is linear in  $X$ , it implies any user’s preference would be such that she prefers either maximally or minimally violent movies—she cannot prefer movies that are only “somewhat violent.” For many soft attributes, this will not be the case—real-world preferences are often nonlinear (e.g., saturating [16]) and even non-monotone (e.g., single-peaked [35, 41]) with respect to many natural attributes. Such attributes are unlikely to be adequately represented linearly in embedding space  $X$ . Fortunately, CAVs can also be applied to nonlinear DNN representations.

We assume a two-tower/dual-encoder model and extract CAVs from hidden layers of the item DNN  $N_I$ . Following Kim et al. [22], we assume that relevant concepts, if learned, can be uncovered within a single hidden layer of the (trained) deep CF model.<sup>6</sup> Given positive and negative examples as in Sec. 3.1, we use activation  $\phi_{I,\ell}(i)$  of the  $\ell$ th layer of  $N_I$  as training input instead of item embedding  $\Phi_I(i)$ . Otherwise, the regressor is trained as above, and the validity of the induced CAV can be tested in the same fashion as well.

The result is a regressor  $\phi_{g,\ell}$  that can be applied to an item’s representation in the intermediate “activation space”  $X_I^\ell$ , where  $\phi_{g,\ell}^\top \phi_{I,\ell}(i)$  captures the degree to which  $i$  satisfies the induced attribute. The projection of this regressor through the last  $L - \ell$  layers of  $N_I$  generates a (nonlinear) manifold in embedding space  $X$ . This gives considerably more flexibility to the relation between user utilities and soft attributes. In our nonlinear CAV experiments, we treat  $\ell$  as a tunable hyper-parameter, and our reported results are based on the best CAVs extracted from these layers.

### 3.3 Empirical Assessment of CAV Quality

We first evaluate our approach on synthetic data, which allows strict control over the generative process and direct access to ground truth. We then test it on real-world data. For linear soft attributes, we train a CF model  $\Phi = (\phi_U, \phi_I)$

<sup>5</sup>Other loss functions (e.g., LambdaRank [11], exponential, hinge [13]) may also be useful as we show empirically in the case of LambdaRank.

<sup>6</sup>Kim et al. [22] argue that TCAV is typically robust to this choice. We can apply TCAV to multiple layers simultaneously as well (see below).

using *weighted alternating least squares* (WALS) [20], with the following regularized objective:

$$(\phi_U^*, \phi_I^*) \in \arg \min \sum_{u,i} c_{u,i} (\hat{r}_{u,i} - r_{u,i})^2 + \kappa (\|\phi_U\|^2 + \|\phi_I\|^2). \quad (3)$$

Here  $c_{u,i}$  is a confidence weight for the predicted rating  $\hat{r}_{u,i} = \phi_U^\top(u; \theta_U) \phi_I(i; \theta_I)$ , and  $\kappa > 0$  is a regularization parameter. We select embeddings  $(\phi_U^*, \phi_I^*)$  using validation loss and use an item-oriented confidence weight  $c_{u,i} \propto m - \sum_u r_{u,i}$  (i.e., lower weight for less-frequently or lower-rated items). For nonlinear attributes, we train two-tower DNN embedding model with SGD/Adam [23]. Further details on synthetic data generation (A.2), datasets (A.3) and training methods (A.4) are provided in the appendix.

**Synthetic Data.** To construct synthetic data, a generative model outputs both user-item ratings  $\mathbf{R}$  and tag data  $\mathbf{T}$  for  $n = 25,000$  users and  $m = 10,000$  items. Users and items are each represented by  $d = 25$ -dimensional embedding vectors, sampled from pre-defined mixture distributions to induce correlation in the data. For linear utility, user ratings are generated by first sampling items (giving a sparse ratings matrix  $\mathbf{R}$ ) and then their ratings (noise added to the user/item dot product). For nonlinear utility, utility is the sum of nonlinear sub-functions (one per dimension) peaked at some (random) point and dropping as the item moves away from that peak. In both cases, users are more likely to rate items with higher utility.

To generate tags, five of the 25 latent item dimensions are treated as user-interpretable or “taggable,” each with a different tag. Intuitively, these correspond to dimensions in latent space that are both “perceivable” and “interpretable” by users. Note, however, that the CF model does not have access to the structure or details of the underlying generative process, only seeing the ratings and tag data so-generated. Each  $u$  has a random *propensity to tag*, which influences the probability of tagging a rated item. Users are also more likely to tag higher-rated items. For a given tag  $g$ , there is a *fixed* (non-subjective) threshold  $\tau_g$  against which  $u$  evaluates an item to be tagged, and noisily applies  $g$  if that threshold is met.

**CAV Accuracy.** We evaluate CAV accuracy—how well the learned regressors capture the underlying soft attribute—using standard metrics of prediction quality with respect to user tag usage on held-out test data. The synthetic model also allows evaluation relative to the *ground-truth item representations and attribute levels of each tag* since we have access (for purposes of evaluation) to the true latent attribute values of each item as generated by the model. We evaluate three training methods, binary logistic regression, RankNet [10], and LambdaRank [11]. We use the following metrics to measure the accuracy of CAVs on the test set: (i) *Accur.*, the mean accuracy of the logistic model or *quality*  $Q(\phi_g; D)$  of the ranking model; and (ii) *Sprmn*, which is *Spearman rank correlation coefficient* between predicted and ground-truth attribute values.<sup>7</sup> The latter compares the *ranking* of all items by their *ground-truth* attribute value with the ranking induced by the attribute’s *learned CAV*. (This metric applies equally to “nonlinear” latent attributes, since the CAV still provides a precise ordering of items according to the attribute.)

**Synthetic Results.** Table 1 shows the performance of the CAVs on synthetic data for three settings: (i) user utility is linear; (ii) utility is nonlinear but we train linear CAVs (Lin-Emb); and (iii) utility is nonlinear and we train nonlinear CAVs (NL-Emb). Results are averaged over the five tags. We see that the CAVs predict user tagging behaviors (Accur) reasonably accurately, and reliably order test items according to the ground truth value of the underlying soft attribute (Sprmn), despite the noise in the tagging process. We bold the best value among the three settings. The ranking methods, RankNet and LambdaRank, dominate logistic regression with respect to both Accur and Sprm, which suggests that

<sup>7</sup>We also computed precision and recall results, but these are omitted since they simply corroborate the findings generated by these metrics.



|            | Linear       |              | NonLin, Lin-Emb |              | Nonlin, NL-Emb |              |
|------------|--------------|--------------|-----------------|--------------|----------------|--------------|
|            | Accur.       | Sprmn        | Accur.          | Sprmn        | Accur.         | Sprmn        |
| Log. Regr. | 0.906        | 0.569        | 0.889           | 0.565        | 0.922          | 0.577        |
| RankNet    | <b>0.968</b> | 0.674        | 0.943           | <b>0.670</b> | <b>0.978</b>   | <b>0.686</b> |
| LambdaNet  | 0.961        | <b>0.679</b> | <b>0.947</b>    | 0.666        | 0.974          | 0.680        |
| PITF       | 0.683        | 0.056        | 0.707           | 0.070        | N/A            | N/A          |

Table 1. CAV Evaluation, Synthetic Data (Non-subjective)

|            | Lin-Emb      | NL-Emb       |
|------------|--------------|--------------|
|            | Accur.       | Accur.       |
| Log. Regr. | 0.727        | 0.745        |
| RankNet    | 0.803        | <b>0.820</b> |
| LambdaNet  | <b>0.804</b> | 0.818        |
| PITF       | 0.715        | N/A          |

Table 2. CAV Evaluation, MovieLens

accounting for variation in user tagging behavior is important (we discuss this further when describing subjective results in the next section). For nonlinear utilities, we also compare the best “linear” CAV (extractable from the output embedding) with that the best nonlinear CAV (extractable from DNN hidden layers). Nonlinear CAVs outperform their linear counterparts, demonstrating the value of seeking nonlinear (or “distributed”) attribute representations within the DNN, and the power of TCAV to help interpret them. We also include a baseline tag recommender *PITF* (*pairwise interaction tensor factorization*) [43], which uses tensor decomposition to model pairwise interactions between users, items and tags (see more details in the appendix)—note that it produces linear representations. Its tag prediction accuracy is worse than that of the CAV approaches.

**MovieLens Results.** We also evaluate our methods on the more realistic MovieLens20M dataset [18], which comprises 20M movie ratings (1–5 scale with half-star increments) and 465K tag-instances applied to 27K movies by 138K users. Tags are user-generated strings that describe movie attributes (e.g., genres like ‘sci-fi,’ ‘comedy,’ ‘action;’ descriptions like ‘emotional,’ ‘funny,’ ‘atmospheric;’ and themes like ‘zombies,’ ‘world war 2,’ ‘cyberpunk’). We split rating and tag data into train and test sets such that *all examples* for any specific user-item pair are present in exactly one of these subsets (i.e., if  $u$  rated and tagged  $i$ , the rating and all tags  $g$  applied by  $u$  to  $i$  are in one of train or test). We generate  $d = 50$ -dimensional user and item embeddings using WALS if linear, two-tower DNNs if nonlinear.

Positive examples for tag  $g$  are user-item pairs to which  $g$  has been applied; negatives are those tagged by that user, but not with  $g$ . Given the sparseness of tags—of 30745 unique tags, more than 28000 are applied to fewer than 10 unique movies—we train CAVs only for tags that are among the 250 most frequently applied both by unique user and to unique items, which results in 164 tags. Table 2 shows test accuracy for linear and nonlinear CAVs: the ranking methods outperform logistic regression, which again hints at some subjectivity (see next section). While we cannot measure Spearman correlation (since we have no ground truth ranking) nor control the form of user utility, we see that nonlinear CAVs perform slightly better than linear CAVs, suggesting that user preferences for some MovieLens tags are nonlinear in their embedding-space representation. As above, the CAV methods consistently outperform PITF.

#### 4 IDENTIFYING SUBJECTIVE ATTRIBUTES

If users largely agree on the usage of tags, it is reasonable to treat the semantics of a tag as a *single* soft attribute or CAV as we do above. But in many cases, different users may have different “senses” in mind when they apply a tag. For example, one user may use the term ‘funny’ to describe movies that are silly, involving, say, physical or slapstick humor, while another may use the same term to refer to dry, political satire.<sup>8</sup> While correlated, these two *tag senses* will order movies quite differently. Such *sense subjectivity* may hinder our ability to produce an accurate CAV and understand a user’s true intent. We now turn to this issue and show how to learn *personalized semantics* for tags.

<sup>8</sup>Some users may offer more “refined” tags to distinguish their intended sense, but many will not. For this reason, we still require the ability to disambiguate a user’s intended meaning in the general case.

#### 4.1 Subjectivity of Degree

As discussed above, *degree subjectivity* is likely to emerge quite naturally. The use of *intra-user pairwise comparisons* with a ranking loss in CAV training (see Eq. (2)) ensures that the induced CAVs are robust to this form of subjectivity. However, since two users may use a tag  $g$  differently if they have different thresholds for applying  $g$ , interpreting  $u$ 's usage (e.g. statements such as “that’s not funny” or “I’m in the mood for something funny”) requires a *personalized semantics* that is sensitive to her threshold. Let  $g$  be a tag that is degree subjective,  $\phi_g$  be  $g$ 's CAV and  $\phi_g(i)$  be the degree to which item  $i$  satisfies  $\phi_g$ . A *user-specific threshold*  $\tau_g^u \in \mathbb{R}$  determines a user-specific semantics for  $g$ : tag  $g$  applies (typically, noisily) to item  $i$  for user  $u$  only if  $\phi_g(i) \geq \tau_g^u$ .<sup>9</sup> The (estimated) *optimal personal threshold*  $\tau_g^u$  minimizes the number of misclassifications with respect to  $u$ 's usage of  $g$ :

$$\tau_g^u \in \arg \min_{\tau} |\{i \in T_{u,g} : \phi_g(i) \geq \tau\} \cup \{i \in T_{u,\bar{g}} : \phi_g(i) < \tau\}|. \quad (4)$$

The threshold  $\tau_g^u$ , among the continuum of minimizers, that maximizes the margin between the nearest bracketing positive and negative items is a natural choice for  $u$ 's personal semantic threshold.<sup>10</sup>

#### 4.2 Subjectivity of Sense

We now turn to *sense subjectivity*. We can readily detect sense subjectivity and assign a *personalized semantics* for a tag  $g$  to different (groups of) users, using *distinct CAVs* for each tag sense. We assume that  $g$  has *at most*  $s_g$  distinct senses  $g[1], \dots, g[s_g]$ , for some small positive integer  $s_g$ , where each sense denotes a different soft attribute (we discuss their relation below). Moreover, each user adopts exactly *one* such sense of  $g$ .<sup>11</sup> We propose a method to discover whether a tag has multiple senses, and to uncover suitable CAVs for each sense if so.

Intuitively, if the CAV quality metric  $Q(\phi_g; D)$  is high, then CAV  $\phi_g$  well explains usage of  $g$  among all users in dataset  $D$ . If not, then model  $\Phi$  is unlikely to have learned a good representation for  $g$ . This could be due to  $g$  being poorly correlated with user ratings (hence, user preferences), or because  $g$  has multiple senses. In the latter case, if  $g$  has  $s$  senses, there should be some user-partitioning of  $D$  into subsets  $D_1, \dots, D_s$  such that there is a CAV  $\phi_{g,k}$  with high quality  $Q(\phi_{g,k}; D_k)$  for each  $k \leq s$ . We first propose a simple scheme to find a good set of CAVs for a fixed  $s$ , then discuss determination of a suitable number of senses  $s \leq s_g$ .

Assume a fixed number of target senses  $s$  and a given data set  $D$ . We propose a simple scheme for generating  $s$  CAVs corresponding to maximally distinct senses for  $g$ . Let  $\Sigma = \{\sigma_1, \dots, \sigma_s\}$  be a partitioning of users into  $s$  clusters, where  $\sigma_k$  is the set of users that adopt a common sense for  $g$ . Let  $D_k$  be the restriction of  $D$  to tag data generated by users  $u \in \sigma_k$ . For a fixed  $\Sigma$ , we can readily generate a CAV  $\phi_{g,k}$  for each data set  $D_k$  capturing the corresponding sense, and measure its quality  $Q(\phi_{g,k}; D_k)$ . Of course, this quality depends on whether the partitioning  $\Sigma$  is sensible (i.e., whether most users in each cluster use  $g$  similarly). If the quality of these CAVs is low, one can repartition users by “assigning” each user  $u$  to the cluster in  $\Sigma$  whose CAV best explains her tag usage:

$$k_u^* = \arg \max_k |\{(i, j) : i \in T_{u,g}, j \in T_{u,\bar{g}}, \phi_{g,k}(i) \geq \phi_{g,k}(j)\}|. \quad (5)$$

<sup>9</sup>Equivalently, this can be viewed as a *personal* linear separator for  $u$  in  $X$ , but constrained to be orthogonal to the direction  $\phi_g$  induced from the *population* labels.

<sup>10</sup>Since tag usage by an individual user is typically sparse, these thresholds are likely to be noisy (this is one reason the CAVs themselves are not computed per user, though see more below). But we can refine  $u$ 's semantics with well-chosen queries, e.g., “Do you consider movie  $m$  to be violent?” This can be used to implement a loose binary search for an approximate threshold (possibly made robust to account for noisy responses), but we defer this to future research). If usage is correlated within user sub-populations, generalization of thresholds across users is also viable, as we discuss in the case of sense subjectivity below.

<sup>11</sup>We defer to future work the possibility of *mixtures* of senses.

|            | Linear       |              | NonLin, Lin-Emb |              | Nonlin, NL-Emb |              |
|------------|--------------|--------------|-----------------|--------------|----------------|--------------|
|            | Accur.       | Sprmn        | Accur.          | Sprmn        | Accur.         | Sprmn        |
| Log. Regr. | 0.872        | 0.566        | 0.860           | 0.523        | 0.886          | 0.548        |
| RankNet    | 0.960        | <b>0.671</b> | <b>0.947</b>    | <b>0.660</b> | <b>0.961</b>   | 0.680        |
| LambdaNet  | <b>0.962</b> | 0.669        | 0.938           | 0.653        | 0.958          | <b>0.684</b> |
| PITF       | 0.700        | 0.064        | 0.708           | 0.068        | N/A            | N/A          |

Table 3. CAV Evaluation, Synthetic Data (Degree subjectivity)

|               | Subjective Tags |              |                 |              |                |              | Objective Tags |              |                 |              |                |              |
|---------------|-----------------|--------------|-----------------|--------------|----------------|--------------|----------------|--------------|-----------------|--------------|----------------|--------------|
|               | Linear          |              | NonLin, Lin-Emb |              | Nonlin, NL-Emb |              | Linear         |              | NonLin, Lin-Emb |              | Nonlin, NL-Emb |              |
|               | Accur.          | Sprmn        | Accur.          | Sprmn        | Accur.         | Sprmn        | Accur.         | Sprmn        | Accur.          | Sprmn        | Accur.         | Sprmn        |
| Log. Regr.    | 0.643           | 0.039        | 0.634           | 0.026        | 0.616          | 0.036        | 0.936          | 0.634        | 0.926           | 0.642        | 0.922          | 0.635        |
| EM Log. Regr. | 0.833           | <b>0.478</b> | 0.796           | 0.419        | 0.828          | 0.476        | 0.937          | 0.631        | 0.926           | <b>0.637</b> | 0.922          | 0.635        |
| RankNet       | 0.615           | 0.042        | 0.606           | 0.035        | 0.603          | 0.022        | <b>0.992</b>   | <b>0.636</b> | <b>0.987</b>    | 0.636        | <b>0.982</b>   | <b>0.636</b> |
| EM RankNet    | <b>0.922</b>    | 0.466        | <b>0.915</b>    | <b>0.468</b> | <b>0.908</b>   | <b>0.468</b> | <b>0.992</b>   | 0.633        | <b>0.987</b>    | 0.633        | <b>0.982</b>   | <b>0.636</b> |
| LambdaRank    | 0.615           | 0.028        | 0.606           | 0.036        | 0.604          | 0.009        | <b>0.992</b>   | 0.634        | <b>0.987</b>    | 0.632        | <b>0.982</b>   | 0.634        |
| EM LambdaRank | 0.920           | 0.458        | <b>0.915</b>    | 0.465        | <b>0.908</b>   | 0.465        | <b>0.992</b>   | 0.631        | <b>0.987</b>    | 0.630        | <b>0.982</b>   | 0.631        |
| PITF          | 0.672           | 0.063        | 0.711           | 0.070        | N/A            | N/A          | 0.640          | 0.050        | 0.675           | 0.074        | N/A            | N/A          |

Table 4. CAV Evaluation, Synthetic Data (Sense Subjectivity): Subjective Tags (left half), Objective Tags (right half).

This leads to a EM-like alternating optimization procedure [4] for finding a good clustering that repeatedly: (a) learns a CAV for each current (user) cluster; then (b) reconstructs the clusters by assigning each user to the CAV that best explains her tag usage. The iterative process proceeds until the partitioning  $\Sigma$  no longer changes or quality improvements become sufficiently small. Since we use a hard clustering scheme and only change  $\Sigma$  when the new clusters yield strictly better quality, it is easy to see that the EM procedure terminates in a finite number of steps. If we assign each  $u$  by minimizing the logistic or ranking loss incurred by  $u$ , using convergence properties of the standard  $k$ -means algorithm (e.g., [6]), one can show that the procedure converges to a local minimum and generates  $s$  distinct CAV senses.<sup>12</sup>

Assuming that the complexity of the regression/ranking subproblem is linear in number of examples  $N$ , with  $L$  iterations of EM the complexity of this approach is  $O(LNs)$ . Empirically the EM method converges very quickly ( $L \leq 5$ ) in our experiments, so the computational cost of the EM algorithm is minimal (and recall that these are offline computations). We can search for the appropriate number of senses—effectively a form of *model selection* [4]—by starting with an initial (single) CAV, and applying the procedure above to gradually increasing numbers of clusters  $s = 2, 3, \dots, s_g$ , terminating once the improvement in average quality,  $\sum_k (|D_k|/|D|)Q(\phi_{g,k}; D_k)$  is negligible.<sup>13</sup>

Notice that our approach to disentangling tag senses is in the style of top-down “disaggregative” clustering. We do this since bottom-up agglomerative clustering is likely to be very noisy—the tag set of any individual user is extremely sparse, so attempts to produce a CAV for very small groups of users will generally be unreliable. Once we find the appropriate number of senses and corresponding CAVs, in practice, we can continue to update them as new users, items and tagging data arises. For example, given a new user  $u$  who has applied tags to items, we can associate a “personal” semantics for  $u$  with a given tag  $g$  by assigning  $u$  to the  $g$ -cluster whose CAV best fits  $u$ ’s usage data by computing  $k_u^*$ .

### 4.3 Empirical Assessment

**Synthetic Results.** We again test our approach on synthetic data to exploit access to ground truth CAV semantics. We generate several data sets incorporating both degree and sense subjectivity, as well as linear and nonlinear soft attributes. The model is similar to that used in Sec 3.3, differing only in the addition of subjectivity to user tagging

<sup>12</sup>A number of variants of this general approach can be explored (but are beyond the scope of this work): various criteria for initialization (e.g., using ICA for initial clustering); hard vs. soft clustering; various termination criteria; etc.

<sup>13</sup>A number of other more complex approaches can be considered.

|                | sci-fi       | atmo-spheric | surreal      | twist ending | action       | funny        | classic      | dark comedy  | quirky       | psychology   | dystopia     | stylized     | thought-provoking |
|----------------|--------------|--------------|--------------|--------------|--------------|--------------|--------------|--------------|--------------|--------------|--------------|--------------|-------------------|
| Log. Regr.     | 0.831        | 0.721        | 0.739        | 0.705        | 0.823        | 0.689        | 0.818        | 0.714        | 0.725        | 0.715        | 0.737        | 0.753        | 0.764             |
| RankNet        | 0.905        | 0.793        | 0.811        | 0.817        | <b>0.899</b> | <b>0.775</b> | 0.877        | 0.822        | <b>0.840</b> | 0.812        | 0.850        | 0.866        | 0.834             |
| LambdaNet      | <b>0.906</b> | 0.784        | 0.788        | 0.869        | 0.876        | 0.605        | 0.838        | 0.830        | 0.821        | 0.788        | 0.867        | 0.809        | 0.839             |
| NL. Log. Regr. | 0.831        | 0.725        | 0.742        | 0.711        | 0.812        | 0.681        | 0.821        | 0.705        | 0.751        | 0.704        | 0.807        | 0.811        | 0.764             |
| NL. RankNet    | 0.893        | 0.838        | <b>0.827</b> | 0.854        | 0.890        | 0.754        | <b>0.888</b> | 0.868        | 0.833        | <b>0.837</b> | 0.882        | 0.856        | 0.844             |
| NL. LambdaNet  | 0.891        | <b>0.843</b> | 0.807        | <b>0.875</b> | 0.880        | 0.744        | 0.865        | <b>0.891</b> | 0.825        | 0.796        | <b>0.921</b> | <b>0.898</b> | <b>0.856</b>      |

Table 5. CAV Accuracy Evaluation, 13 Possible Subjective Concepts in MovieLens

behavior. To test degree subjectivity, we use five tags as above, but with each user’s personal tagging threshold sampled from a mixture distribution with two components. To test sense subjectivity, we introduce a subjective tag “tag-S” with three senses, each reflecting one of three (of the five) taggable dimensions. Each user adopts exactly one of these three senses—when applying tag-S to an item, they assess it based on their assigned dimension. The remaining two tags are objective. We evaluate the three CAV training methods used in Sec. 3.3, applying each to a linear (WALS) model and a nonlinear two-tower model as needed. For sense subjectivity, we test our EM-like algorithm with each training method. Further details on data generation and the experimental set up can be found in the appendix.

Table 3 summarizes performance of the CAVs under degree subjectivity, using the same methods and models as in Sec. 3.3 (results averaged over the five degree-subjective tags). In contrast to the non-subjective case in Sec 3.3, where users have the same threshold for each tag, here the per-user ranking-based methods (RankNet and LambdaRank) significantly outperform logistic regression, demonstrating the need to be sensitive to a user’s degree subjectivity.

Table 4 summarizes results for sense subjectivity, showing CAV accuracy for our baseline methods both with and without our EM-based approach for distinguishing senses. The left side of the table shows results for the sense-subjective tag-S, demonstrating that EM can dramatically improve CAV accuracy by disentangling the three distinct senses. This shows that treating a subjective concept as objective can be problematic. Note also that the ranking methods perform better than logistic regression. The right side of the table shows CAV accuracy on the two *objective* tags: the EM and non-EM methods perform almost identically, indicating that we are unlikely to identify spurious senses. Somewhat intriguingly, we see that the use of nonlinear CAVs does not offer much improvement over linear CAVs with ranking methods, though nonlinear CAVs perform better when trained using logistic regression. The performance of the PITF baseline is worse than that of the CAV approaches in both the degree and sense subjectivity experiments.

**MovieLens Results.** We also evaluate our subjective CAV methods on MovieLens20M. To assess degree subjectivity, we select 13 tags deemed to be degree subjective and compare the accuracy of different CAV methods for each (Table 5). Since MovieLens data has no ground truth with respect to possible subjectivity, Spearman rank correlation cannot be measured. Generally the ranking methods outperform logistic regression, suggesting that real users exhibit some variation in their thresholds (degree subjectivity) with some tags (e.g., *sci-fi*) having much higher agreement and CAV-predictability than others (e.g., *funny*). We also observe differences in the degree of improvement offered by nonlinear CAVs vs. linear CAVs across the tags: those with larger improvements (e.g., *dark comedy*, *dystopia*) suggest that user utility may often be nonlinear in the degree of that attribute (so extreme degrees may not be preferred); while those where nonlinear CAVs perform no better, or even worse (e.g., *sci-fi*, *action*, *funny*), may be most preferred at their maximum or minimum degree.

For sense subjectivity, we construct two types of *artificial tags* from MovieLens data. First are *objective tags*, capturing four *genres* (comedy, horror, fantasy and romance). For a random subset of user-item pairs in the tag dataset, we add the corresponding user-item-genre triple to the set if the item’s meta-data lists that genre. This ensures that the new “genre tag” data replicates natural tagging patterns. We also add a synthetic tag *odd year*—was a movie was released in an even

|                          | Odd Year     | Comedy       | Horror       | Fantasy      | Romance      | Monsters     | Funny        | Intrigue     | Relationship |
|--------------------------|--------------|--------------|--------------|--------------|--------------|--------------|--------------|--------------|--------------|
| Log. Regr., Lin. Emb.    | 0.519        | 0.521        | 0.770        | 0.685        | 0.693        | 0.671        | 0.658        | 0.669        | 0.662        |
| EM Log. Regr., Lin. Emb. | 0.532        | <b>0.685</b> | 0.759        | 0.744        | 0.704        | 0.831        | 0.769        | 0.712        | 0.730        |
| RankNet, Lin. Emb.       | 0.505        | 0.620        | 0.790        | 0.778        | 0.730        | 0.718        | 0.705        | 0.660        | 0.634        |
| EM RankNet, Lin. Emb.    | <b>0.593</b> | 0.676        | 0.833        | <b>0.824</b> | 0.749        | <b>0.892</b> | <b>0.874</b> | 0.834        | 0.840        |
| LambdaRank, Lin. Emb.    | 0.533        | 0.609        | 0.809        | 0.779        | 0.716        | 0.719        | 0.718        | 0.661        | 0.623        |
| EM LambdaRank, Lin. Emb. | 0.582        | 0.670        | <b>0.838</b> | 0.819        | <b>0.762</b> | 0.883        | 0.870        | <b>0.836</b> | <b>0.847</b> |

Table 6. CAV Accuracy Evaluation, Artificial MovieLens Tags (5 objective, 4 sense-conflated)

or odd year—to 50% of user-item pairs. This (presumably) *preference-irrelevant* attribute acts as a baseline for which no good CAV should be discoverable. These tags are “objective”—their presence does not depend on a user’s interpretation of tags (though they may depend on a user’s inclination to tag certain types of movies). The second artificial tag type are *sense-conflated* tags, constructed by coalescing several related “ground” tags into a single “meta-tag” then replacing each ground tag with that meta-tag. Each ground tag in the group can be viewed as a subjective sense of the meta-tag. We test our ability to “disentangle” the different senses of the meta-tag relative to the ground truth. We introduce four meta-tags: *monsters* (grouping *zombies*, *ghosts* and *vampires*), *funny* (*parody*, *satire*, *dark humor*), *intrigue* (*corruption*, *conspiracy*, *politics*) and *relationship* (*family*, *friendship*, *love story*). For each user and meta-tag, we choose *exactly one* ground tag from the group as that user’s *designated sense* and add the meta-tag to the data set for each user-item-tag triple that uses the ground tag (e.g., some users have all of their *friendship* tags replaced with *relationship*, while others have all of their *love story* tags replaced this way).

Table 6 summarizes the accuracy of our trained CAVs for these two types of artificial tags. For the four “objective” *genre* tags, the ranking-based methods outperform logistic regression. However, adding EM to our ranking methods provides only modest incremental benefit (especially relative to lift it offers for the “subjective” sense-conflated tags below). This implies that *genres* exhibit at most modest, if any, subjectivity of sense, as expected.<sup>14</sup> The ‘horror’ and ‘fantasy’ tags are easiest to learn, suggesting they are more “objective” and “linear.” None of our methods find a good CAV for the artificial tag *odd year*—their predictive accuracy is barely above random—corroborating our hypothesis that CAVs are useful for identifying *preference-related* attributes/tags. On the other hand, results on our four *sense-conflated* tags clearly demonstrate the ability of our EM-style approach to disentangle the distinct subjective or personal senses of each of the meta-tags—this is true for each baseline algorithm—thus greatly improving tag prediction accuracy.

## 5 USING CAVS FOR EXAMPLE CRITIQUING

While preference elicitation in RSs often uses attributes [5, 38, 48], an important question is the extent to which such methods can be adapted to handle soft attributes (see, e.g., [39]). We do not consider this question in its full depth, but examine how the CAV semantics for tags can be used for *example critiquing* [14]. In lieu of live experiments, we adopt a stylized but plausible *user response model*, in which a user’s critiques are driven by her underlying utility function and her personal tag/attribute semantics.<sup>15</sup> While other response models are possible (e.g., models that are fit to real user interaction data), this model suffices to demonstrate the value of CAVs for critiquing. We run both synthetic experiments in which we have access to the ground truth utility and semantics for each user, and a MovieLens experiment, for which we propose a novel method for generating utilities and responses.

We assume a predefined list of critiquable tags and adopt a simple interactive RS that supports user critiques. We provide a brief description of our set up herem but refer to Appendix A.5 for further details, parameter values used, etc.

<sup>14</sup>EM will often provide some improvement in accuracy, even if distinct senses do not exist, by allowing some overfitting. In this case, EM may also overcome the limitations of the linear CAVs if user utilities for these objective genres are non-linear.

<sup>15</sup>Synthetic user models are commonly used to evaluate RSs [21, 55].

The interactive RS, at each iteration of interaction with user  $u$ , the RS presents a slate  $S$  of  $k$  items. User  $u$  can *accept* one of the recommended items (at which point the session terminates). Otherwise,  $u$  can *critique*  $S$  using a tag  $g$  and a desired direction (‘more,’ ‘less’); e.g., for movies,  $u$  might offer a critique like “more funny” or “less violent.” The RS then updates its *user representation* given this response and generates the next recommended slate. The process repeats until the user terminates by accepting a recommendation or the RS reaches the maximum critiquing steps  $T$ .

User interactions assume a *user response model* in which user  $u$  has (i) a (personal) ground truth utility function over items (i.e.,  $u$  can assess an item’s utility if recommended) and (ii) a (personal) ground-truth semantics for attributes (i.e.,  $u$  can assess any item’s attribute values). User  $u$  also has a rough estimate of the maximum and minimum levels any tag/attribute can attain in the item corpus—this information is used to guide her critiquing behavior without assuming she has unrealistic knowledge of the item corpus (see Appendix A.5). When presented with slate  $S$ ,  $u$  accepts an item  $i$  if its utility is sufficiently large. Otherwise,  $u$  critiques using the *most salient tag*  $g$  with respect to utility improvement of  $S$ , i.e.,  $g = \operatorname{argmax}_g \delta_u^T w_g$ , where  $\delta_u = (\phi_I(i_u^*) - \frac{1}{|S|} \sum_{i \in S} \phi_I(i)) \odot \phi_U(u)$  is the utility difference vector between  $u$ ’s estimated ideal item  $i_u^*$  and her average utility vector over the  $k$  items in  $S$ , and  $w_g$  is  $u$ ’s interpretation of  $g$ .

We assume item embeddings  $\phi_I(i)$  are fixed, and use a *simple heuristic RS strategy* for incorporating critiques.<sup>16</sup> Given a user embedding  $\phi_U(u)$ , the RS scores all items  $i$  in the corpus with respect to utility  $r_{i,u} = \phi_I(i)^T \phi_U(u)$ , and presents the slate  $S$  of the top  $k$  scoring items. If the user critiques  $S$  with a tag  $g$  (and a specific direction), the RS updates the user embedding as follows:  $\phi_U(u) \leftarrow \phi_U(u) + \alpha_t(g) \cdot \phi_g$ , where  $\phi_g$  is  $g$ ’s CAV,  $\alpha_t(g)$  is a tag-specific step size at iteration  $t$  and the sign of  $\alpha_t(g)$  reflects the direction (‘more,’ ‘less’) of the critique. We treat the magnitude of  $\alpha_t(g)$  as a hyper-parameter selected to optimize the utilities of the resulting recommendations.<sup>17</sup> The RS then recommends the top  $k$  items given the updated user embedding, and the process repeats. If the tag  $g$  is sense-subjective, the critique is interpreted relative to the RS’s estimate of  $u$ ’s sense (or cluster) based on past usage.

We first analyze critiquing with CAVs using the same synthetic models described above, where a user’s critiques are generated with the user’s ground truth utility and tag semantics. By contrast, the RS uses its *estimated user embedding* and the *tag’s CAV* to interpret a user’s critique (and not the ground truth). In our experiment, the RS slate size is  $k = 10$ , and the maximum number of critiquing steps is  $T = 25$ . We evaluate how recommendation quality improves as the number of critiques increases, by measuring both *user max utility* of the top- $k$  slate, i.e.,  $UMU(S) = \max_{i \in S} U(i)$ , and *user average utility*  $UAU(S) = \operatorname{avg}_{i \in S} U(i)$ , where  $U(i)$  is  $u$ ’s *true* utility for  $i$ .  $UMU(S)$  reflects the utility of a user who is able to select the best item from  $S$ , while  $UAU(S)$  corresponds to random selection from the slate.

Fig. 2 presents interactive critiquing results in three experiments with different synthetic data sets: the first has no subjectivity and linear utility; the second no subjectivity and nonlinear utility; the third degree subjectivity and nonlinear utility. In all experiments, user utility (both  $UMU$  and  $UAU$ ) improves with additional critiquing steps, eventually converging to a steady-state value. These results corroborate our hypothesis that, since CAVs represent soft attributes well in embedding space, they can be used to effectively update an RS’s beliefs about user preferences as the user critiques recommended items, which in turn improves recommendation quality. Furthermore, recall from Sections 3 and 4 that CAVs trained with logistic regression generally have lower accuracy than those trained with RankNet or LambdaRank. While our CAV algorithms are not optimized to support critiquing, more accurate CAVs learned using ranking methods give rise to better interpretations of user critiques (this observation is reflected by both

<sup>16</sup>We emphasize that the RS strategy used and method for incorporating critiques are fairly generic and are not intended to reflect the state-of-the-art, since our goal is to measure the ability to exploit learned CAVs. More elaborate strategies for updating user embedding are possible, including the use of Bayesian updates relative to a prior over the user embedding [47]. Here instead we adopt a simple heuristic, based on [28], to focus attention on the CAV semantics itself.

<sup>17</sup>Ultimately, this heuristic adjustment should be tuned to the specifics of a user response model generated from live experiments.



the faster improvement and greater steady-state values for both  $UAU$  and  $UMU$ ). This is most likely due to the fact that more accurate ranking-based CAVs better capture a user’s intended semantics during critiquing. Similarly, the improved accuracy of nonlinear CAVs when utility is nonlinear is manifest in the improved critiquing performance (in both the objective and degree-subjectivity tests, and for both  $UAU$  and  $UMU$ ). Again, this performance improvement is likely due to the better CAV representation uncovered from the intermediate layers of the DNN.

Figure 3 shows interactive critiquing results using synthetic data with *sense subjectivity* with (i) linear utility, (ii) nonlinear utility with linear CAVs, and (iii) nonlinear CAVs. Again, both  $UMU$  and  $UAU$  improve with more critiquing steps, eventually converging to a steady-state value. Recall that only three of 25 utility-relevant dimensions correspond to “conflated” senses of a single tag. Still in the case of nonlinear utility, EM-LambdaRank, even with linear CAVs, outperforms LambdaRank without EM. This suggests that disentangling sense subjectivity is more critical than having an “accurate” but incorrectly-assumed objective single CAV. These results also show that EM Logistic Regression can be improved with the use of nonlinear CAVs as suggested in Table 4.

To evaluate critiquing with MovieLens data, we propose a novel method for hypothesizing “ground-truth” user utility. We first train a CF model with all users and items, then train (non-subjective) CAVs for 164 tags (see Sec. 3.3). We then construct a small set of *test users*, each of whom has rated at least 50 movies. We use the learned embedding  $\phi_U(u)$  for each test user  $u$  as if it were their *ground truth utility* (since  $u$  has rated a large number of movies, we expect this to be reasonably stable and accurate). We then run the interactive critiquing RS by *forgetting* each test user (their ratings and tags) and treating them as a “cold start” user, who is given an generic prior embedding.<sup>18</sup> This  $u$  then generates critiques of the RS slates using  $\phi_U(u)$  as their true utility. Since we have no ground truth tag semantics, each  $u$  treats the RS’s *learned CAV* as her semantics (admittedly giving the RS some advantage when interpreting critiques). Otherwise, the user response model is exactly as in the synthetic case. We evaluate as in the synthetic case, but user utility improvements are *estimates* using the learned embedding  $\phi_U(u)$ . Because of this we also assess some additional metrics (see below).

Fig. 4 shows critiquing results with MovieLens data. In addition to  $UAU(S)$  we also report normalized discounted cumulative gain (NDCG) [46], mean reciprocal rank (MRR) [32] and average binarized rating<sup>19</sup> [40] of slates  $S$  generated during critiquing. We compare critiquing results generated by four sets of CAVs, trained with the following methods: RankNet, nonlinear RankNet, nonlinear LambdaRank and nonlinear logistic regression (as these methods usually generate more accurate CAVs as shown above). While all methods perform similarly with respect to  $UAU$ , nonlinear LambdaRank outperforms the others with respect to the other three metrics. This again provides evidence that: (i) incorporating CAVs in RSs to capture user-critiquing behavior can improve recommendation quality (both utilities and ratings); and (ii) the performance of critiquing-based RSs tends to improve with the quality of the learned CAVs.

## 6 CONCLUSIONS

We have presented a novel methodology for discovering the semantics of soft attribute/tag usage in RSs using concept activation vectors. Its benefits include: (i) using a CF representation to identify attributes of greatest relevance to the recommendation task; (ii) distinguishing objective and subjective tag usage; (iii) identifying personalized, user-specific semantics for subjective attributes; and (iv) relating attribute semantics to preference, thereby allowing interactions using soft attributes/tags in example critiquing and other forms of preference elicitation.

<sup>18</sup>We use the average of all learned user embeddings as a prior.

<sup>19</sup>We set the binarized rating to 1 if the numeric rating exceeds 3, and 0 otherwise. This is a binary precision measure with respect to highly-rated items.

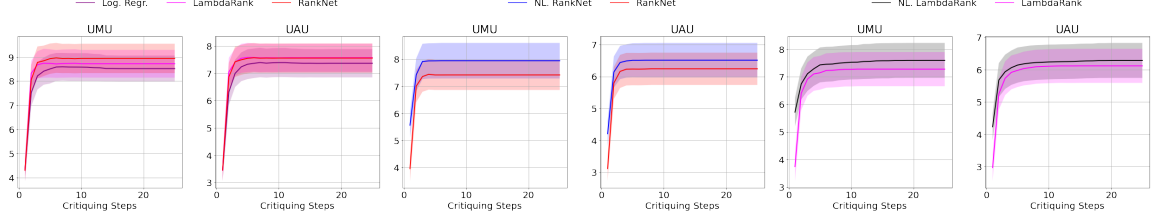


Fig. 2. Results of Interactive Critique with Synthetic Data. Left two: No Subjectivity Linear Utility with All Methods; Middle two: No Subjectivity Nonlinear Utility with RankNet; Right two: Degree Subjectivity Nonlinear Utility with LambdaRank

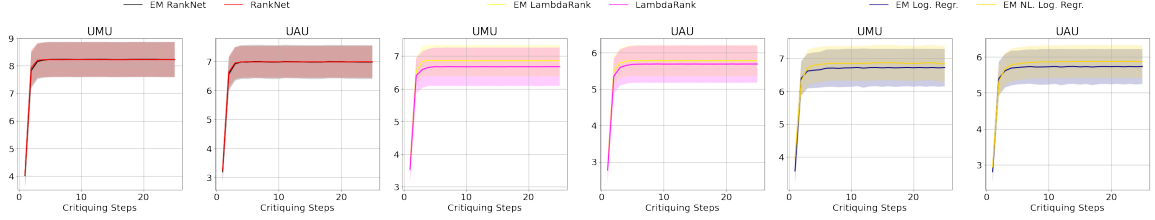


Fig. 3. Results of Interactive Critique with Synthetic Data. Left two: Sense Subjectivity Linear Utility with RankNet; Middle two: Sense Subjectivity Nonlinear Utility with LambdaRank, Lin-Emb; Right two: Sense Subjectivity Nonlinear Utility with Log. Regr., NL-Emb.

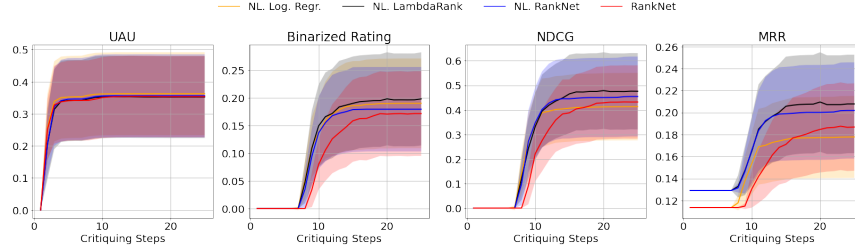


Fig. 4. Results of Interactive Critique with MovieLens Data

A number of future directions can provide additional value to the use of CAVs for elicitation and critiquing in RSs. While our empirical results suggests that CAVs are useful for subjective attributes and critiquing, additional study with real user critiques is critical, as is developing real-world data sets with ground truth utility and personal semantics (e.g., via survey instruments or controlled experiments).

In our current formulation, CAVs are learned offline under the assumption that the dataset contains sufficient tagging data reflecting the personal preferences and semantics of diverse users. When this is not that case, a useful extension is to consider an interactive methods in which the system can actively elicit a user’s personal semantics, for example, using *pairwise tag-comparison queries* to identify a user’s tag interpretation. For example, a few well-chosen queries like “Which of these two books is more thought-provoking?” or “Do you consider this song to be upbeat?” could help identify a user’s personal semantics (both degree thresholds and tag senses).

Our focus in this work has been on soft attribute usage in an item-space that is characterized entirely by latent attributes (e.g., as is common in content recommendation). Other domains, such as product recommendation, are traditionally characterized by hard attributes (e.g., price, color, weight, efficiency, capacity, and other product-specific features). However, the use of CAVs to allow the more flexible use of soft and subjective attributes (e.g., cozy, comfortable,

vibrant, stylish, compact, inexpensive, etc.) to navigate such domains is of great interest. This raises interesting questions of how to integrate soft and hard attributes, including the possibility of exploiting the ground-truth semantics of hard attributes to facilitate the discovery of soft-attribute semantics (e.g., relating (possibly subjective) terms like “compact” to product dimensions or weight, or “inexpensive” to price).

Finally, it would be of great value for critiquing-based RSs to have CAV-learning algorithms that not only maximize concept accuracy, but whose representations are more directly tuned to support preference elicitation. and to more directly compare CAVs to alternative ways of understanding soft and subjective attributes.

## ACKNOWLEDGEMENTS

Thanks to Filip Radlinski, Steffen Rendle, Li Zhang, Dima Kuzmin and Tania Bedrax-Weiss for helpful discussions and feedback, as well as the anonymous reviewers of an earlier version of the manuscript.

## REFERENCES

- [1] Krisztian Balog, Filip Radlinski, and Alexandros Karatzoglou. 2021. On Interpretation and Measurement of Soft Attributes for Recommendation. In *Proceedings of the 44th International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR '21)*. 890–899.
- [2] G. Bang, G. Barash, R. Bea, J. Cali, M. Castillo-Effen, X. Chen, N. Chhaya, R. Cummings, R. Dhoopar, S. Dumanci, H. Espinoza, E. Farchi, F. Fioretto, R. Fuentesetaja, C. Geib, O. E. Gundersen, J. Hernández-Orallo, X. Huang, K. Jaidka, S. Keren, S. Kim, M. Galley, X. Liu, T. Lu, Z. Ma, R. Mallah, J. McDermid, M. Michalowski, R. Mirsky, S. Ó hÉigearthaigh, D. Ramachandran, J. Segovia-Aguas, O. Shehory, A. Shaban-Nejad, V. Schwartz, S. Srivastava, K. Talamadupula, J. Tang, P. Van Hentenryck, D. Zhang, and J. Zhang. 2020. The Association for the Advancement of Artificial Intelligence 2020 Workshop Program. In *AI Magazine*. 100–114.
- [3] Alex Beutel, Paul Covington, Sagar Jain, Can Xu, Jia Li, Vince Gatto, and Ed H. Chi. 2018. Latent Cross: Making Use of Context in Recurrent Recommender Systems. In *Proceedings of the Eleventh ACM International Conference on Web Search and Data Mining (WSDM-18)*. Marina Del Rey, CA, 46–54.
- [4] Christopher M Bishop. 2006. *Pattern Recognition and Machine Learning*. Springer, New York.
- [5] Edwin V. Bonilla, Shengbo Guo, and Scott Sanner. 2010. Gaussian Process Preference Elicitation. In *Advances in Neural Information Processing Systems 23 (NIPS-10)*. Vancouver, 262–270.
- [6] Léon Bottou and Yoshua Bengio. 1994. Convergence Properties of the K-Means Algorithms. In *Advances in Neural Information Processing Systems 7 (NIPS-94)*. 585–592.
- [7] Craig Boutilier, Kevin Regan, and Paolo Viappiani. 2009. Online Feature Elicitation in Interactive Optimization. In *Proceedings of the Twenty-sixth International Conference on Machine Learning (ICML-09)*. Montreal, 73–80.
- [8] Craig Boutilier, Kevin Regan, and Paolo Viappiani. 2010. Simultaneous Elicitation of Preference Features and Utility. In *Proceedings of the Twenty-fourth AAAI Conference on Artificial Intelligence (AAAI-10)*. Atlanta, 1160–1167.
- [9] Craig Boutilier, Richard S. Zemel, and Benjamin Marlin. 2003. Active Collaborative Filtering. In *Proceedings of the Nineteenth Conference on Uncertainty in Artificial Intelligence (UAI-03)*. Acapulco, 98–106.
- [10] Chris Burges, Tal Shaked, Erin Renshaw, Ari Lazier, Matt Deeds, Nicole Hamilton, and Greg Hullender. 2005. Learning to Rank using Gradient Descent. In *Proceedings of the Twenty-second International Conference on Machine Learning (ICML-05)*. 89–96.
- [11] Christopher JC Burges. 2010. From RankNet to LambdaRank to LambdaMART: An Overview. *Learning* 11, 23–581 (2010), 81.
- [12] Robin Burke. 2002. Interactive Critiquing for Catalog Navigation in E-Commerce. *Artificial Intelligence Review* 18, 3–4 (2002), 245–267.
- [13] Yunbo Cao, Jun Xu, Tie-Yan Liu, Hang Li, Yalou Huang, and Hsiao-Wuen Hon. 2006. Adapting ranking SVM to document retrieval. In *Proceedings of the 29th annual international ACM SIGIR conference on Research and development in information retrieval*. 186–193.
- [14] Li Chen and Pearl Pu. 2012. Critiquing-based Recommenders: Survey and Emerging Trends. *User Modeling and User-Adapted Interaction* 22, 1 (2012), 125–150.
- [15] Deborah Cohen, Michal Aharon, Yair Koren, Oren Somekh, and Raz Nissim. 2017. Expediting Exploration by Attribute-to-feature Mapping for Cold-start Recommendations. In *Proceedings of the 11th ACM Conference on Recommender Systems (RecSys17)*. Como, Italy, 184–192.
- [16] Simon French. 1986. *Decision Theory*. Halsted Press, New York.
- [17] Zeno Gantner, Lucas Drumond, Christoph Freudenthaler, Steffen Rendle, and Lars Schmidt-Thieme. 2010. Learning Attribute-to-Feature Mappings for Cold-Start Recommendations. In *2010 IEEE International Conference on Data Mining (ICDM-10)*. 176–185.
- [18] F. Maxwell Harper and Joseph A. Konstan. 2016. The MovieLens Datasets: History and Context. *ACM Transactions on Interactive Intelligent Systems* 5, 4 (2016), 19:1–19:19.
- [19] Xiangnan He, Lizi Liao, Hanwang Zhang, Liqiang Nie, Xia Hu, and Tat-Seng Chua. 2017. Neural Collaborative Filtering. In *Proceedings of the 26th International Conference on World Wide Web (WWW-17)*. Perth, Australia, 173–182.

- [20] Yifan Hu, Yehuda Koren, and Chris Volinsky. 2008. Collaborative Filtering for Implicit Feedback Datasets. In *Proceedings of the 8th IEEE International Conference on Data Mining (ICDM 2008), December 15-19, 2008, Pisa, Italy*. 263–272.
- [21] Guangda Huzhang, Zhen-Jia Pang, Yongqing Gao, Yawen Liu, Weijie Shen, Wen-Ji Zhou, Qianying Lin, Qing Da, An-Xiang Zeng, Han Yu, Yang Yu, and Zhi-Hua Zhou. 2021. AliExpress Learning-To-Rank: Maximizing online model performance without going online. *IEEE Transactions on Knowledge and Data Engineering* (2021). <https://doi.org/10.1109/TKDE.2021.3098898>
- [22] Been Kim, Martin Wattenberg, Justin Gilmer, Carrie Cai, James Wexler, Fernanda Viegas, and Rory Sayres. 2018. Interpretability Beyond Feature Attribution: Quantitative Testing with Concept Activation Vectors (TCAV). In *Proceedings of the Thirty-fifth International Conference on Machine Learning (ICML-18)*. Stockholm, 2668–2677.
- [23] Diederik Kingma and Jimmy Ba. 2015. Adam: A Method for Stochastic Optimization. In *Proceedings of the Third International Conference on Learning Representations (ICLR-15)*. San Diego, CA.
- [24] Joseph A. Konstan, Bradley N. Miller, David Maltz, Jonathan L. Herlocker, Lee R. Gordon, and John Riedl. 1997. GroupLens: Applying Collaborative Filtering to Usenet News. *Commun. ACM* 40, 3 (1997), 77–87.
- [25] Jonathan Koren, Yi Zhang, and Xue Liu. 2008. Personalized Interactive Faceted Search. In *Proceedings of the 17th International Conference on World Wide Web (WWW-08)*. Beijing, 477–486.
- [26] Ralf Krestel, Peter Fankhauser, and Wolfgang Nejdl. 2009. Latent Dirichlet Allocation for Tag Recommendation. In *Proceedings of the 3rd ACM Conference on Recommender Systems (RecSys09)*. 61–68.
- [27] Artus Krohn-Grimberghe, Lucas Drumond, Christoph Freudenthaler, and Lars Schmidt-Thieme. 2012. Multi-relational matrix factorization using bayesian personalized ranking for social network data. In *Proceedings of the Fifth International Conference on Web Search and Web Data Mining, WSDM 2012*. 173–182.
- [28] Kai Luo, Scott Sanner, Ga Wu, Hanze Li, and Hojin Yang. 2020. Latent Linear Critiquing for Conversational Recommender Systems. In *Proceedings of The Web Conference 2020*. 2535–2541.
- [29] Leandro Balby Marinho, Christine Preisach, and Lars Schmidt-Thieme. 2009. Relational Classification for Personalized Tag Recommendation. In *Proceedings of ECML PKDD (The European Conference on Machine Learning and Principles and Practice of Knowledge Discovery in Databases) Discovery Challenge 2009*, Vol. 497. 7–15.
- [30] Benjamin M. Marlin and Richard S. Zemel. 2007. Collaborative Filtering and the Missing at Random Assumption. In *Proceedings of the Twenty-third Conference on Uncertainty in Artificial Intelligence (UAI-07)*. Vancouver, 50–54.
- [31] Julian McAuley, Jure Leskovec, and Dan Jurafsky. 2012. Learning Attitudes and Attributes from Multi-aspect Reviews. In *12th International Conference on Data Mining (ICDM-12)*. 1020–1025.
- [32] Brian McFee and Gert RG Lanckriet. 2010. Metric learning to rank. In *ICML*.
- [33] Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg Corrado, and Jeffrey Dean. 2013. Distributed Representations of Words and Phrases and their Compositionality. In *Proceedings of the 26th International Conference on Neural Information Processing Systems-Volume 2*. 3111–3119.
- [34] Martin Mladenov, Chih wei Hsu, Vihan Jain, Eugene Ie, Christopher Colby, Nicolas Mayoraz, Hubert Pham, Dustin Tran, Ivan Vendrov, and Craig Boutilier. 2020. Demonstrating Principled Uncertainty Modeling for Recommender Ecosystems with RecSim NG. In *RecSys 2020: Fourteenth ACM Conference on Recommender Systems, Virtual Event, Brazil, September 22-26, 2020*. 591–593.
- [35] Hervé Moulin. 1980. On Strategy-proofness and Single Peakedness. *Public Choice* 35, 4 (1980), 437–455.
- [36] Preksha Nema, Alexandros Karatzoglou, and Filip Radlinski. 2021. Disentangling Preference Representations for Recommendation Critiquing with  $\beta$ -VAE. In *30th ACM International Conference on Information and Knowledge Management (CIKM 2021)*. New York.
- [37] Daniel N. Osherson and Edward E. Smith. 1981. On the Adequacy of Prototype Theory as a Theory of Concepts. In *Cognition*.
- [38] Pearl Pu and Li Chen. 2008. User-involved Preference Elicitation for Product Search and Recommender Systems. *AI Magazine* 29, 4 (2008), 93–103.
- [39] Filip Radlinski, Krisztian Balog, Bill Byrne, and Karthik Krishnamoorthi. 2019. Coached Conversational Preference Elicitation: A Case Study in Understanding Movie Preferences. In *Proceedings of the Annual SIGDial Meeting on Discourse and Dialogue*.
- [40] Al Mamunur Rashid, Istvan Albert, Dan Cosley, Shyong K Lam, Sean M McNee, Joseph A Konstan, and John Riedl. 2002. Getting to know you: learning new user preferences in recommender systems. In *Proceedings of the 7th international conference on Intelligent user interfaces*. 127–134.
- [41] Michel Regenwetter, Bernard Grofman, A. A. J. Marley, and Ilia Tsetlin. 2006. *Behavioral Social Choice: Probabilistic Models, Statistical Inference, and Applications*. Cambridge University Press, Cambridge.
- [42] Steffen Rendle, Christoph Freudenthaler, Zeno Gantner, and Lars Schmidt-Thieme. 2009. BPR: Bayesian Personalized Ranking from Implicit Feedback. In *Proceedings of the Twenty-fifth Conference on Uncertainty in Artificial Intelligence (UAI-09)*. Montreal, 452–461.
- [43] Steffen Rendle and Lars Schmidt-Thieme. 2010. Pairwise Interaction Tensor Factorization for Personalized Tag Recommendation. In *Proceedings of the 3rd ACM International Conference on Web Search and Data Mining (WSDM-10)*. 81–90.
- [44] Ruslan Salakhutdinov and Andriy Mnih. 2007. Probabilistic Matrix Factorization. In *Advances in Neural Information Processing Systems 20 (NIPS-07)*. Vancouver, 1257–1264.
- [45] Mukund Sundararajan, Ankur Taly, and Qiqi Yan. 2017. Axiomatic Attribution for Deep Networks. In *International Conference on Machine Learning*. 3319–3328.
- [46] Hamed Valizadegan, Rong Jin, Ruofei Zhang, and Jianchang Mao. 2009. Learning to Rank by Optimizing NDCG Measure.. In *NIPS*, Vol. 22. 1883–1891.
- [47] Ivan Vendrov, Tyler Lu, Qingqing Huang, and Craig Boutilier. 2020. Gradient-based optimization for Bayesian preference elicitation. In *Proceedings of the AAAI Conference on Artificial Intelligence*, Vol. 34. 10292–10301.

- [48] Paolo Viappiani and Craig Boutilier. 2010. Optimal Bayesian Recommendation Sets and Myopically Optimal Choice Query Sets. In *Advances in Neural Information Processing Systems 23 (NIPS)*. Vancouver, 2352–2360.
- [49] Bifan Wei, Jun Liu, Qinghua Zheng, Wei Zhang, Xiaoyu Fu, and Boqin Feng. 2013. A Survey of Faceted Search. *Journal of Web Engineering* 12, 1&2 (2013), 041–064.
- [50] Charles Welch, Jonathan K. Kummerfeld, Verónica Pérez-Rosas, and Rada Mihalcea. 2020. Exploring the Value of Personalized Word Embeddings. In *Proceedings of the 28th International Conference on Computational Linguistics*.
- [51] 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*. 137–145.
- [52] Ji Yang, Xinyang Yi, Derek Zhiyuan Cheng, Lichan Hong, Yang Li, Simon Xiaoming Wang, Taibai Xu, and Ed H Chi. 2020. Mixed Negative Sampling for Learning Two-tower Neural Networks in Recommendations. In *Proceedings of the Web Conference (WWW-20)*. Taipei, 441–447.
- [53] Xinyang Yi, Ji Yang, Lichan Hong, Derek Zhiyuan Cheng, Lukasz Heldt, Aditee Kumthekar, Zhe Zhao, Li Wei, and Ed Chi. 2019. Sampling-bias-corrected Neural Modeling for Large Corpus Item Recommendations. In *Proceedings of the Thirteenth ACM Conference on Recommender Systems (RecSys19)*. Copenhagen, 269–277.
- [54] Xiaoxue Zhao, Weinan Zhang, and Jun Wang. 2013. Interactive Collaborative Filtering. In *Proceedings of the 22nd ACM International Conference on Information and Knowledge Management (CIKM-13)*. 1411–1420.
- [55] Hao Zou, Peng Cui, Bo Li, Zheyang Shen, Jianxin Ma, Hongxia Yang, and Yue He. 2020. Counterfactual Prediction for Bundle Treatment. In *Advances in Neural Information Processing Systems 33 (NeurIPS-20)*.

## A APPENDIX

We provide additional details on various aspects of this work. We recap key terms and definitions in App. A.1. We fully specify the synthetic data generation process used in App. A.2, describe our precise use of the MovieLens data set in App. A.3, elaborate on our training methods (incl. model architectures, parameter values, etc.) in App. A.4, and provide addition detail on our example critiquing set up in App. A.5.

### A.1 Key Terms and Notations

We first recap several key concepts used in our work.

- *Rating*: measure  $r_{u,i}$  of user  $u$ ’s preference for item  $i$ .
- *User/item embedding*: vector representations of users  $\phi_U(u)$  and items  $\phi_I(i)$  learned using, say, collaborative filtering on ratings data. The estimate of  $u$ ’s rating for  $i$  is  $\hat{r}_{i,u} = \phi_U(u)^\top \phi_I(i)$ . If  $\phi_I$  is represented by a DNN, we denote the activations for item  $i$  at the  $\ell$ -th layer by  $\phi_{I,\ell}(i)$ .
- *Tags*: set of terms  $\mathcal{T}$  propositionally applied by users to describe items. Each tag corresponds an attribute or concept.
- *Concept activation vector (CAV)*: the CAV  $\phi_g$  for a tag  $g$  is a vector in embedding or activation space that represents a direction in which items “possess more of” the concept represented by  $g$ . CAVs can be learned using classification or learning-to-rank methods on tag data.
- *Subjectivity*: we distinguish three types of tags: (1) *objective* tags, where users agree on whether (or the degree to which) an item satisfies the attribute underlying the tag; (2) *degree subjective* tags, where users agree on the degree, but may disagree on whether the (boolean) attribute/tag applies; and (3) *sense subjective* tags, where (groups of) users may disagree on which items possess the attribute.

### A.2 Synthetic Data Generation with RecSim

We use a stylized, but structurally realistic generative model to produce synthetic ratings and tag data for some of our experiments. Its purpose is twofold. First, it offers various parameters or “knobs” that can be used to generate data sets that can increase/decrease the level of difficulty faced by methods designed to extract the semantics of soft or subjective attributes. Second, synthetic data generation provides us with a “ground truth” against we can test (i) the quality of

our learned CAV representations of soft attributes and (ii) the effectiveness of our elicitation methods at using soft, subjective attributes to improve recommendations.

The generative user-response model is implemented using RecSim NG [34], a platform for simulating user behavior when interacting with RSs that supports the authoring of structured, graphical and causal models of agent behavior or learning such behavior from data. (We use the former capability.)

We start by describing the process for generating ratings and tags for “non-subjective” tags, where users have linear utility for the corresponding soft attributes. We then describe mild modifications of this core generative model to allow for subjective tags (both degree and sense subjectivity) and nonlinear attribute utility.

**Non-subjective, linear utility model.** The generative process proceeds in the following stages: we first generate items (with their latent and soft attribute values and other properties); then users (with their utility functions and other behavioral characteristics); then user-item ratings; and finally user-item tags. The model reflects realistic characteristics such as item and user “clustering,” popularity bias, not-missing-at-random ratings, the sparsity of ratings, the relative sparsity of tags compared to ratings, etc.

Specifically each item  $i$  is characterized by an *attribute vector*  $\mathbf{v}(i) \in [0, 1]^D$ , where  $D = L + S$ :  $L$  dimensions correspond to latent item features and  $S$  to soft attributes. For a soft dimension  $L < s \leq L + S$ ,  $v^s(i)$  captures the degree to which  $i$  exhibits soft attribute  $s$ . We sample  $m$  items from a mixture of  $K$   $D$ -dimensional Gaussian distributions (truncated on  $[0, 1]^D$ )  $\mathcal{N}(\mu_k, \sigma_k)$ ,  $k \leq K$ , with mean vector  $\mu_k \in [0, 1]^D$  and (diagonal) covariance  $\sigma_k$ . We set  $D$  to 25 and  $K$  to 100 in our experiments. For simplicity, we assume all  $\sigma_k$  are identical to 0.5, and sample means uniformly. Mixture weights are sampled uniformly at random and normalized. For each item  $i$ , we also randomly generate a popularity bias  $b_i$  from  $[0, 1]$  (whose purpose is described below).

Each user  $u$  has a *utility vector*  $\mathbf{w}(u) \in [0, 1]^D$  reflecting its utility for items. We sample  $n$  users from the above  $K$ -mixture-of-Gaussian distribution similar to that for items. Here the means and variances of these distributions are same as the ones used for item distributions, but the mixture weights are fully resampled. This step ensures that the generated samples of users and items are distributed in different parts of the latent “topic space”.

Next, we generate the user-item ratings with the following steps:

- (i) For each  $u$ , we draw  $Num_u$  samples from a Zipf (or zeta) distribution with a power parameter  $a = 1.05$  to reflect the natural power law over the number of ratings provided by users. Parameter  $a$  is chosen in a way that the average number of rated items by each user is approximately 100. We also set the maximum number of ratings by each user to 1,000.
- (ii) To generate the candidate items to be rated by each user  $u$ , we generate a set of  $Rated_u$  items, by sampling them without replacement from the overall set of items via a multinomial logit (or softmax) choice model, where the probability associated with each item  $i$  is proportional to  $e^{\tau \cdot (\mathbf{w}_u \mathbf{v}_i + b_i)}$ . Here  $\tau$  is a temperature parameter that controls the degree of randomness in choosing the item wrt greatest affinity, and we use  $\tau = 1$  in our experiments.
- (iii) For each user  $u$  and item  $i \in Rated_u$ , the rating  $r_{ui}$  is generated as follows. We denote by  $s(u, i) := \mathbf{w}_u \mathbf{v}_i + \varepsilon$  be the score of item  $i$ , where  $\varepsilon$  is a small, zero-mean random noise. We then discretize all the scores provided by user  $u$  into 5 equally sized sub-intervals in  $[\min_u, \max_u]$ , where  $\min_u = \min\{s(u, i) : i \in Rated_u\}$  and  $\max_u = \max\{s(u, i) : i \in Rated_u\}$  and assign a 1 to 5 rating to each item  $i \in Rated_u$  accordingly.

For each soft attribute  $s$  we assume there is a unique tag  $g_s$  that users can apply when referring to that attribute. For each generic tag  $g$ , we denote by  $s(g)$  the corresponding soft attribute (so  $g = g_{s(g)}$ ). To complete the data-generation procedure we also generate user-item tags with the following steps.



- (i) For each user  $u$ , we generate  $PT_u$ , the probability of tagging an item, from a mixture of two distributions: (a) a Dirac distribution at 0 with weight  $0 < x < 1$ ; and (b) a Uniform distribution over  $[p_-, p_+]$ , where  $0 < p_- \leq p_+$ , with weight  $1 - x$ . This reflects the fact that a large fraction of users never use tags, and among those who do, some users tag much more frequently than others. In our experiments, we set  $x$  to 0.8,  $p_-$  to 0.1, and  $p_+$  to 0.5.
- (ii) For each user  $u$  with non-zero tag probability, i.e.,  $PT_u > 0$ , we first generate the set  $Tagged_u$ , which represents the items that are tagged by  $u$ . Here  $Tagged_u$  is a subset of rated items  $Rated_u$  such that each rated item will be tagged with (independent) probability  $PT_u$ . This reflects the fact that a user will not tag an unrated movie, but may leave some rated movies untagged.<sup>20</sup> For any item  $i \notin Tagged_u$ , the corresponding indicator value  $t_{u,i,g} = 0$  for every tag  $g$ , which means that no tag is applied by user  $u$  on item  $i$ .
- (iii) For every (non-subjective) tag  $g$ , we use a user-independent threshold  $\tau_g = 0.5$  indicating the degree to which an item must possess attribute  $s(g)$  to be tagged with tag  $g$  by a user.
- (iv) For every item  $i \in Tagged_u$  and tag  $g$ , we set the indicator value  $t_{u,i,g} = 1$  (i.e., user  $u$  applied tag  $g$  to item  $i$ ) if  $v^{s(g)}(i) \geq \tau_g + \varepsilon$  (where  $\varepsilon$  is a small, zero-mean random noise drawn independently from  $\mathcal{N}(0, 0.01)$  for each  $(u, i, g)$ ). Otherwise the indicator value  $t_{u,i,g}$  remains at 0.

**Subjective model.** The generative model above is modified in a straightforward way to handle subjective attributes. For degree subjectivity, we generate user-dependent tag-application thresholds  $\tau_g^u$  for each user-tag pair, rather than user-independent thresholds  $\tau_g$ . In general, to allow for some “commonality” across user sub-populations, we draw these thresholds from mixture distributions with a small number of components and small variance. Threshold distributions that are widely dispersed can be viewed as “fully” subjective, while for objective attributes their thresholds (as above) are special cases for which each of their distributions follow a Dirac at  $\tau_g$ . In our experiments, for each degree-subjective tag the threshold is randomly chosen between 0.5 and 0.7.

For sense subjectivity, the model is as above with the following modification. We maintain  $S_{obj}$  soft attributes of the form above—which we now call *objective*—each  $s \in S_{obj}$  corresponds to one item dimension and to a specific *objective* (in sense) tag  $g_s$ . In addition, we have  $S_{subj}$  *subjective* soft attributes, partitioned into *tag groups*,  $S^1, \dots, S^J$  under the following condition: (a)  $S^i \cap S^j = \emptyset$  for  $i \neq j$ ; (b)  $\cup_{j \leq J} S^j = S_{subj}$ ; and (c)  $|S^j| > 1$  for all  $j \leq J$ . Each tag group  $S^j$  is associated with a single tag  $g^j$ , with each  $s \in S^j$  reflecting a different *sense* for  $g^j$ .

For each tag group  $S^j$ , each user  $u$  is randomly assigned to exactly one such sense  $s(u, j) \in S^j$ . This has two implications. First, when user  $u$  considers applying tag  $g^j$  to an item, it is evaluated according to soft attribute  $s(u, j)$ . This means that  $u$  uses that specific sense when applying that tag. Second, the utility vector  $\mathbf{w}(u)$  of user  $u$  is such that its  $s^{th}$  component is zero for each  $s \in S^j$  except for  $s(u, j)$ . This implies that  $u$  assesses her utility for an item using only her designated attribute (or sense) from each of the tag groups.

**Nonlinear utility model.** There are many forms of nonlinearity. We propose one especially simple form that allows one to test whether our CAV method (or any other soft-attribute method) can identify such attributes. Specifically, we consider the simple single-peaked utility function, and for simplicity we only describe the utility function for non-subjective attributes. Extending it to the subjective case is straightforward.

Assume the domain of any arbitrary attribute  $1 \leq a \leq D$  is  $[0, 1]$ , and the user utility is *additive-independent* across attributes, i.e., the utility for an item is the sum of “local utilities” for each attribute (the dot-product model satisfies this trivially). A user  $u$ ’s utility function is *single-peaked* with respect to  $s$  if  $u$  has an ideal point  $p_{u,a} \in [0, 1]$  such that  $u$ ’s local utility for the attribute  $l_{u,a}(x)$  is maximized at  $p_{u,a}$  and decreases monotonically as  $x$  moves away from  $p_{u,a}$ . The

<sup>20</sup>One could also allow, if desired, the propensity to tag to vary with the tag  $g$ , and/or bias the application of tags to higher-rated items.

functional form of a single-peaked utility can be arbitrary with the simplest form being a piecewise linear function. Here we use  $l_{u,a}(x) = p_{u,a} - |x - p_{u,a}|$ , where  $x = \mathbf{w}_{u,a} \mathbf{v}_i$ . As both vectors  $\mathbf{w}_u$  and  $\mathbf{v}_i$  are in  $[0, 1]^D$ , it is immediate to see that  $x \in [0, 1]$ . We sample  $p_{u,a}$  from a uniform distribution  $U(L_a, 1)$ , where  $L_a$  is a user-specified per-attribute parameter. In the special case when  $L_a = 1$  then  $U(L_a, 1)$  becomes the linear utility. In our experiments, we set  $L_a$  to 0.3 for sense-subjective tags and  $L_a$  to 0.5 for the rest.

### A.3 MovieLens-20m Data

The MovieLens-20m dataset consists of 20 million movie ratings on a scale from 1 to 5 and 465,000 free form tag applications for 27,000 movies by 138,000 users. For our analysis, we transform all tags to lowercase and filter tag-and-rating data to only include the user-item-tags whose corresponding ratings are at least 4. This results in around 235,000 user-item-tag triples, which comprise 20,068 unique tags. Many of these are applied to only a few movies or by a few users: 11,145 tags are only applied by a single user, and just 268 tags are applied to at least 50 unique movies. Since the CAVs are trained in 50-dimensional latent space, we restrict the CAV training data to the top 250 tags in terms of unique tagged movies. Inspection of the tag data further shows that tags that are applied by only a few users tend to be overly-specific or overly-generic rather than descriptive of the particular movies tagged. For example, the tags ‘memasa’s movies’, ‘on dvr’ and ‘bd-video’ were applied by single users to 216, 210 and 192 movies, respectively. The tags ‘vhs’ and ‘owned’ were applied to 194 and 155 movies by 20 individual users. To exclude these types of tags, we further filter the data to include only the top 250 tags in terms of unique users who have used the tag at least once. This leaves us with a total of 164 tags for evaluation. User-item-tag triples are then split into train-test with a roughly (0.75, 0.25) split, with *all examples* for any specific user-item pair present in exactly one of these subsets, i.e., if user  $u$  rated item  $i$ , the all tags  $g$  applied by  $u$  to  $i$  (if any) will only be in either train or test dataset.

### A.4 More Details on Training Procedures

We provide additional details on the training methods used in our experiments.

**A.4.1 Weighted Alternating Least Squares.** We learn user and item embeddings using CF. We consider two approaches, matrix factorization using *weighted alternating least squares* (WALS) [20], which we describe in this subsection and a two-tower DNN (next subsection). CF typically generates with a low-rank approximation of the user-item ratings matrix, which models both users and items in the low-dimensional (latent) feature space. Both users and items are represented by feature vectors in  $X \subset \mathbb{R}^d$ , where we let  $\phi_U$  (resp.,  $\phi_I$ ) be the mapping from users (resp., items) to features.

In our experiments with linear utility models, we learn  $(\phi_U, \phi_I)$  using WALS which uses the following regularized objective:

$$(\phi_U^*, \phi_I^*) \in \arg \min \sum_{u,i} c_{u,i} (\hat{r}_{u,i} - r_{u,i})^2 + \kappa (\|\phi_U\|^2 + \|\phi_I\|^2), \quad (6)$$

where  $c_{u,i}$  is the confidence weight of the predicted rating  $\hat{r}_{u,i}$  and  $\kappa > 0$  is the regularization parameter. We pick the feature vectors  $(\phi_U^*, \phi_I^*)$  based on the best validation loss and train with an item-oriented confidence weight, i.e.,  $c_{u,i} \propto m - \sum_u r_{u,i}$ , which assigns lower weight to less-frequently rated or lower-rated items. Example confidence weights include: (i) uniform, i.e.,  $c_{u,i} = \delta$  for some  $\delta > 0$  for missing entries  $(u, i)$  and  $c_{u,i} = 1$  otherwise; (ii) user-oriented, i.e.,  $c_{u,i} \propto \sum_i r_{u,i}$  which assigns higher confidence to users with more ratings, assuming he/she has greater item familiarity; and (iii) item-oriented, i.e.,  $c_{u,i} \propto m - \sum_u r_{u,i}$ , which assigns lower weight to less-frequently rated or lower-rated items.

In our experiments with linear utility models, we train a CF model  $\Phi = (\phi_U, \phi_I)$  using *WALS* given ratings  $\mathbf{R}$ , in order to obtain user and item feature vectors. We set the regularization parameter  $\kappa$  to 250 for MovieLens and 1 for the synthetic data. The number of *WALS* iterations is 100. We train with the third option for confidence scores and pick the feature vectors  $(\phi_U^*, \phi_I^*)$  based on the best validation loss.

**A.4.2 DNN Training.** When training nonlinear CAVs (i.e., when user utility is nonlinear with respect to the soft attribute in question), user and item embeddings  $\phi_U$  and  $\phi_I$  are generated by DNNs,  $N_U$  and  $N_I$ , respectively, and (as above) predicted ratings are generated by taking dot products, using a “two-tower” architecture. following deep neural network (DNN) models. Letting  $\theta_U, \theta_I$  denote the parameters of  $N_U, N_I$ ,<sup>21</sup> similar to the linear case, we train the two-tower model by minimizing the regularized (squared) RMSE loss:

$$(\theta_U^*, \theta_I^*) \in \arg \min_{\theta_U, \theta_I} \text{RMSE}^2(\theta_U, \theta_I) + \kappa(\|\phi_U(\cdot; \theta_U)\|^2 + \|\phi_I(\cdot; \theta_I)\|^2) + \rho(\|\theta_U\|^2 + \|\theta_I\|^2), \quad (7)$$

where  $\text{RMSE}(\theta_U, \theta_I) = \sqrt{\frac{1}{N} \sum_{u,i} (\hat{r}_{u,i}(\theta_U, \theta_I) - r_{u,i})^2}$ , the predicted rating is  $\hat{r}_{u,i}(\theta_U, \theta_I) = \phi_U^\top(u; \theta_U) \phi_I(i; \theta_I)$ ,  $\kappa > 0$  is the activity regularization constant, and  $\rho > 0$  is the weight regularization constant. Since the above objective function is nonlinear and nonconvex, we simply optimize this objective function with stochastic gradient descent based approaches, e.g., ADAM [23].

We use one-hot encodings of users and items as inputs to  $N_U$  and  $N_I$ , respectively. For simplicity, we use the same architecture for both the user DNN and item DNN, which is a feed-forward network with  $\ell$  layers, in which the first layer has no bias term and has a dimension of  $d$ —we let  $d$  have the same dimension as the linear embedding space (25 for synthetic data, 50 for MovieLens). The main motivation for this architectural choice is to allow weight initialization of the first layer using the linear user and item embeddings as a form of warm start. In our experiments, we use a DNN architecture with  $\ell = 3$  fully connected layers, all of size (width)  $d$ , and ReLU activations. As in the linear case, we set the activity regularization parameter  $\kappa = 250$  for MovieLens and  $\kappa = 1$  for the synthetic data. For weight regularization, we set  $\rho = 1$  in all the experiments. Given the trained DNNs, the activation vectors for training (item) CAVs are simply the  $\ell$ -th intermediate activation layer  $N_{I,\ell}(i)$ , extracted from the trained item DNN. We treat the choice of intermediate layer  $\ell$  as a tunable hyper-parameter chosen to optimize the downstream task (CAV prediction).

**A.4.3 PITF Training.** When evaluating CAVs for their ability to predict tag usage, we use the *PITF* (pairwise interaction tensor factorization) algorithm [43] as a natural baseline, since it is a method designed to predict (or recommend) tags for users. We train PITF using tag triples  $t_{u,i,t}$  with the same train-test split as in CAV training. Notice that positive tags are generally more sparse in all datasets, to balance the portion of positive and negative samples for PITF training we also use the same negative sampling methodology as in CAV training. For fairer evaluation, we use the same embedding dimension as in CAV for PITF training (50 for MovieLens, 25 for synthetic data), but notice that the resulting PITF model is still much larger than the CAV model since PITF learns an embedding for each tag-user pair and tag-item pair, while CAV embedding is user-independent. In evaluating the accuracy of PITF, we do not adopt the F-score over Top-N items as in [43], but instead we check if  $\text{sign}(y_{u,i,g}) = t_{u,i,g}$ , where  $y_{u,i,g}$  is the prediction of PITF. We employ the implementation at <https://github.com/yamaguchi-yuto/pitf/> to train the PITF embedding. For hyper-parameters, we set the learning rate ( $\alpha$  in [43]) to 0.001 for MovieLens and 0.0002 for synthetic data, the regularization parameter ( $\lambda$  in [43]) to 0.01 for MovieLens and 0.00005 for synthetic data. These parameters are tuned with a validation set. The number of PITF training iterations is set to 100.

<sup>21</sup>We remove dependence of various terms (e.g.,  $\phi_U, \phi_I, \hat{r}_{u,i}$ ) on the DNN parameters  $\theta_U, \theta_I$  to unclutter the notation when this dependence is obvious.

### A.5 Additional Critiquing Details

We fill in a few additional details of the critiquing experiments in Sec. 5. We emphasize again that our aim is not to evaluate state-of-the-art critiquing and elicitation methods, but to demonstrate the use of CAV-based soft-attribute semantics in allowing users to effectively interact with the RSs and refine the recommendation results.

In the synthetic data experiment, we construct each user’s *estimated ideal item* using the knowledge of her ground-truth utility function, which is either linear or single-peaked linear in each (latent or soft-attribute) dimension. We note that a user’s (actual or estimated) “ideal” item may not actually exist in the item corpus  $\mathcal{I}$ . Insisting that the user’s ideal item exist is unrealistic, since it requires a dense item space. Moreover, the user generally does not know the identify of the actual best item in  $\mathcal{I}$ , since this would assume too great a state of knowledge for most users. Instead, we assume each  $u$  has a rough estimate of the maximum and minimum levels any tag/attribute can attain in the item corpus and uses this to drive her critiques. (For example, we truncate sampled items to  $[0, 1]^D$ .) Note that when user utility is linear, the ideal item must occur at the boundary of item space, so the estimates inform her estimated ideal. In the MovieLens experiment, we use the ratings data to derive an estimated ground-truth utility for each *test user* (who must have rated sufficiently many items as described in the main text).

During the critiquing process, the RS updates its user embedding based on the user’s response.<sup>22</sup> We assume item embeddings  $\phi_I(i)$  are fixed, and use a *simple heuristic RS strategy* for incorporating critiques.<sup>23</sup> Given a user embedding  $\phi_U(u)$ , the RS scores all items  $i$  in the corpus with respect to utility  $r_{i,u} = \phi_I(i)^T \phi_U(u)$ , and presents the slate  $S$  of the  $k$  top scoring items.

Suppose at step  $t > 0$  the user critiques  $S$  with a specific tag  $g$  (and a specific direction, *more* or *less*). The RS updates the user embedding in response to this critique using a simple heuristic update function:  $\phi_U(u) \leftarrow \phi_U(u) + \text{Sgn} \cdot \alpha_t(g) \cdot \phi_g$ . Here  $\text{Sgn} \in \{+1, -1\}$  indicates the direction of the move (+1 more, -1 less), and  $\alpha_t(g)$  is a step size that controls the scale of the move of the user embedding in the direction of tag  $g$ ’s CAV. For each  $g$ , we decay the step size  $\alpha_t(g)$  with each critique using  $g$ ,  $\alpha_t(g) = \alpha_0(g)/(1 + t)$ , where  $\alpha_0$  is  $g$ ’s initial step size. This ensures that the updating process converges to a stable point (and does not cycle or repeat slates of items) given the coarse control mechanism offered to the user. If the tag  $g$  is sense-subjective, the critique is interpreted relative to the RS’s estimate of  $u$ ’s sense (or user cluster) based on past usage.

In our experiments,  $\alpha_0(g)$  is constant ( $\alpha_0$ ) across all tags, and we treat it as a tunable hyper-parameter selected to optimize user utility metrics such as *UMU* and *UAU* utilities.<sup>24</sup> The critiquing results we report are based on the set of hyper-parameters optimized using a validation set.

<sup>22</sup>We use the average of all learned user embeddings as the RS’s prior.

<sup>23</sup>We emphasize that the RS strategy used and method for incorporating critiques are fairly generic and are not intended to reflect the state-of-the-art, since our goal is to measure the ability to exploit learned CAVs. More elaborate strategies for updating user embedding are possible, including the use of Bayesian updates relative to a prior over the user embedding [47]. Here instead we adopt a simple heuristic, based on [28], to focus attention on the CAV semantics itself.

<sup>24</sup>Ultimately, this heuristic adjustment should be tuned to the specifics of a real-world user response models.