Personalized Visualization Recommendation

XIN QIAN, University of Maryland, USA
RYAN A. ROSSI, Adobe Research, USA
FAN DU, Adobe Research, USA
SUNGCHUL KIM, Adobe Research, USA
EUNYEE KOH, Adobe Research, USA
SANA MALIK, Adobe Research, USA
TAK YEON LEE, KAIST, South Korea
NESREEN K. AHMED, Intel Labs, USA

Visualization recommendation work has focused solely on scoring visualizations based on the underlying dataset, and not the actual *user* and their past visualization feedback. These systems recommend the same visualizations for every user, despite that the underlying user interests, intent, and visualization preferences are likely to be fundamentally different, yet vitally important. In this work, we formally introduce the problem of *personalized visualization recommendation* and present a generic learning framework for solving it. In particular, we focus on recommending visualizations personalized for each individual user based on their past visualization interactions (*e.g.*, viewed, clicked, manually created) along with the data from those visualizations. More importantly, the framework can learn from visualizations relevant to other users, even if the visualizations are generated from completely different datasets. Experiments demonstrate the effectiveness of the approach as it leads to higher quality visualization recommendations tailored to the specific user intent and preferences. To support research on this new problem, we release our user-centric visualization corpus consisting of 17.4k users exploring 94k datasets with 2.3 million attributes and 32k user-generated visualizations.

CCS Concepts: • **Information systems** → **Personalization**; **Recommender systems**; **Web searching and information discovery**; **Web mining**; *Multimedia and multimodal retrieval.*

Additional Key Words and Phrases: Visualization recommendation, user personalization, user modeling, data visualization, machine learning

ACM Reference Format:
Xin Qian, Ryan A. Rossi, Fan Du, Sungchul Kim, Eunyee Koh, Sana Malik, Tak Yeon Lee, and Nesreen K. Ahmed. 2022. Personalized Visualization Recommendation. *ACM Trans. Web* 0, 1, Article 00 (August 2022), 47 pages. https://doi.org/0000001.0000001

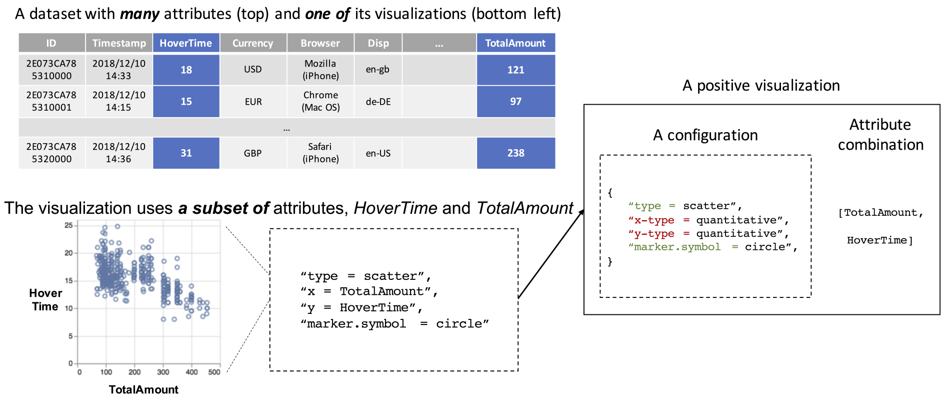
1 INTRODUCTION

With massive datasets becoming ubiquitous, visualization recommendation systems have become increasingly important. These systems have the promise of enabling rapid visual analysis and exploration of such datasets. However, existing end-to-end visualization recommendation systems output a long list of visualizations based solely on simple visual rules [Wongsuphasawat et al. 2015, 2017]. These systems lack the ability to recommend visualizations that are personalized to the specific user and the tasks that are important to them. This makes it both time-consuming and difficult for users to effectively explore such datasets and find meaningful visualizations.

Authors' addresses: Xin Qian, University of Maryland, College Park, MD, USA, xinq@umd.edu; Ryan A. Rossi, Adobe Research, San Jose, CA, USA, rrossi@adobe.com; Fan Du, Adobe Research, San Jose, CA, USA, fdu@adobe.com; Sungchul Kim, Adobe Research, San Jose, CA, USA, skim@adobe.com; Eunyee Koh, Adobe Research, San Jose, CA, USA, eunyee@adobe.com; Sana Malik, Adobe Research, San Jose, CA, USA, sana.malik@adobe.com; Tak Yeon Lee, KAIST, Seoul, South Korea, takyeonlee@kaist.ac.kr; Nesreen K. Ahmed, Intel Labs, Santa Clara, CA, USA, nesreen.k.ahmed@intel.com.

2022. 1559-1131/2022/8-ART00 $15.00
https://doi.org/0000001.0000001

Recommending visualizations that are *personalized* to a specific user is an important unsolved problem. Prior work on visualization recommendation have focused mainly on rule-based or ML-based approaches that are completely agnostic to the user of the system. In particular, these systems recommend the same ranked list of visualizations for every user, despite that the underlying user interests, intent, and visualization preferences are fundamentally different, yet vitally important for recommending useful and interesting visualizations for a specific user. The rule-based methods use simple visual rules to score visualizations whereas the existing ML-based methods have focused solely on classifying design choices [Hu et al. 2019a] or ranking such design choices [Moritz et al. 2018] using a corpus of visualizations that are *not* tied to a user. Neither of these existing classes of visualization recommendation systems focus on modeling individual user behavior nor personalizing for individual users, which is at the heart of our work.


A dataset with ***many*** attributes (top) and ***one of*** its visualizations (bottom left)
The visualization uses ***a subset of*** attributes, *HoverTime* and *TotalAmount*
Hover Time
TotalAmount
A positive visualization
A configuration
Attribute combination

ID	Timestamp	HoverTime	Currency	Browser	Disp	...	TotalAmount
2E073CA78 5310000	2018/12/10 14:33	18	USD	Mozilla (iPhone)	en-gb		121
2E073CA78 5310001	2018/12/10 14:15	15	EUR	Chrome (Mac OS)	de-DE		97
...							
2E073CA78 5320000	2018/12/10 14:36	31	GBP	Safari (iPhone)	en-US		238

```
"type = scatter",
"x = TotalAmount",
"y = HoverTime",
"marker.symbol = circle"
```

```
{
   "type = scatter",
   "x-type = quantitative",
   "y-type = quantitative",
   "marker.symbol = circle",
}
```

```
[TotalAmount,
 HoverTime]
```

Fig. 1. Visualization to Data Independent Visual Configuration. A visualization consists of the data attributes along with a set of design choices, and thus are dataset dependent. In this work, we propose the notion of a visual configuration that removes the data dependency of a visualization while capturing the visual design choices. Notably, visual configurations naturally generalize across datasets, since they are independent of the dataset, and thus unlike visualizations, a visual configuration *can* be shared among users that use entirely different datasets. The visualization shown in the above toy example uses only two data attributes from the dataset. Note actual visual configurations have many other design choices which are not shown in the toy example above for simplicity.

In this work, we introduce a new problem of *personalized visualization recommendation* and propose an expressive framework for solving it. The problem studied in this work is as follows: Given a set of n users where each user has their own specific set of datasets, and each of the user datasets contains a set of user-relevant visualizations (*i.e.*, visualizations a specific user has interacted with in the past, either implicitly by clicking/viewing or explicitly by liking or adding the visualization to their favorites or a dashboard they are creating), the problem of *personalized visualization recommendation* is to learn an individual recommendation model for every user such that when a user selects a possibly new dataset of interest, we can apply the model for that specific user to recommend the top most relevant visualizations that are most likely to be of interest to them (despite that there is no previous implicit/explicit feedback on any of the visualizations from the new dataset). Visualizations are fundamentally tied to a dataset as they consist of the (i) set of visual design choices (*e.g.*, chart-type, color/size, x/y) and the (ii) subset of data attributes from the full dataset used in the visualization. See Figure 1 for an example. Therefore, how can we develop a learning framework for solving the personalized visualization recommendation problem that is able to learn from other users and their relevant visualizations, even when those visualizations are from tens of thousands of completely different datasets with no shared attributes?

We develop a model that takes as input the visualizations generated by the user, which serves as known ground-truth. The model is then trained with these visualizations, and outputs a list of visualizations for that specific user along with a score indicating the strength of that visualization for the specific user. While we use the user-generated visualizations in this work since they serve as ground-truth, our model can also use a variety of other types of implicit and explicit user feedback as discussed in Section 2.2 in more detail. During training, each visualization from a user is decomposed into a visualization configuration and the data-attributes used in it.

There are two important and fundamental issues at the heart of the personalized visualization recommendation problem. First, since visualizations are defined based on the attributes within a single user-specific dataset, there is no way to leverage user-feedback for visualizations across different datasets. This is due to the fact that the space of visualizations for any users dataset is disjoint (and users typically do not share datasets), see Section 2 for further details. Second, since each user often has their own dataset of interest (not shared by other users), there is no way to leverage user-feedback across different datasets. In this work, we address both problems. Notably, the framework proposed in this paper naturally generalizes to the following problem settings: (a) single dataset with a single set of visualizations shared among all users, and (b) tens of thousands of datasets that are not shared between users where each dataset of interest to a user gives rise to a completely different set of possible visualizations. However, the existing work cannot be used to solve the new problem formulation that relaxes the single dataset assumption to make it more general and widely applicable.

In the problem formulation of personalized visualization recommendation, each user can have their own set of datasets, and since each visualization represents a series of design choices and data (*i.e.*, attributes tied to a specific dataset), then this gives rise to a completely disjoint set of visualizations for each user. Hence, there is no way to directly leverage visualization feedback from other users, since the visualizations are from different datasets. Furthermore, visualizations are dataset specific, since they are generated based on the underlying dataset, and therefore any feedback from a user cannot be directly leveraged for making better recommendations for other users and datasets. To understand the difficulty of the proposed problem of personalized visualization recommendation, the equivalent problem with regards to traditional recommender systems would be as if each user on Amazon (or Netflix) had their own separate set of disjoint products (or movies) that no other user could see and provide feedback. In such a setting, how can we then use feedback from other users? Furthermore, given a single dataset uploaded by some user, there are an exponential number of possible visualizations that can be generated from it. This implies that even if there are some users interested in a single dataset, the amount of preferences by those users is likely to be extremely small compared to the exponential number of possible visualizations that can be generated and preferred by such users.

To overcome these issues and make it possible to solve the personalized visualization recommendation problem, we introduce two new models and representations that enable learning from dataset and visualization preferences across different datasets and users, respectively. First, we propose a novel model and representation that encodes users, their interactions with attributes (i.e., attributes in any dataset) and we map every attribute to a shared k-dimensional meta-feature space that enables the model to learn from *user-level data preferences* across all the different datasets of the users. Most importantly, the shared meta-feature space is independent of the specific datasets and the meta-features represent general functions of an arbitrary attribute, independent of the user or dataset that it arises. This enables the model to learn from user-level data preferences, despite that those preferences are on entirely different datasets. Second, we propose a novel *user-level* *visual preference* graph model for visualization recommendation using the proposed notion of a visualization configuration (Figure 1) that enables learning from user-level visual preferences across different datasets and users. Importantly, the graph model is able to directly learn from user-level visual preferences across different datasets. This model encodes users and their visual-configurations (sets of design choices). Since each visual-configuration node represents a set of design choices that are by definition not tied to a user-specific dataset, then the proposed model can use this user-level visual graph to infer and make connections between other similar visual-configurations that are also likely to be useful to that user. This new graph model is critical since it allows the learning component to learn from user-level visual preferences (which are visual-configurations) across the different datasets and users. Without this novel component, there would be no way to learn from other users visual preferences (sets of design choices). For the experiments in Section 3, we train the model with actual visualizations that users have created by hand, which serve as ground-truth. The visualizations can be from one or more datasets. The model then outputs a list of new unseen visualizations for that specific user.

1.1 Summary of Contributions

This work makes the following key contributions:

- **Problem Formulation:** We introduce and formulate the problem of *personalized visualization recommendation* that learns a personalized visualization recommendation model for every individual user based on their past visualization feedback, and the feedback of other users and their relevant visualizations from completely different datasets. Our formulation removes the unrealistic assumption of a single dataset shared across all users (and thus that there exists a single set of dataset-specific visualizations shared among all users). To solve this problem, the model must be able to learn from the visualization and data preferences of many users across tens of thousands of different datasets.
- **Framework:** We propose a flexible framework that expresses a class of methods for the *personalized visualization recommendation* problem. To solve this new problem, we introduce new graph representations and models that enable learning from the visualization and data preferences of users despite them being in different datasets entirely. More importantly, the proposed framework is able to exploit the visualization and data preferences of users across tens of thousands of different datasets.
- **Effectiveness:** The extensive experiments demonstrate the importance and effectiveness of learning personalized visualization recommendation models for each individual user. Notably, our personalized models perform significantly better than SOTA baselines with a mean improvement of 29.8% and 64.9% for HIT@5 and NDCG@5, respectively. Furthermore, the deep personalized visualization recommendation models are shown to perform even better. Finally, comprehensive ablation studies are performed to understand the effectiveness of the different learning components.

1.2 Organization of article

First, we introduce a new problem of visualization recommendation in Section 2 that learns a personalized model for each of the n individual users by leveraging a large collection of datasets and relevant visualizations from each of the datasets in the collection. Notably, the learning of the individual user models are able to exploit the preferences of other users (even if the preferences are on a completely different dataset) including the data attributes used in a visualization, visual design choices, and actual visualizations generated despite that no other user may have used the underlying dataset of interest. In Section 3, we propose a computational framework for solving the new problem of personalized visualization recommendation. Further, we also propose *deep personalized visualization recommendation models* in Section 4 that are able to learn complex non-linear functions between the embeddings of the users, visualization-configurations, datasets and the data attributes used in the visualizations. Next, Section 5 describes the user-centric visualization corpus we created and made publicly accessible for studying this problem. Then Section 6 provides a comprehensive and systematic evaluation of the proposed approach and framework for the personalized visualization recommendation problem while Section 8 discusses related work. Finally, Section 9 concludes with a summary of the key findings and briefly discusses directions for future work on this new problem.

2 PERSONALIZED VISUALIZATION RECOMMENDATION

In this section, we formally introduce the *Personalized Visualization Recommendation* problem.

2.1 Overview

The personalized visualization recommendation problem has two main parts: (1) training a personalized visualization recommendation model for every user $i \in [n]$ (Section 2.3), and (2) leveraging the user-personalized model to recommend personalized visualizations based on the users past dataset and visualization feedback/preferences (Section 2.4).

(1) **Personalized Model Training (Sec. 2.3):** Given a user-level training visualization corpus $\boldsymbol{\mathcal{D}} = \{(\boldsymbol{\mathcal{X}}_i, \mathbb{V}_i)\}_{i=1}^{n}$ consisting of n users and their corresponding datasets of interest $\boldsymbol{\mathcal{X}}_i = \{\mathbf{X}_{i1}, \ldots, \mathbf{X}_{ij}, \ldots\}$ as well as their relevant sets of visualizations $\mathbb{V}_i = \{\boldsymbol{\mathcal{V}}_{i1}, \ldots, \boldsymbol{\mathcal{V}}_{ij}, \ldots\}$ for those datasets, we first learn a user-level personalized model $\boldsymbol{\mathcal{M}}$ from the training corpus $\boldsymbol{\mathcal{D}}$ that best captures and scores the effective visualizations for user i highly while assigning low scores to visualizations that are likely to not be preferred by the user.

(2) **Recommending Personalized Visualizations (Sec. 2.4):** Given a user $i \in [n]$ and a dataset $\mathbf{X}_{ij}$ of interest to user i, we use the trained personalized visualization recommendation model $\boldsymbol{\mathcal{M}}$ for user i to generate, score, and recommend the top visualizations of interest to user i for dataset $\mathbf{X}_{ij}$. Note that we naturally support the case when the dataset $\mathbf{X}_{ij} \notin \boldsymbol{\mathcal{X}}_i$ is new or when the dataset $\mathbf{X}_{ij} \in \boldsymbol{\mathcal{X}}_i$ is not new, but we have at least one or more previous user feedback about the visualizations the user likely prefers from that dataset.

The fundamental difference between the ML-based visualization recommendation problem introduced in [Qian et al. 2020] and the personalized visualization recommendation problem described above is that the personalized problem focuses on modeling the behavior, data, and visualization preferences of individual users. Since local visualization recommendation models are learned for every user $i \in [n]$ (as opposed to training a global visualization recommendation model), it becomes important to leverage every single piece of feedback from the users. For instance, global visualization recommendation models essentially ignore the notion of a user, and therefore can leverage all available training data to learn the best global visualization recommendation model. However, personalized visualization recommendation models explicitly leverage specific user feedback to learn the best personalized local model for every user $i \in [n]$, and there of course is far less feedback from individual users.

An overview of the system is provided in Figure 2. Users select (or upload) a dataset of interest, then the system updates the graph representations and user-specific models. The model learned for the specific user is then used to obtain personalized scores of the visualizations for the specific user at hand. The top visualizations for each user and their dataset of interest are then displayed to the user (from most relevant to least). Quality of the personalized recommendations for the user improves over time as the system continuously learns from additional implicit/explicit feedback from the user on the visualizations. Notice that if the user selects an existing dataset, we only need to use the previously learned models to infer the top-k personalized visualization recommendations for that user. Furthermore, during the prediction step where we use the models to infer rankings and recommendations that are personalized for that specific user, we can also apply the visual rule-based approach to ensure that each visualization in the top-k is actually valid with respect to the visual rule-based approach, and if not, then we remove it and use the next visualization with largest weight.

The approach can also be used when a user specifies part of the query, that is, suppose a user selects a chart-type, or requires that a specific attribute be used on the x-axis. More specifically, suppose the user is only interested in bar charts, then we simply perform an initial filtering prior to the scoring and ranking stage. Alternatively, when we are scoring the visualization candidates, we simply check if the candidate visualization is a bar-chart, and if so, we score it, otherwise it is removed from consideration. A similar filtering step can be applied to any other fields or constraints specified by the user, including any set of visual or data constraints. While the approach is obviously faster when the user specifies a single or few constraints on the visualization space, as the work required for inference and scoring of the filtered visualizations are avoided. However, as the number of constraints becomes large, the space of potential visualizations shrink, and more work is required to identify the space of such visualizations since now we must check a large number of constraints as opposed to only a few. In such rare cases, one can leverage heuristics to speedup this search process.

2.2 Implicit and Explicit User Feedback for Personalized Vis. Rec.

In this work, relevant visualizations $\boldsymbol{\mathcal{V}}_{ij} \in \mathbb{V}_i$ for a specific user i and dataset $\mathbf{X}_{ij} \in \boldsymbol{\mathcal{X}}_i$ are defined generally, as the term relevant may refer to visualizations that a user clicked, liked, generated, among many other user actions that demonstrate positive feedback towards a visualization. In terms of personalized visualization recommendation, there are two general types of user feedback: implicit or explicit user feedback. Implicit user visualization feedback corresponds to user feedback that is not explicitly stated and includes user actions such as when a user clicks on a visualization or hovers over a visualizations for more than a specific time. Conversely, explicit user feedback on a visualizations refers to feedback that is more explicitly stated about a visualization such as when a

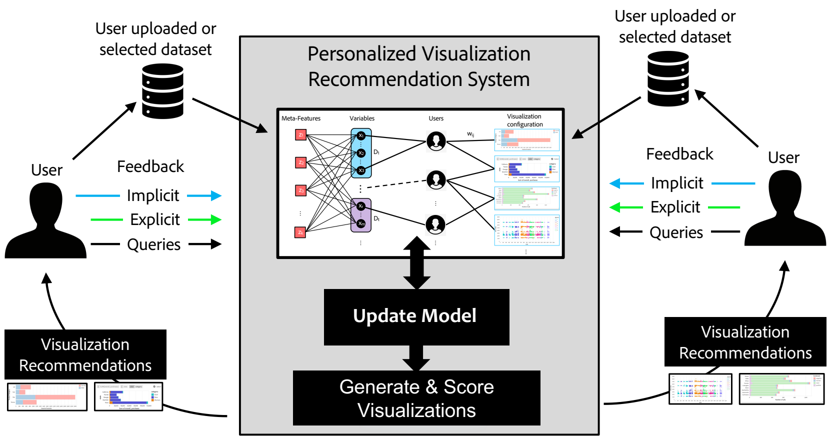

User uploaded or selected dataset
Personalized Visualization Recommendation System
User uploaded or selected dataset
Meta-Features
Variables
Users
Visualization configuration
User
Feedback
Implicit
Explicit
Queries
Feedback
User
Implicit
Explicit
Queries
Update Model
Visualization Recommendations
Generate & Score Visualizations
Visualization Recommendations

Fig. 2. Overview of the Personalized Visualization Recommendation System. See text for discussion.

user explicitly likes a visualizations, or generates a visualization. Obviously, implicit user feedback is available at a larger quantity than explicit user feedback. However, implicit user feedback is not as strong as user feedback that is explicit, *e.g.*, a user that clicked a visualization is not as strong as a user that explicitly liked a visualization.

We propose two different types of user preferences (implicit and explicit user feedback) that are important for learning personalized visualization recommendation models for individual users, including the data preferences *and* visual preferences of each individual user. For learning the data and visual preferences of a user, there is both implicit and explicit user feedback that can be used for developing personalized visualization recommender systems. There are many ways the model can leverage such user feedback during training, *e.g.*, some feedback may have an associated score such as a rating, others may simply be binary where 1 indicates positive user feedback. We now discuss the two different types of user feedback below including data user-feedback (Section 2.2.1) and visual user-feedback (Section 2.2.2).

2.2.1 Data Preferences of Users: Implicit and Explicit Data Feedback. There is naturally both implicit *and* explicit user feedback regarding the *data preferences* of users. Explicit user feedback about the data preferences of a user is a far stronger signal than implicit user feedback, however, there is typically a lot more implicit user feedback for learning than explicit feedback from the user.

- **Implicit Data Preferences of Users.** An example of *implicit feedback w.r.t. data preferences of the user* is when a user clicks (or hovers over) a visualization that uses two attributes $\mathbf{x}$ and $\mathbf{y}$ from some arbitrary user-selected dataset. We can then extract the users data preferences from the visualization by encoding the two attributes that were used in the visualization preferred by that user.
- **Explicit Data Preferences of Users.** Similarly, an example of *explicit feedback w.r.t. data preferences of the user* is when a user explicitly likes a visualization (or adds a visualization to their dashboard) that uses two attributes $\mathbf{x}$ and $\mathbf{y}$ from some arbitrary user-selected dataset. In this work, we use another form of explicit feedback based on a user-generated visualization and the attributes (data) used in the generated visualization. Hence, this is a form of explicit feedback, since the user explicitly selects the attributes and creates a visualization using them (as opposed to clicking on a visualization automatically generated by a system).

Besides using implicit and explicit feedback provided by the user based on the click or like of a visualization and the data used in it, we can also leverage an even more direct feedback about a users data preferences. For instance, many visualization recommender systems allow users to select an attribute of interest to use in the recommended visualizations. As such we can naturally leverage any feedback of this type as well.

2.2.2 Visual Preferences of Users: Implicit and Explicit Visual Feedback. In terms of visual preferences of users, there is both implicit *and* explicit user feedback that can be used to learn a better personalized visualization recommendation model for individual users. Similar to Section 2.2.1, both implicit and explicit visual user-feedback can be used by the model in various ways. For instance, the user-feedback can be included in Figure 4 as a new link between the user and visual-configuration.

- **Implicit Visual Preferences of Users.** An example of *implicit feedback* w.r.t. *visual preferences of the user* is when a user clicks (or hovers over) a visualization from some arbitrary user-selected dataset. We can then extract the users visual preferences from the visualization, and appropriately encode it for learning the visual preferences of the individual user.
- **Explicit Visual Preferences of Users.** Similarly, an example of *explicit feedback w.r.t. visual preferences of the user* is when a user explicitly likes a visualization, adds a visualization to their dashboard, or even generates a visualization on the web (*e.g.*, using plot.ly). Just as before,

we can then extract the visual preferences of the user from the visualization (mark/chart type, x-type, y-type, color, size, x-aggregate, and so on) and leverage the individual visual preferences or a combination of them for learning the user-specific personalized vis. rec. model.

2.3 Training User Personalized Visualization Recommendation Model

Given user-feedback data

$$\boldsymbol{\mathcal{D}} = \{(\boldsymbol{\mathcal{X}}_1, \mathbb{V}_1), \ldots, (\boldsymbol{\mathcal{X}}_i, \mathbb{V}_i), \ldots, (\boldsymbol{\mathcal{X}}_n, \mathbb{V}_n)\} = \{(\boldsymbol{\mathcal{X}}_i, \mathbb{V}_i)\}_{i=1}^n \tag{1}$$

where for each user $i \in [n]$, we have the set of datasets of interest to that user denoted as $\boldsymbol{\mathcal{X}}_i$ along with the sets of relevant visualizations $\mathbb{V}_i$ generated by user i for every dataset $\mathbf{X}_{ij} \in \boldsymbol{\mathcal{X}}_i$. More specifically,

$$\mathbb{V}_i = \{\boldsymbol{\mathcal{V}}_{i1}, \ldots, \boldsymbol{\mathcal{V}}_{ij}, \ldots\} \quad \text{and} \quad \boldsymbol{\mathcal{V}}_{ij} = \{\mathcal{V}_{ij1}, \ldots, \mathcal{V}_{ijk}, \ldots\} \tag{2}$$

$$\boldsymbol{\mathcal{X}}_i = \{\mathbf{X}_{i1}, \ldots, \mathbf{X}_{ij}, \ldots\} \quad \text{and} \quad \mathbf{X}_{ij} = [\mathbf{x}_{ij1}\ \mathbf{x}_{ij2}\ \cdots] \tag{3}$$

where $\mathbf{x}_{ijk}$ is the kth attribute (column vector) of $\mathbf{X}_{ij}$. Hence, the number of attributes in $\mathbf{X}_{ij}$ has no relation to the number of relevant visualizations $|\boldsymbol{\mathcal{V}}_{ij}|$ that a user i preferred for that dataset. For a single user i, the number of user preferred visualizations across all datasets of interest for that user is

$$v_i = \sum_{\boldsymbol{\mathcal{V}}_{ij} \in \mathbb{V}_i} |\boldsymbol{\mathcal{V}}_{ij}| \tag{4}$$

where $\boldsymbol{\mathcal{V}}_{ij}$ is the set of visualizations preferred by user i from dataset j. Thus, the total number of user generated visualizations across all users and datasets is

$$v = \sum_{i=1}^{n} \sum_{\boldsymbol{\mathcal{V}}_{ij} \in \mathbb{V}_i} |\boldsymbol{\mathcal{V}}_{ij}| \tag{5}$$

For simplicity, let $\mathcal{V}_{ijk} \in \boldsymbol{\mathcal{V}}_{ij} = \{\mathcal{V}_{ij1}, \ldots, \mathcal{V}_{ijk}, \ldots\}$ denote the visualization generated by user i from dataset j, that is, $\mathbf{X}_{ij} \in \boldsymbol{\mathcal{X}}_i$, specifically using the subset of attributes $\mathbf{X}_{ij}^{(k)}$ from the dataset $\mathbf{X}_{ij}$. Further, every user $i \in [n]$ is associated with a set of datasets $\boldsymbol{\mathcal{X}}_i = \{\mathbf{X}_{i1}, \ldots, \mathbf{X}_{ij}, \ldots\}$ of interest. Let $\mathbf{X}_{ij}$ be the jth dataset of interest for user i and let $|\mathbf{X}_{ij}|$ denote the number of attributes (columns) of the dataset matrix $\mathbf{X}_{ij}$. Then the number of attributes across all datasets of interest to user i is

$$m_i = \sum_{\mathbf{X}_{ij} \in \boldsymbol{\mathcal{X}}_i} |\mathbf{X}_{ij}| \tag{6}$$

and the number of attributes across all n users and all their datasets is

$$m = \sum_{i=1}^{n} \sum_{\mathbf{X}_{ij} \in \boldsymbol{\mathcal{X}}_i} |\mathbf{X}_{ij}| \tag{7}$$

Definition 1 (Space of Attribute Combinations). *Given an arbitrary dataset matrix $\mathbf{X}_{ij}$, let $\mathbb{X}_{ij}$ denote the space of attribute combinations of $\mathbf{X}_{ij}$ defined as*

$$\Sigma : \mathbf{X}_{ij} \rightarrow \mathbb{X}_{ij}, \quad \textit{s.t.} \tag{8}$$

$$\mathbb{X}_{ij} = \{\mathbf{X}_{ij}^{(1)}, \ldots, \mathbf{X}_{ij}^{(k)}, \ldots\}, \tag{9}$$

where Σ is an attribute combination generation function and every $\mathbf{X}_{ij}^{(k)} \in \mathbb{X}_{ij}$ is a different subset (combination) of attributes from $\mathbf{X}_{ij}$ consisting of one or more attributes from $\mathbf{X}_{ij}$.

Property 1. *Let $|\mathbf{X}_{ij}|$ and $|\mathbf{X}_{ik}|$ denote the number of attributes (columns) of two arbitrary datasets $|\mathbf{X}_{ij}|$ and $|\mathbf{X}_{ik}|$ of user i. If $|\mathbf{X}_{ij}| > |\mathbf{X}_{ik}|$, then $|\mathbb{X}_{ij}| > |\mathbb{X}_{ik}|$.*

It is straightforward to see that if $|\mathbf{X}_{ij}| > |\mathbf{X}_{ik}|$, then the number of attribute combinations of $\mathbf{X}_{ij}$ denoted as $|\mathbb{X}_{ij}|$ is larger than the number of different attribute subsets that can be generated from $\mathbf{X}_{ik}$ denoted as $|\mathbb{X}_{ik}|$. Property 1 is important as it characterizes the space of attribute combinations/subsets for a given dataset $\mathbf{X}_{ij}$ and therefore can be used to understand the corresponding space of possible visualizations that can be generated from a given dataset, as these are also tied.

In this work, we assume a visualization is specified using some grammar such as Vega-Lite [Satyanarayan et al. 2016]. Therefore, the data mapping and design choices of the visualization are encoded in a json-like format, and can easily render a visualization. A visualization configuration C (set of design choices) *and* the data attributes $\mathbf{X}_{ij}^{(k)}$ selected from a dataset $\mathbf{X}_{ij}$ is everything necessary to generate a visualization $\mathcal{V} = (\mathbf{X}_{ij}^{(k)}, C)$. Hence, the tuple $(\mathbf{X}_{ij}^{(k)}, C)$ defines a unique visualization $\mathcal{V}$ that leverages the subset of attributes $\mathbf{X}_{ij}^{(k)}$ from dataset $\mathbf{X}_{ij}$ along with the visualization configuration $C \in \mathcal{C}$.

Definition 2 (Visualization Configuration). *Given a visualization $\mathcal{X}$ generated using a subset of attributes $\mathbf{X}_{ij}^{(k)}$ from dataset $\mathbf{X}_{ij}$, we define a function*

$$\Gamma : \mathcal{V} \rightarrow C \tag{10}$$

*where Γ maps every data-dependent design choice of the visualization to its corresponding type (*i.e.*, the attribute mapping to the x-axis of the visualization $\mathcal{V}$ is replaced with its general type such as quantitative, nominal, ordinal, temporal, etc). The resulting visualization configuration C is an abstraction of the visualization $\mathcal{V}$, in the sense that all the data attribute bindings have been abstracted and replaced with their general data attribute type. Hence, C is an abstraction of $\mathcal{V}$.*

Intuitively, it is by replacing the data-specific design choices with their general type or set of general properties that enables us to capture and learn from these visualization-configurations. An example of a visual-configuration is shown in Figure 1. These visual-configurations can be thought of as a type of Vega-Lite spec. with the precise data-attributes abstracted to make it data-independent.

Definition 3 (Space of Visualization Configurations). *Let $\mathcal{C}$ denote the space of all visualization configurations such that a visualization configuration $C_{ik} \in \mathcal{C}$ defines an abstraction of a visualization where for each visual design choice (x, y, marker-type, color, size, etc.) that maps to an attribute in some dataset $\mathbf{X}_{ij}$, we replace it with its type such as quantitative, nominal, ordinal, temporal or some other general property characterizing the attribute that can be selected. Therefore visualization configurations are essentially visualizations without any attributes (data), or visualization abstractions that are by definition data-independent.*

Property 2. *Every visualization configuration $C_{ik} \in \mathcal{C}$ is independent of any data matrix $\mathbf{X}$ (by Definition 3).*

The above implies that $C_{ik} \in \mathcal{C}$ can potentially arise from any arbitrary dataset and is therefore not tied to any specific dataset since visualization configurations are general abstractions where the data bindings have been replaced with their general type, *e.g.*, if x/y in some visualization mapped to an attribute in $\mathbf{X}$, then it is replaced by its type (*i.e.*, ordinal, quantitative, categorical, etc). Note that while $\mathcal{C}$ can technically be very large, it is reasonable in practice. For instance, the size of $\mathcal{C}$ depends on the set of design choices and the possible values of each one. However, most are very small such as the chart-type, x/y-attribute type and so on. For other design choices such as size or color, which can be continuous and large in general, we can instead restrict them to a much smaller discrete set of possibilities.

A visualization configuration *and* the attributes selected from a dataset is everything necessary to generate a visualization. The size of the space of visualization configurations is large since visualization configurations come from all possible combinations of design choices and their values.

Definition 4 (Space of Visualizations of $\mathbf{X}_{ij}$). *Given an arbitrary dataset matrix $\mathbf{X}_{ij}$, we define $\mathbb{V}^{\star}_{ij}$ as the space of all possible visualizations that can be generated from $\mathbf{X}_{ij}$. More formally, the space of visualizations $\mathbb{V}^{\star}_{ij}$ is defined with respect to a dataset $\mathbf{X}_{ij}$ and the space of visualization configurations $\boldsymbol{\mathcal{C}}$,*

$$\mathbb{X}_{ij} = \Sigma(\mathbf{X}_{ij}) = \{\mathbf{X}^{(1)}_{ij}, \ldots, \mathbf{X}^{(k)}_{ij}, \ldots\} \tag{11}$$

$$\xi : \mathbb{X}_{ij} \times \boldsymbol{\mathcal{C}} \rightarrow \boldsymbol{\mathcal{V}}^{\star}_{ij} \tag{12}$$

where $\mathbb{X}_{ij} = \{\mathbf{X}^{1}_{ij}, \ldots, \mathbf{X}^{(k)}_{ij}, \ldots\}$ is the set of all possible attribute combinations of $\mathbf{X}_{ij}$ (Def. 1). More succinctly, $\xi : \Sigma(\mathbf{X}_{ij}) \times \boldsymbol{\mathcal{C}} \rightarrow \boldsymbol{\mathcal{V}}^{\star}_{ij}$, and therefore $\xi(\Sigma(\mathbf{X}_{ij}), \boldsymbol{\mathcal{C}}) = \boldsymbol{\mathcal{V}}^{\star}_{ij}$. The space of all visualizations $\boldsymbol{\mathcal{V}}^{\star}_{ij}$ is determined entirely by the underlying dataset, and therefore remains the same for all n users. The difference in our personalized visualization recommendation problem is the relevance of each visualization in the space of all possible visualizations generated from an arbitrary dataset. Given a subset of attributes $\mathbf{X}^{(k)}_{ij} \in \mathbb{X}_{ij}$ from dataset $\mathbf{X}_{ij}$ and a visualization configuration $C \in \boldsymbol{\mathcal{C}}$, then $\xi(\mathbf{X}^{(k)}_{ij}, C) \in \boldsymbol{\mathcal{V}}^{\star}_{ij}$ is the corresponding visualization.

Importantly, fix $\boldsymbol{\mathcal{C}}$ and let $\mathbf{X} \neq \mathbf{Y} \implies \forall i, j\ \mathbf{x}_i \neq \mathbf{y}_j$, then $\xi(\Sigma(\mathbf{X}), \boldsymbol{\mathcal{C}}) \cap \xi(\Sigma(\mathbf{Y}), \boldsymbol{\mathcal{C}}) = \emptyset$. This implies the space of possible visualizations that can be generated is entirely dependent on the dataset (not the user). Hence, for any two datasets $\mathbf{X}$ and $\mathbf{Y}$ without any shared attributes between them, the set of visualizations that can be generated from $\mathbf{X}$ or $\mathbf{Y}$ is completely different,

$$\xi(\Sigma(\mathbf{X}), \boldsymbol{\mathcal{C}}) \cap \xi(\Sigma(\mathbf{Y}), \boldsymbol{\mathcal{C}}) = \emptyset$$

This has important consequences for the new problem of personalized visualization recommendation. Since it is unlikely that any two users care about the same underlying dataset, and even if they did, it is even far more unlikely that they have any relevant visualizations in common (just *w.r.t.* the exponential size of the visualization space for a single dataset with a reasonable amount of attributes). Therefore, it is not possible nor practical to leverage the relevant visualizations of a user directly. Instead, we need to decompose a visualization $\mathcal{V}$ into its more meaningful components such as: (i) the *characteristics* of the data attributes $\mathbf{X}^{(k)}_{ij}$ used in a visualization, and (ii) the visual design choices (chart-type/mark, color, size, and so on).

Definition 5 (Relevant Visualizations of User i and Dataset $\mathbf{X}_{ij}$). *Let $\boldsymbol{\mathcal{V}}_{ij} \in \mathbb{V}_i$ define the set of relevant (positive) visualizations for user i with respect to dataset $\mathbf{X}_{ij}$. Therefore, $\mathbb{V}_i = \bigcup_{\mathbf{X}_{ij} \in \boldsymbol{\mathcal{X}}_i} \boldsymbol{\mathcal{V}}_{ij}$ where $\mathbb{V}_i$ is the set of all relevant visualizations across all datasets $\boldsymbol{\mathcal{X}}_i$ of interest to user i.*

Definition 6 (Non-relevant Visualizations of User i and Dataset $\mathbf{X}_{ij}$). *For a user i, let $\mathbb{V}^{\star}_{ij}$ denote the space of all visualizations that arise from the jth dataset $\mathbf{X}_{ij}$ such that the relevant (positive) visualizations $\boldsymbol{\mathcal{V}}_{ij}$ satisfies $\boldsymbol{\mathcal{V}}_{ij} \subseteq \boldsymbol{\mathcal{V}}^{\star}_{ij}$, then the* space of non-relevant visualizations for user i on dataset $\mathbf{X}_{ij}$ *is $\boldsymbol{\mathcal{V}}^{-}_{ij} = \boldsymbol{\mathcal{V}}^{\star}_{ij} \setminus \boldsymbol{\mathcal{V}}_{ij}$, which follows from $\boldsymbol{\mathcal{V}}^{-}_{ij} \cup \boldsymbol{\mathcal{V}}_{ij} = \boldsymbol{\mathcal{V}}^{\star}_{ij}$.*

We denote Y_{ijk} as the *ground-truth label* of a visualization $\mathcal{V}_{ijk} \in \boldsymbol{\mathcal{V}}^{\star}_{ij}$ where $Y_{ijk} = 1$ if $\mathcal{V}_{ijk} \in \boldsymbol{\mathcal{V}}_{ij}$ and $Y_{ijk} = 0$ otherwise. Now we formulate the problem of training a user-level personalized

Table 1. Summary of notation. Matrices are bold upright roman letters; vectors are bold lowercase letters.

$\boldsymbol{\mathcal{D}}$	user log data $\boldsymbol{\mathcal{D}} = \{(\boldsymbol{\mathcal{X}}_i, \mathbb{V}_i)\}_{i=1}^n$ consisting of a set of datasets $\boldsymbol{\mathcal{X}}_i$ for every user $i \in [n]$ and the sets $\mathbb{V}_i$ of relevant visualizations for each of those datasets.
$\boldsymbol{\mathcal{X}}_i$	set of datasets (data matrices) of interest to user i where $\boldsymbol{\mathcal{X}}_i = \{\mathbf{X}_{i1}, \ldots, \mathbf{X}_{ij}, \ldots\}$
$\mathbf{X}_{ij}$	the jth dataset (data matrix) of interest to user i.
$\mathbb{V}_i$	sets of visualizations relevant to user i where $\mathbb{V}_i = \{\boldsymbol{\mathcal{V}}_{i1}, \ldots, \boldsymbol{\mathcal{V}}_{ij}, \ldots\}$
$\boldsymbol{\mathcal{V}}_{ij}$	set of visualizations relevant (generated) by user i for dataset j ($\mathbf{X}_{ij} \in \boldsymbol{\mathcal{X}}_i$) where $\boldsymbol{\mathcal{V}}_{ij} = \{\ldots, \mathcal{V}, \}$
$\mathcal{V} = (\mathbf{X}^{(k)}, C)$	a visualization $\mathcal{V}$ consisting of the subset of attributes $\mathbf{X}^{(k)}$ from some dataset $\mathbf{X}$ and the visual-configuration (design choices)
$\boldsymbol{\mathcal{C}}$	set of visual-configurations where $C \in \boldsymbol{\mathcal{C}}$ represents the visualization design choices for a single visualization $\mathcal{V}$ such as the chart-type, x-axis, y-axis, color, and so on.
$\mathbb{X}_{ij}$	space of attribute combinations/subsets $\mathbb{X}_{ij} = \{\mathbf{X}_{ij}^{(1)}, \ldots, \mathbf{X}_{ij}^{(k)}, \ldots\}$ of dataset $\mathbf{X}_{ij}$
n	number of users
m	number of attributes (columns, variables) across all datasets, $m = \sum_i m_i$ where m_i = number of attributes in the i-th dataset
v	number of relevant (user-generated) visualizations across all users and datasets
h	number of visualization configurations
k	dimensionality of the *shared attribute feature space*, *i.e.*, number of attribute features
d	shared latent embedding dimensionality
t	number of types of implicit/explicit user feedback, *i.e.*, attribute and visualization click, like, add-to-dashboard, among others
$\mathbf{x}$	a attribute (column) vector from an arbitrary user uploaded dataset
$\lvert\mathbf{x}\rvert$	cardinality of $\mathbf{x}$, *i.e.*, number of unique values in $\mathbf{x}$
$\mathsf{nnz}(\mathbf{x})$	number of nonzeros in a vector $\mathbf{x}$
$\mathsf{len}(\mathbf{x})$	length of a vector $\mathbf{x}$
$\mathbf{A}$	user by attribute preference matrix
$\mathbf{C}$	user by visualization configuration matrix
$\mathbf{D}$	attribute preference by visual-configuration matrix
$\mathbf{M}$	attribute by meta-feature matrix
$\mathbf{U}$	shared user embedding matrix
$\mathbf{V}$	shared attribute embedding matrix
$\mathbf{Z}$	shared visualization configuration embedding matrix
$\mathbf{Y}$	meta-feature embedding matrix for the attributes across all datasets

visualization recommendation model $\boldsymbol{\mathcal{M}}_i$ for user i from a large user-centric visualization training corpus $\boldsymbol{\mathcal{D}}$.

Definition 7 (Training *Personalized* Vis. Recommendation Model). *Given a set of training datasets and user-relevant visualizations $\boldsymbol{\mathcal{D}} = \{(\boldsymbol{\mathcal{X}}_i, \mathbb{V}_i)\}_{i=1}^n$, the goal is to learn a* personalized *visualization recommendation model $\boldsymbol{\mathcal{M}}_i$ for user i by solving the following general objective function,*

$$\underset{\boldsymbol{\mathcal{M}}_i}{\arg\min} \sum_{j=1}^{|\boldsymbol{\mathcal{X}}_i|} \sum_{(\mathbf{X}_{ij}^{(k)}, C_{ijk}) \in \boldsymbol{\mathcal{V}}_{ij} \cup \widehat{\boldsymbol{\mathcal{V}}}_{ij}^-} \mathbb{L}\Big(Y_{ijk} \,\Big|\, \Psi(\mathbf{X}_{ij}^{(k)}), f(C_{ijk}), \boldsymbol{\mathcal{M}}_i\Big), \quad i = 1, \ldots, n \tag{13}$$

where $\mathbb{L}$ is the loss function, $Y_{ijk} = \{0, 1\}$ is the ground-truth label of the kth visualization $\mathcal{V}_{ijk} = (\mathbf{X}_{ij}^{(k)}, C_{ijk}) \in \boldsymbol{\mathcal{V}}_{ij} \cup \widehat{\boldsymbol{\mathcal{V}}}_{ij}^-$ for dataset $\mathbf{X}_{ij} \in \boldsymbol{\mathcal{X}}_i$ of user i. Further, $\mathbf{X}_{ij}^{(k)} \subseteq \mathbf{X}_{ij}$ is the subset of attributes used in the visualization. In Eq. 13, Ψ and f are general functions over the subset of attributes $\mathbf{X}_{ij}^{(k)} \subseteq \mathbf{X}_{ij}$ and the visualization configuration C_{ijk} of the visualization $\mathcal{V}_{ijk} = (\mathbf{X}_{ij}^{(k)}, C_{ijk}) \in \boldsymbol{\mathcal{V}}_{ij}^- \cup \boldsymbol{\mathcal{V}}_{ij}$, respectively.

For learning individual models $\boldsymbol{\mathcal{M}}_i$ for every user $i \in [n]$, we can also leverage the visualization and data preferences from other users. The simplest and most straightforward situation is when there is another user $i' \in [n]$ with a set of relevant visualizations that use attributes from the same exact dataset, hence $|\boldsymbol{\mathcal{X}}_i \cap \boldsymbol{\mathcal{X}}_{i'}| > 0$. While the above strict assumption is convenient as it makes the problem far simpler, it is unrealistic in practice (and not very useful) to assume there exists a single dataset of interest to all users. Therefore, we designed the approach to be able to learn from visualizations preferred by other users on completely different datasets. To make this possible, we leverage the similarity between the attributes (used in the user-relevant visualizations) across completely different datasets. However, it is non-trivial since similarity functions require both attribute vectors to be the same size (dimension), but also the same data-type. In our case, both assumptions are unlikely to hold, making it infeasible to use directly. To overcome this issue, we introduce the notion of a meta-feature function that can be used to embed all attributes from any arbitrary dataset to a shared fixed dimensional space. More formally,

Definition 8 (Meta-Feature Function). *Given an attribute* $\mathbf{x}$ *of any dimensionality from any dataset* $\mathbf{X}$*, let* Ψ *denote the meta-feature function that maps a data-attribute* $\mathbf{x}$ *to a shared* K*-dimensional meta-feature space that captures the important data characteristics of* $\mathbf{x}$ *that are universal (shared) across any arbitrary dataset of interest to the users. More formally,*

$$\Psi : \mathbf{x} \to \mathbb{R}^K \tag{14}$$

where $\mathbf{x}$ *can be of an arbitrary attribute type (*e.g.*, real-valued, integral, nominal, ordinal, etc) and size. For convenience, we also define* $\Psi : \mathbf{X} \to \mathbb{R}^{K \times M}$ *where* $\mathbf{X}$ *is an arbitrary dataset with* M *attributes. Hence,* $\Psi(\mathbf{x}) \in \mathbb{R}^K$ *and* $\Psi(\mathbf{X}) \in \mathbb{R}^{K \times M}$*.*

In addition, we also leverage the similarity between the visual-configurations of the user-relevant visualizations, despite the visualizations arising from different datasets. More formally, given any two users $i, i' \in [n]$ along with one of their relevant visualizations, $\mathcal{V}_{ijk} = (\mathbf{X}_{ij}^{(k)}, C_{ijk}) \in \boldsymbol{\mathcal{V}}_{ij}$ and $\mathcal{V}_{i'j'k'} = (\mathbf{X}_{i'j'}^{(k')}, C_{i'j'k'}) \in \boldsymbol{\mathcal{V}}_{i'j'}$, then since we know that the datasets used in these visualizations are completely different, we instead can leverage this across-dataset training information if they use similar attributes, where across-dataset similarity is measured by first mapping each attribute used in the visualization to a shared K-dimensional meta-feature space, where we can then measure the similarity between each of the attributes used in the visualizations generated by different users. Hence, $s\langle \Psi(\mathbf{X}_{ij}^{(k)}), \Psi(\mathbf{X}_{i'j'}^{(k')}) \rangle > 1 - \epsilon$ where $\mathbf{X}_{i'j'} \notin \boldsymbol{\mathcal{X}}_i$ and $\mathbf{X}_{ij} \notin \boldsymbol{\mathcal{X}}_{i'}$. Intuitively, this implies that even though the visualizations are generated using different data, they visualize data that is similar with respect to its overall characteristics and patterns. By construction, visualizations $\mathcal{V}_{ijk}$ and $\mathcal{V}_{i'j'k'}$ from two different users $i, i' \in [n]$ and datasets $\mathbf{X}_{ij} \neq \mathbf{X}_{i'j'}$ may use the same visual-configuration (set of design choices), $C_{ijk} = C_{i'j'k'} \in \boldsymbol{\mathcal{C}}$, since we defined the notion of visual-configurations to be data-independent, and thus, even though two visualizations may visualize data attributes from completely different datasets, they can still share the same visual-configuration (design choices). Therefore, as we will see later, we are able to learn from other users with visualizations that use attributes from completely different datasets.

2.4 Personalized Visualization Scoring and Recommendation

After learning the personalized visualization recommendation model $\boldsymbol{\mathcal{M}}_i$ for an individual user $i \in [n]$ (Eq. 13), we can then use $\boldsymbol{\mathcal{M}}_i$ to score and recommend the top most relevant visualizations for user i from any arbitrary dataset $\mathbf{X}$. There are three possible cases that are naturally supported by the learned model $\boldsymbol{\mathcal{M}}_i$ for recommending visualizations specifically of interest to user i based on their past interactions (visualizations the user viewed/clicked or more generally interacted with):

(1) The dataset X used for recommending personalized visualizations to user i via $\boldsymbol{\mathcal{M}}_i$ can be a new previously unseen dataset of interest $\mathbf{X} \notin \{\boldsymbol{\mathcal{X}}_1, \boldsymbol{\mathcal{X}}_2, \ldots, \boldsymbol{\mathcal{X}}_n\}$
(2) The dataset X is not a previous dataset of interest to user i, but has been used previously by one or more other users $\mathbf{X} \in \{\boldsymbol{\mathcal{X}}_1, \ldots, \boldsymbol{\mathcal{X}}_n\} \setminus \boldsymbol{\mathcal{X}}_i$
(3) The dataset $\mathbf{X} \in \boldsymbol{\mathcal{X}}_i$ is a previous dataset of interest to user i

A fundamental property of the personalized visualization recommendation problem is that the user visualization scores for an arbitrary visualization $\mathcal{V}$ (that visualizes data from an arbitrary dataset X) are different depending on the individual user and their historical preferences and interests. More formally, given users $i, i' \in [n]$ and a visualization $\mathcal{V}$ from a new unseen dataset $\mathbf{X}_{\text{test}}$, we obtain personalized visualization scores for user i and i' as $\boldsymbol{\mathcal{M}}_i(\mathcal{V})$ and $\boldsymbol{\mathcal{M}}_{i'}(\mathcal{V})$, respectively. While existing rule-based [Moritz et al. 2018; Wongsuphasawat et al. 2015, 2017] or ML-based systems [Qian et al. 2020] score the visualization $\mathcal{V}$ the same, no matter the actual user of the system (hence, are agnostic to the actual user and their interests, past interactions, and intent), our work instead focuses on learning individual personalized visualization recommendation models for every user $i \in [n]$ such that the personalized score $\boldsymbol{\mathcal{M}}_i(\mathcal{V})$ of visualization $\mathcal{V}$ for user i is almost surely different from the score $\boldsymbol{\mathcal{M}}_{i'}(\mathcal{V})$ given by the personalized model of another user i', $\boldsymbol{\mathcal{M}}_i(\mathcal{V}) \neq \boldsymbol{\mathcal{M}}_{i'}(\mathcal{V})$. We can state this more generally for all pairs of users $i, i' \in [n]$ with respect to a single arbitrary visualization $\mathcal{V}$,

$$\boldsymbol{\mathcal{M}}_i(\mathcal{V}) \neq \boldsymbol{\mathcal{M}}_{i'}(\mathcal{V}), \quad \forall i, i' = 1, \ldots, n \quad \text{s.t. } i < i' \tag{15}$$

Hence, given an arbitrary visualization $\mathcal{V}$, the personalized scores $\boldsymbol{\mathcal{M}}_i(\mathcal{V})$ and $\boldsymbol{\mathcal{M}}'_i(\mathcal{V})$ for any two distinct users i and i' are not equal with high probability. This is due to the fact that the personalized visualization recommendation models $\boldsymbol{\mathcal{M}}_1, \boldsymbol{\mathcal{M}}_2, \ldots, \boldsymbol{\mathcal{M}}_n$ capture each of the n users individual data preferences, design/visual preferences, and overall visualization preferences. Data-efficiency is a significant problem for any naive model and formulation due to the sparsely observed visualizations per user and dataset dependence of the visualizations. Therefore, we address these data-efficiency issues by mapping the data-attribute and visual feedback of users into a shared space where the model can leverage feedback from other users. For instance, we map the data attributes from every dataset of a user into a shared K-dimensional space (via meta-features) that enables us to leverage data-attribute feedback from other users, despite that the data-attributes are from different datasets. These aspects are discussed further in Section 3.

Definition 9 (Personalized Visualization Scoring). *Given the personalized visualization recommendation model $\boldsymbol{\mathcal{M}}_i$ for user i and a dataset $\mathbf{X}_{\text{test}}$ of interest to user i, we can obtain the personalized scores for user i of every possible visualization that can be generated as,*

$$\boldsymbol{\mathcal{M}}_i : \boldsymbol{\mathcal{X}}_{\text{test}} \times \boldsymbol{\mathcal{C}} \rightarrow \mathbb{R} \tag{16}$$

where $\boldsymbol{\mathcal{X}}_{\text{test}} = \{\ldots, \mathbf{X}^{(k)}_{\text{test}}, \ldots\}$ is the space of attribute subsets from $\mathbf{X}_{\text{test}}$ and $\boldsymbol{\mathcal{C}}$ is the space of visualization configurations. Hence, given an arbitrary visualization $\mathcal{V}$, the learned model $\boldsymbol{\mathcal{M}}_i$ outputs a personalized score for user i describing the effectiveness or importance of the visualization with respect to that individual user.

Definition 10 (Personalized Visualization Ranking). *Given the set of generated visualizations $\mathbb{V}_{\text{test}} = \{\boldsymbol{\mathcal{V}}_1, \boldsymbol{\mathcal{V}}_2, \ldots, \boldsymbol{\mathcal{V}}_Q\}$ where $Q = |\mathbb{V}_{\text{test}}|$, we derive a personalized ranking of the visualizations $\mathbb{V}_{\text{test}}$ from $\mathbf{X}_{\text{test}}$ for user i as follows:*

$$\rho_i\big(\{\mathcal{V}_1, \mathcal{V}_2, \ldots, \mathcal{V}_Q\}\big) = \operatorname*{arg\,sort}_{\mathcal{V}_t \in \mathbb{V}_{\text{test}}} \boldsymbol{\mathcal{M}}_i(\mathcal{V}_t) \tag{17}$$

where for any two visualizations $\mathcal{V}_t$ and $\mathcal{V}_{t'}$ in the personalized ranking $\rho_i(\{\mathcal{V}_1, \mathcal{V}_2, \ldots, \mathcal{V}_{|Q|}\})$ of visualizations for the individual user i (from dataset $\mathbf{X}_{\text{test}}$) such that $t < t'$, then $\boldsymbol{\mathcal{M}}_i(\mathcal{V}_t) \geq \boldsymbol{\mathcal{M}}_i(\mathcal{V}_{t'})$ holds by definition.

Informally, given a *new dataset* $\mathbf{X}_{\text{test}}$ to recommend visualizations for via the trained model $\boldsymbol{\mathcal{M}}_i$ (Eq. 13), then $\boldsymbol{\mathcal{M}}_i(\xi(\Sigma(\mathbf{X}_{\text{test}}), \boldsymbol{C}))$ where $\boldsymbol{C}$ is the space of relevant visualization configurations. Notice that $\boldsymbol{\mathcal{M}}_i(\mathbb{V}_{\text{test}}) = \boldsymbol{\mathcal{M}}_i(\xi(\Sigma(\mathbf{X}_{\text{test}}), \boldsymbol{C}))$. For tractability, we replace the set of possible visualization configurations $\boldsymbol{C}$ with the set of relevant configurations $\boldsymbol{C}_r = \mathcal{R}(\boldsymbol{C})$ where $\mathcal{R}$ is a function consisting of visual rules that enables us to discard configurations that are invalid with respect to the manually defined rules. The list of rules from Voyager and other rule-based systems can read from a file similar to stopwords in information retrieval. Hence, $\boldsymbol{C}_r \subseteq \boldsymbol{C}$.

Furthermore, given a new dataset of interest, the space of visualizations to search over is completely different from the space of visualizations that arises from any other (non-identical) dataset. More formally, let $\mathbb{V}_i^\star$ and $\mathbb{V}_j^\star$ denote the space of all possible visualizations that arise from $\mathbf{X}_i$ and $\mathbf{X}_j$ held-out datasets, then $\forall r, s$, $\boldsymbol{\mathcal{V}}_r \in \mathbb{V}_i^\star \neq \boldsymbol{\mathcal{V}}_s \in \mathbb{V}_j^\star$ holds. Further, this obviously holds $\forall i, j \in [T]$ as well. Clearly, the above holds, since a visualization consists of a subset of attributes (data) and design choices. The above demonstrates the difficulty of the visualization recommendation learning problem, in the sense that, the model must recommend relevant visualizations from a space of visualizations never seen by the learning algorithm. Moreover, we can even show a weaker property regarding the cardinality of the space of visualizations that arise from different held-out datasets,

Claim 2.1. *Let $\mathbb{V}_i^\star$ and $\mathbb{V}_j^\star$ denote the space of all possible visualizations that arise from $\mathbf{X}_i$ and $\mathbf{X}_j$ held-out datasets, then with high probability $|\mathbb{V}_i^\star| \neq |\mathbb{V}_j^\star|$ almost surely holds $\forall i, j \in [T]$.*

3 PERSONALIZED VISUALIZATION RECOMMENDATION FRAMEWORK

In this section, we present the framework for solving the personalized visualization recommendation problem from Section 2. In Section 3.1, we first describe the meta-feature learning approach for mapping user datasets to a shared universal meta-feature space where relationships between the corpus of tens of thousands of datasets can be automatically inferred and used for learning individual personalized models for each user. Then Section 3.2 introduces a graph model that captures the data preferences of users while Section 3.3 proposes graph models that naturally encode the visual preferences of users. The personalized visualization recommendation models learned from the proposed graph representations are described in Section 3.4, while the visualization scoring and recommendation techniques are presented in Section 3.5.

3.1 Representing Datasets in a Universal Shared Meta-Feature Space

To learn from user datasets of different sizes, types, and characteristics, we first embed the attributes (columns) of each dataset $\mathbf{X} \in \mathcal{X}_1 \cup \mathcal{X}_2 \cup \cdots \cup \mathcal{X}_n$ (from any user) in a shared K-dimensional meta-feature space. This also enables the personalized visualization recommendation model to learn from users with similar data preferences. Recall that each user $i \in [n]$ is associated with a set of datasets $\mathcal{X}_i = \{\mathbf{X}_{i1}, \mathbf{X}_{i2}, \ldots\}$.

Claim 3.1. *Let $\boldsymbol{\mathcal{X}} = \bigcup_{i=1}^n \mathcal{X}_i$ denote the set of all datasets. Then*

$$\sum_{i=1}^{n} |\mathcal{X}_i| \geq |\boldsymbol{\mathcal{X}}| \tag{18}$$

Table 2. Meta-feature learning framework overview

Framework Components	Examples
1. Data representations $\mathcal{G}$	$\mathbf{x}$, $\mathbf{p}$, $g(\mathbf{x})$, $\ell_b(\mathbf{x})$ log-binning, ...
2. Partitioning functions Π	Clustering, binning, quartiles, ...
3. Meta-feature functions ψ	Statistical, information theoretic, ...
4. Meta-embedding of meta-features	$\underset{\mathbf{H},\Sigma,\mathbf{Q}}{\arg\min}\ \mathbb{D}_{\mathcal{L}}(\mathbf{M} \Vert \mathbf{H}\Sigma\mathbf{Q}^{\top})$, then $\widehat{\mathbf{q}} = \Sigma^{-1}\mathbf{H}^{\top}\widehat{\mathbf{m}}$

Hence, if $\sum_{i=1}^{n} |\mathcal{X}_i| = |\boldsymbol{\mathcal{X}}|$, then this implies that all users have completely different datasets (there does not exist any two users $i, j \in [n]$ that have a dataset in common). Otherwise, if there exists two users that have at least one dataset in common with one another, then $\sum_{i=1}^{n} |\mathcal{X}_i| > |\boldsymbol{\mathcal{X}}|$.

In our personalized visualization recommendation problem (Sec. 2), it is possible (and in many cases likely) that users are interested in completely different datasets. In the worst case, every user has a completely disjoint set of datasets, and thus, the implicit and/or explicit user feedback regarding the attributes of interest to the users is also completely disjoint. In such a case, the question then becomes how can we leverage the feedback from users like this, to better recommend attributes from different datasets that may be of interest to a new and/or previous user? To do this, we need a general method that can derive a fixed-size embedding $\Psi(\mathbf{x}) \in \mathbb{R}^K$ of an attribute $\mathbf{x}$ from any arbitrary dataset $\mathbf{X}$ such that the K-dimensional embedding $\Psi(\mathbf{x})$ captures the important data characteristics and statistical properties of $\mathbf{x}$, independent of the dataset and size of $\mathbf{x}$. One possibility of Ψ is the meta-feature functions shown in Table 3. Afterwards, given two attributes $\mathbf{x}$ and $\mathbf{y}$ from different datasets (*i.e.*, $\mathbf{X}$ and $\mathbf{Y}$) and users, we can derive the similarity between $\mathbf{x}$ and $\mathbf{y}$. Suppose there is implicit/explicit user feedback regarding an attribute $\mathbf{x}$, then given another arbitrary user i interested in a new dataset $\mathbf{Y}$ (without any feedback on the attributes in $\mathbf{Y}$), then we can derive the similarity between $\mathbf{x}$ and $\mathbf{y}$, and if $\mathbf{y}$ is similar to an attribute $\mathbf{x}$ that was preferred by some user(s), then we can assign the attribute $\mathbf{y}$ a higher probability (weight, score), despite that it doesn't yet have any user feedback. Therefore, as discussed above, it is clear that this idea of transferring user feedback about attributes across different datasets is extremely powerful and fundamentally important for personalized visualization recommendation (especially when there is only limited sparse feedback available). Moreover, the proposed idea above is also important when there is no feedback about an attribute in some dataset, or a completely new dataset of interest by a user. This enables us to learn better personalized visualization recommendation models for individual users while requiring significantly less feedback.

Property 3. *Two attributes* $\mathbf{x}$ *and* $\mathbf{y}$ *are similar iff*

$$s\langle \Psi(\mathbf{x}), \Psi(\mathbf{y}) \rangle > 1 - \epsilon. \tag{19}$$

where $s\langle \cdot, \cdot \rangle$ is the similarity function. Notice that since almost surely $|\mathbf{x}| \neq |\mathbf{y}|$ (different sizes), then the similarity of $\mathbf{x}$ and $\mathbf{y}$ cannot be computed directly. Therefore, we embed $\mathbf{x}$ and $\mathbf{y}$ into the same K-dimensional meta-feature space where there similarity can be computed directly as $s\langle \Psi(\mathbf{x}), \Psi(\mathbf{y}) \rangle$.

Attributes from different datasets are naturally of different sizes, types, and even from different domains. Therefore as shown above, there is no way to compute similarity between them directly. Instead, we propose to map each attribute from any arbitrary dataset into a shared K-dimensional space using meta-feature functions. After every attribute is mapped into this K-dimensional meta-feature space, we can then compare their similarity directly.

In this work, we propose a meta-feature learning framework with four main components as shown in Table 2. Many of the framework components use the meta-feature functions denoted as

ψ. A meta-feature function is a function that maps an arbitrary vector to a value that captures a specific characteristic of the vector of values. In this work, we leverage a large class of meta-feature functions formally defined in Table 3. The specific meta-feature functions shown in Table 3 were chosen since they capture a wide range of fundamental data characteristics that are important for visualization and related tasks. In particular, they represent statistical properties of the data relating to the distribution, sequential properties, informativeness, noisiness, dispersion, imputation effects, and so on. These properties serve as the fundamental basis that more complex meta-features are learned via the other framework components in Table 2. However, the framework is flexible and can leverage any arbitrary collection of meta-feature functions. Notably, mapping every attribute from any dataset into a low-dimensional meta-feature space enables the model to capture and learn from the similarity between user preferred attributes in completely different datasets.

Let $\mathbf{x}$ denote an attribute (column vector) from any arbitrary user dataset. Then we may apply the collection of meta-features ψ from Table 3 directly to $\mathbf{x}$ to obtain a low-dimensional representation of $\mathbf{x}$ as $\psi(\mathbf{x})$. In addition, we can also apply the meta-feature functions ψ to various representations and transformation of $\mathbf{x}$. For instance, we can first derive the probability distribution $p(\mathbf{x})$ of $\mathbf{x}$ such that $p(\mathbf{x})^\top \mathbf{e} = 1$, and then use the meta-feature functions ψ over $p(\mathbf{x})$ to characterize this representation of $\mathbf{x}$. We can also use the meta-feature functions to characterize other important representations and transformations of the attribute vector $\mathbf{x}$ such as different scale-invariant and dimensionless representations of the data using different normalization functions $g_h(\cdot)$ over the attribute (column) vector $\mathbf{x}$, and from each of these representations, we can apply the above meta-feature functions, *e.g.*, $g_h(\mathbf{x}) = \frac{\mathbf{x}-\min(\mathbf{x})}{\max(\mathbf{x})-\min(\mathbf{x})}$, then $\psi(g_h(\mathbf{x}))$. More generally, let $\mathcal{G} = \{g_1, g_2, \ldots, g_\ell\}$ denote a set of data representation and transformation functions that can be applied over an attribute vector $\mathbf{x}$ from any arbitrary user dataset. We first compute the meta-feature functions ψ (*e.g.*, from Table 3) over the ℓ different representations of the attribute vector $\mathbf{x}$ given by the functions $\mathcal{G} = \{g_1, g_2, \ldots, g_\ell\}$ as follows:

$$\psi(g_1(\mathbf{x})), \psi(g_2(\mathbf{x})), \ldots, \psi(g_\ell(\mathbf{x})) \tag{20}$$

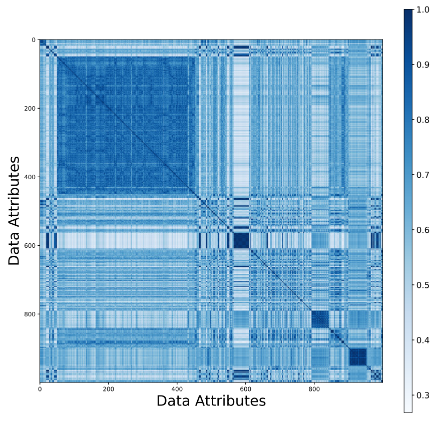
Note that if $g \in \mathcal{G}$ is the identity function, then $\psi(g(\mathbf{x})) = \psi(\mathbf{x})$. In all cases, the meta-feature function ψ maps a vector of arbitrary size to a fixed size lower-dimensional vector.

For each of the different representation/transformation functions $\mathcal{G} = \{g_1, \ldots, g_\ell\}$ of the attribute vector $\mathbf{x}$, we use a partitioning function Π to group the different values into k different subsets (*i.e.*, partitions, clusters, bins). Then we apply the meta-feature functions ψ to each of the k different groups as follows:

$$\underbrace{\psi(\Pi_1(g_1(\mathbf{x}))), \ldots, \psi(\Pi_k(g_1(\mathbf{x})))}_{g_1(\mathbf{x})}, \ldots, \underbrace{\psi(\Pi_1(g_\ell(\mathbf{x}))), \ldots, \psi(\Pi_k(g_\ell(\mathbf{x})))}_{g_\ell(\mathbf{x})} \tag{21}$$

where Π_k denotes the kth partition of values from the partitioning function Π. Note that to ensure every attribute is mapped to the same K-dimensional meta-feature space, we only need to fix the number of partitions k. In Eq. 21, we show only a single partitioning function Π, however, multiple partitioning functions are used in this work and each is applied in a similar fashion as Eq. 21. All the meta-features derived from Eq. 20 and Eq. 21 are then concatenated into a single vector of meta-features describing the characteristics of the attribute $\mathbf{x}$. More formally, the meta-feature function $\Psi : \mathbf{x} \to \mathbb{R}^K$ that combines the different components from the framework (in Table 2) is defined as

$$\begin{aligned}\Psi(\mathbf{x}) = \big[&\psi(g_1(\mathbf{x})) \cdots \psi(g_\ell(\mathbf{x})) \cdots \psi(\Pi_1(g_1(\mathbf{x}))) \cdots \psi(\Pi_k(g_1(\mathbf{x}))) \\ &\cdots \psi(\Pi_1(g_\ell(\mathbf{x}))) \cdots \psi(\Pi_k(g_\ell(\mathbf{x})))\big]\end{aligned} \tag{22}$$



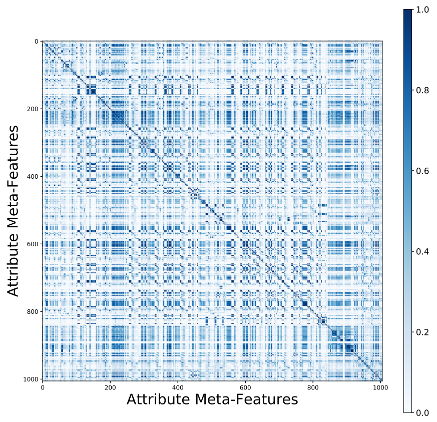


Fig. 3. Similarity of attributes across different datasets using the attribute embeddings in the universal meta-feature space. (a) uses the first 1000 attributes across different datasets and takes the cosine similarity between each pair of attributes with respect to their fixed k-dimensional meta-feature vectors. (b) shows the cosine similarity between the attribute meta-features used to characterize the different attributes. See text for discussion.

The resulting $\Psi(\mathbf{x})$ is a K-dimensional meta-feature vector for attribute $\mathbf{x}$. Our approach is agnostic to the precise meta-feature functions used, and is flexible for use with any alternative set of meta-feature functions (Table 3).

Let $\boldsymbol{\mathcal{X}} = \cup_{i=1}^{n} \mathcal{X}_i$ denote the set of dataset matrices across all n users. Given an arbitrary dataset matrix $\mathbf{X} \in \boldsymbol{\mathcal{X}}$ (which can be shared among multiple users), let $\Psi(\mathbf{X}) \in \mathbb{R}^{K \times |\mathbf{X}|}$ be the resulting meta-feature matrix obtained by applying Ψ independently to each of the $|\mathbf{X}|$ attributes (columns) of $\mathbf{X}$. Then, we can derive the overall meta-feature matrix $\mathbf{M}$ as

$$\mathbf{M} = \bigoplus_{\mathbf{X} \in \boldsymbol{\mathcal{X}}} \Psi(\mathbf{X}) \tag{23}$$

where $\bigoplus$ is the concatenation operator, *i.e.*, $\Psi(\mathbf{x}) \oplus \Psi(\mathbf{y}) = [\Psi(\mathbf{x})\ \Psi(\mathbf{y})] \in \mathbb{R}^{K \times 2}$. Note that Eq. 23 is not equivalent to $\bigoplus_{i=1}^{n} \bigoplus_{\mathbf{X} \in \mathcal{X}_i} \Psi(\mathbf{X})$ since any two users $i, j \in [n]$ can share one or more datasets. With slight abuse of notation, let $d = |\boldsymbol{\mathcal{X}}|$ and $\boldsymbol{\mathcal{X}} = \{\mathbf{X}_1, \ldots, \mathbf{X}_d\}$, then $\mathbf{M} = \Psi(\{\mathbf{X}_1, , \ldots, \mathbf{X}_d\})$ where $\mathbf{M}$ is a $K \times (|\mathbf{X}_1| + \cdots + |\mathbf{X}_d|)$ matrix.

In Figure 3, we investigate the similarity of attributes across different datasets in the personalized visualization corpus (Section 5), and observe two important findings. First, Figure 3(a) indicates that attributes across different datasets may be similar to one another and the latent relationships between the attributes can benefit learning personalized visualization recommendation models, especially for users with very few or even no visualization feedback. Second, the meta-features used to characterize attributes from any arbitrary dataset are diverse and fundamentally different from one another as shown in Figure 3(b). This finding is important and validates the proposed meta-feature learning framework since the meta-features must be able to capture the fundamental patterns and characteristics for a dataset from any arbitrary domain.

3.1.1 *Meta-Embedding of Meta-Features.*

We can derive an embedding using the current meta-feature matrix $\mathbf{M}$. Note this meta-feature matrix may contain all meta-features across all previous datasets or simply the meta-features of a single dataset. However, the more datasets, the better

Table 3. Summary of attribute meta-feature functions. Let $\mathbf{x}$ denote an arbitrary attribute (variable, column, field) vector and $\pi(\mathbf{x})$ is the sorted vector.

Function Name	**Equation**
Num. instances	$\lvert\mathbf{x}\rvert$
Num. missing values	s
Frac. of missing values	$\lvert\mathbf{x}\rvert - s / \lvert\mathbf{x}\rvert$
Num. nonzeros	$\mathsf{nnz}(\mathbf{x})$
Num. unique values	$\mathsf{card}(\mathbf{x})$
Density	$\mathsf{nnz}(\mathbf{x}) / \lvert\mathbf{x}\rvert$
Q_1, Q_3	median of the $\lvert\mathbf{x}\rvert / 2$ smallest (largest) values
IQR	$Q_3 - Q_1$
Outlier LB $\alpha \in \{1.5, 3\}$	$\sum_i \mathbb{I}(x_i < Q_1 - \alpha IQR)$
Outlier UB $\alpha \in \{1.5, 3\}$	$\sum_i \mathbb{I}(x_i > Q_3 + \alpha IQR)$
Total outliers $\alpha \in \{1.5, 3\}$	$\sum_i \mathbb{I}(x_i < Q_1 - \alpha IQR) + \sum_i \mathbb{I}(x_i > Q_3 + \alpha IQR)$
(αstd) outliers $\alpha \in \{2, 3\}$	$\mu_\mathbf{x} \pm \alpha\sigma_\mathbf{x}$
Spearman (ρ, p-val)	$\mathsf{spearman}(\mathbf{x}, \pi(\mathbf{x}))$
Kendall (τ, p-val)	$\mathsf{kendall}(\mathbf{x}, \pi(\mathbf{x}))$
Pearson (r, p-val)	$\mathsf{pearson}(\mathbf{x}, \pi(\mathbf{x}))$
Min, max	$\min(\mathbf{x}), \max(\mathbf{x})$
Range	$\max(\mathbf{x}) - \min(\mathbf{x})$
Median	$\mathsf{med}(\mathbf{x})$
Geometric Mean	$\lvert\mathbf{x}\rvert^{-1} \prod_i x_i$
Harmonic Mean	$\lvert\mathbf{x}\rvert / \sum_i \frac{1}{x_i}$
Mean, Stdev, Variance	$\mu_\mathbf{x}, \sigma_\mathbf{x}, \sigma_\mathbf{x}^2$
Skewness	$\mathbb{E}(\mathbf{x}-\mu_\mathbf{x})^3 / \sigma_\mathbf{x}^3$
Kurtosis	$\mathbb{E}(\mathbf{x}-\mu_\mathbf{x})^4 / \sigma_\mathbf{x}^4$
HyperSkewness	$\mathbb{E}(\mathbf{x}-\mu_\mathbf{x})^5 / \sigma_\mathbf{x}^5$
Moments [6-10]	–
k-statistic [3-4]	–
Quartile Dispersion Coeff.	$\frac{Q_3-Q_1}{Q_3+Q_1}$
Median Absolute Deviation	$\mathsf{med}(\lvert\mathbf{x} - \mathsf{med}(\mathbf{x})\rvert)$
Avg. Absolute Deviation	$\frac{1}{\lvert\mathbf{x}\rvert}\mathbf{e}^T \lvert\mathbf{x} - \mu_\mathbf{x}\rvert$
Coeff. of Variation	$\sigma_\mathbf{x} / \mu_\mathbf{x}$
Efficiency ratio	$\sigma_\mathbf{x}^2 / \mu_\mathbf{x}^2$
Variance-to-mean ratio	$\sigma_\mathbf{x}^2 / \mu_\mathbf{x}$
Signal-to-noise ratio (SNR)	$\mu_\mathbf{x}^2 / \sigma_\mathbf{x}^2$
Entropy	$H(\mathbf{x}) = -\sum_i \; x_i \log x_i$
Norm. entropy	$H(\mathbf{x}) / \log_2 \lvert\mathbf{x}\rvert$
Gini coefficient	–
Quartile max gap	$\max(Q_{i+1} - Q_i)$
Centroid max gap	$\max_{ij} \lvert c_i - c_j\rvert$
Histogram prob. dist.	$\mathbf{p}_h = \frac{\mathbf{h}}{\mathbf{h}^T\mathbf{e}}$ (with fixed # of bins)
Landmarker(4-Means)	(i) sum of squared dist., (ii) mean silhouette coeff., (iii) num. of iterations

the meta-embedding of the meta-feature will reveal the important latent structures between the meta-features. We learn the latent structure in the meta-feature matrix $\mathbf{M}$ by solving[1]

$$\arg\min_{\mathbf{H},\boldsymbol{\Sigma},\mathbf{Q}} \; \mathbb{D}_{\mathcal{L}}\big(\mathbf{M} \| \mathbf{H}\boldsymbol{\Sigma}\mathbf{Q}^{\top}\big) \tag{24}$$

Given meta-features for a new attribute $\widehat{\mathbf{m}}$ in another arbitrary unseen dataset, we use the latent low-rank meta-embedding matrices to map the meta-feature vector $\widehat{\mathbf{m}} \in \mathbb{R}^k$ into the low-rank meta-embedding space as

$$\widehat{\mathbf{q}} = \boldsymbol{\Sigma}^{-1}\mathbf{H}^{\top}\widehat{\mathbf{m}} \tag{25}$$

Hence, the meta-feature vector $\widehat{\mathbf{m}}$ of a new previously unseen attribute is mapped into the same meta-embedding space $\widehat{\mathbf{q}}$. The resulting meta-embedding of the meta-features of the new attribute can then be concatenated onto the meta-feature vector. This has several important advantages. First, using the proposed meta-feature learning framework shown in Table 2 results in hundreds or thousands of meta-features for a single attribute. Many of these meta-features may not be important for a specific attribute, while a few meta-features may be crucial in describing the attribute and its data characteristics. Therefore, the meta-embedding of the meta-features can be viewed as a noise reduction step that essentially removes redundant or noisy signals from the data while preserving the most important signals that describe the fundamental direction and characteristics of the data. Second, the meta-embedding of the meta-features reveals the latent structure and relationships in the meta-features. This step can also be viewed as a type of landmark feature since we solve a learning problem to find a low-rank approximation of $\mathbf{M}$ such that $\mathbf{M} \approx \mathbf{H}\boldsymbol{\Sigma}\mathbf{Q}^{\top}$.

However, we include it as a different component of the meta-feature learning framework in Table 2 since instead of concatenating the meta-embedding of the meta-features for a attribute, we can also use it directly by replacing it with the meta-feature vector. This is especially important when there is a large number of datasets (*e.g.*, more than 100K datasets with millions of attributes in total) for learning. For instance, if there are 2.3M attributes (see 100K dataset in Table 4), and each attribute is encoded with a dense $K = 1006$ dimensional meta-feature vector, then $\mathbf{M}$ has $1006 \times 2,300,000$ values that need to be stored, which use 18.5GB space (assuming 8 bytes per value). However, if we use the meta-embedding of the meta-features with $K = 10$, then $\mathbf{M}$ takes about 200MB (0.18GB) of space.

3.2 Learning from User-level Data Preferences Across Different Datasets

Given users i and j that provide feedback on the attributes of interest from two completely different datasets, how can we leverage the user feedback (data preferences) despite it being across different datasets without any shared attributes? To address this important problem, we propose a novel representation and model that naturally enables the transfer of user-level data preferences across different datasets to improve predictive performance, recommendations, and reduce data sparsity. The across dataset transfer learning of user-level data preferences becomes possible due to the proposed representation and model for personalized visualization recommendation.

We now introduce the novel user-level data preference graph model for personalized visualization recommendation that naturally enables across-dataset and across-user transfer learning of preferences. This model encodes users, their interactions with attributes (columns/variables from any arbitrary dataset) and the meta-features of the attributes. This new representation enables us to learn from user-level data preferences across different datasets and users, and therefore very important for personalized visualization recommendation systems. In particular, we first derive the

[1]Assume *w.l.o.g.* that columns of $\mathbf{M}$ and the meta-features of a new attribute $\widehat{\mathbf{m}}$ are normalized to length 1.

following user-by-attribute preference matrix **A** as follows:

$$\mathbf{A} = \left[\mathbf{A}\right]_{ij} = \text{\# of times user } i \text{ clicked (a visualization with) attribute } j \tag{26}$$

In terms of the implicit or explicit user "action" encoded by A_{ij}, it could be an implicit user action such as when a user clicks or hovers-over a specific attribute j or when a user clicks or hovers-over a visualization that uses attribute j in it. Similarly, A_{ij} can encode an explicit user action/feedback such as the attribute j that a user explicitly liked (independent of a visualization), or more generally, the attribute j used in a *visualization* that a user explicitly liked, or added-to-their dashboard, and so on. In other words, there are two different types of explicit and implicit user feedback about attributes, notably, user feedback regarding a visualization that used an attribute j (whether the user action is a click, hover, added-to-dashboard, etc), or more directly, whether a user liked or clicked on an attribute in the dataset directly via some UI.

Given **A** defined in Eq. 26, we are able to learn from two or more users that have at least one attribute preference in common. More precisely, $\mathbf{A}_{i,:}^{\top}\mathbf{A}_{j,:} > 0$ for two arbitrary users i and j, which implies two users i and j share a dataset of interest, *and* have preferred at least one of the same attributes in that dataset. Unfortunately, finding two users that satisfy the above constraint is often unlikely. Therefore, we need to add another representation to **A** that creates meaningful connections between attributes in different datasets based on their similarity. In particular, we do this by leveraging the meta-feature matrix **M** from Section 3.1 derives by mapping every attribute in a user-specific dataset to a k-dimensional meta-feature vector. This defines a universal meta-feature space that is shared among the attributes in any arbitrary dataset, and therefore allowing the learning of connections between users and their preferred attributes in completely different datasets. This new component is very important, since without it, we have no way to learn from other users (and across different datasets), since each user has its own datasets, and thus has their own set of visualizations (where each visualization consists of a set of design choices and data choices) that are not shared by any other users.

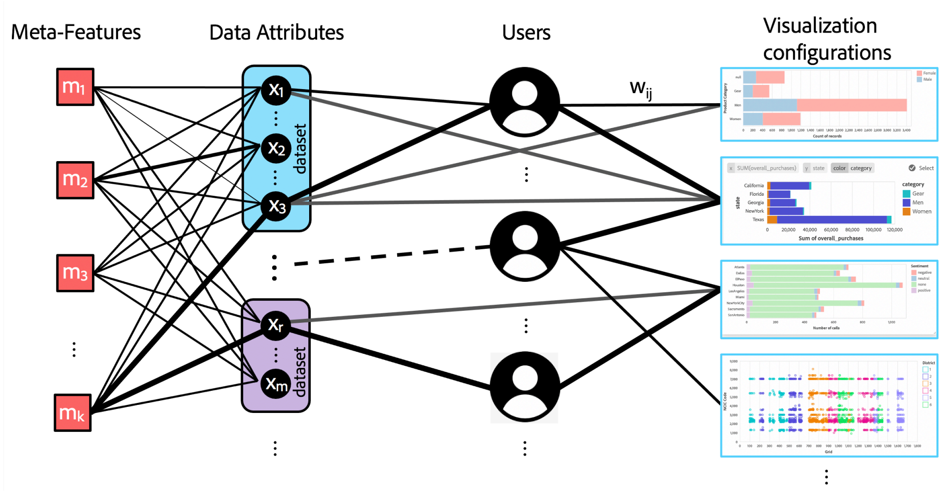


Fig. 4. Overview of the proposed graph model for personalized visualization recommendation. Links between meta-features and data attributes represent the meta-feature matrix **M** from Section 3.1 whereas links between users and their preferred data attributes represent **A** (Section 3.2). Both of these capture the user-level data preferences across different datasets. Links between users and their visualization configurations represent **A** and capture the user-level visual preferences. Finally, links between visualization configurations and data attributes represent **D**. Note that all links are weighted; and the data attribute/column nodes of a specific dataset are grouped together.

3.3 Learning from *User-level Visual Preferences* Across Different Datasets

Visualizations naturally consist of data and visual design choices. The dependence of data in a visualization means that visualizations generated for one dataset will be completely different from the visualizations generated by any other dataset. This is problematic if we want to develop a personalized visualization recommendation system that can learn from the *visual preferences* of users despite that the users may not have preferred any visualizations from the same datasets. This is important since visualizations are fundamentally tied to the dataset, and each user may have their own set of datasets that are not shared by any other user. Moreover, even if two users had a dataset in common, the probability that the users prefer the same visualization is very small (almost surely zero) due to the exponential space of visualizations that may arise from a single dataset.

To overcome these issues, we introduce a dataset independent notion of a visualization called a visualization configuration. Using this notion, we propose a novel graph representation that enables us to learn from the *visual preferences* of users despite that they may not have preferred any visualizations from the same datasets. In particular, to learn from visualizations across different datasets and users, we introduce a dataset independent notion called a visualization configuration that removes the dataset dependencies of visualizations, enabling us to capture the general user-level visual preferences of users, independent of the dataset of interest. A visualization configuration is an abstraction of a visualization where instead of mapping specific data attributes to specific design choices of the visualization (*e.g.*, x, y, color, etc.), we replace them with their general type (*e.g.*, numerical, categorical, temporal, ...) or other general property or set of properties that generalize across the different datasets. Most importantly, it is by replacing the data-specific design choices with their general type or set of general properties that enables us to capture and learn from these visualization-configurations. More formally,

Definition 11 (Visual Configuration). *Given a visualization $\mathcal{V}$ consisting of a set of design choices and data (attributes) associated to a subset of the design choices. For every design choice such as chart-type, there is a set of possible options. Other design choices such as color can also be associated to a set of options,* e.g.*, static color definitions, or color map for specific data attributes. Let $\mathcal{T} : \mathbf{x} \rightarrow \mathcal{P}$ be a function that maps an attribute $\mathbf{x}$ of some arbitrary dataset $\mathbf{X} \in \boldsymbol{\mathcal{X}}$ to a property P that generalizes across any arbitrary dataset, and therefore, is independent of the specific dataset. Hence, given attributes $\mathbf{x}$ and $\mathbf{y}$ from two different datasets, then it is possible that $\mathcal{T}(\mathbf{x}) = P$ and $\mathcal{T}(\mathbf{y}) = P$. A visualization configuration is defined as an abstraction of a visualization where every design choice of a visualization that is bound to a data attribute is replaced with a general property of the attribute $\mathcal{T}(\mathbf{x}) = P$.*

Claim 3.2. *There exists $\mathbf{x}$ and $\mathbf{y}$ from different datasets such that $\mathcal{T}(\mathbf{x}) = P$ and $\mathcal{T}(\mathbf{y}) = P$ hold.*

The size of the space of visualization configurations is large since visualization configurations come from all possible combinations of design choices and their values such as,

chart-type: bar, scatter, ...
x-type: quantitative, nominal, ordinal, temporal, ..., none
y-type: quantitative, nominal, ordinal, temporal, ..., none
color: red, green, blue, ...
size: 1pt, 2pt, ...
x-aggregate: sum, mean, bin, ..., none
y-aggregate: sum, mean, bin, ..., none
...

A visualization configuration *and* the attributes selected is everything necessary to generate a visualization. In Figure 1, we provide a toy example showing the process of extracting a data-independent visual-configuration from a visualization. Using the notion of a visualization configuration (Definition 11), we can now introduce a model that captures the visualization preferences of users while ensuring that the visual preferences are not tied to specific datasets. In particular, we define the visual preference matrix **C** as follows:

$$\mathbf{C} = \big[\mathbf{C}\big]_{ij} = \text{\# of times user i clicked visualization configuration j} \tag{27}$$

Note that clicked is simply one such example. Other possibilities of defining **C** (or other similar visual preference matrices) include C_{ij} = # of times user i performed action ∈ {clicked, hovered, liked, added-to-dashboard} visualization configuration j. From our proposed graph model shown in Eq. 27, we can directly learn from user-level visual preferences across different datasets. This novel user-level visual preference graph model for visualization recommendation encodes users and their visual-configurations. Since each visual-configuration node represents a set of design choices that are by definition not tied to a user-specific dataset, then the model can use this user-level visual graph to infer and make connections between other similar visual-configurations likely to be of interest to that user. This new graph model is critical since it allows the learning component to learn from user-level visual preferences (which are visual-configurations) across the different datasets and users. Without this novel component, there would be no way to learn from other users visual preferences (sets of design choices).

The novel notion of a visualization configuration that removes the dataset dependencies of a visualization enabling us to model and learn from the visual preferences of users across different datasets. We introduce the notion of a visualization-configuration, which is an abstraction of a visualization. For instance, suppose we have a json encoding of the actual visualization, that is the design choices + the actual attributes and their data used in the visualization (hence, using this json, we can create the visualization precisely). Recall that this is not very useful for personalized visualization recommendation since the visualization is clearly tied to the specific dataset used by a single user. Hence, if we used visualizations directly, then the optimization method used to optimize an arbitrary objective function to obtain the embeddings for inference would not be able to use other user preferences, since they would also be for visualizations tied to other datasets. To overcome this issue, we propose the novel notion of a visualization-configuration that removes the data-dependency. In particular, given a visualization which includes the design choices + data choices (e.g., data used for the x, y, color attributes), we derive a visualization-configuration from it by replacing the data (attributes) and data attribute names by general properties that are dataset-independent. For instance, in this work, we have used the type of the attribute (e.g., categorical, real-valued, etc.), but we can also use any other general property of the data as well. Most importantly, this new abstraction enables us to learn from users and their visual preferences (design choices), despite that these visual preferences are for visualizations generated for a completely different dataset. This is because we carefully designed the notion of a visualization-configuration to generalize across datasets. In other words, the proposed notion of visualization-configuration are independent of the dataset at hand, and therefore can be shared among users. Notice that traditional recommender systems used in movie or product recommendation are comparatively simple, since these systems assume a single universal dataset (set of movies, set of items/products) that all users share and have feedback about. However, none of these simple assumptions hold in the case of visualization recommendation, and therefore we had to develop and propose these new notions and models for learning.

Notice the matrix **C** encodes the data-independent visual preferences of each user, which is very important. However, this representation does not capture how the visual configurations map to the actual data preferences of the users (attributes and general meta-features that characterize the attributes). Therefore, we also introduce another representation to encode these important associations. In particular, we encode the attributes associated with each visual-configurations as,

$$\mathbf{D} = \big[\mathbf{D}\big]_{kt} = \text{\# of times attribute } k \text{ was used in visual-configuration } t \text{ clicked by some user} \quad (28)$$

As an example, given a relevant visualization $\mathcal{V} = (\mathbf{X}_{ij}^{(k)}, C_t) \in \boldsymbol{\mathcal{V}}_{ij}$ of user $i \in [n]$ for dataset $\mathbf{X}_{ij}$ with attributes $\mathbf{X}_{ij}^{(k)} = [\mathbf{x}_p \, \mathbf{x}_q]$ and visual-configuration $C_t \in \boldsymbol{\mathcal{C}}$, we set $D_{pt} = D_{pt} + 1$ and $D_{qt} = D_{qt} + 1$. We repeat this for all relevant visualizations of each user. In Figure 4, we provide an overview of the proposed personalized visualization recommendation graph model.

3.4 Models for Personalized Visualization Recommendation

We first introduce the PVisRec model that uses the learned meta-feature matrix **M** from Section 3.1 and all the graph representations proposed in Section 3.2 for capturing the shared data preferences between users despite using completely different datasets along with the graph representations from Section 3.3 that capture the visual preferences of users across all datasets in the corpus. Then we discuss two variants of PVisRec that are investigated later in Section 6.

3.4.1 PVisRec: Given the sparse user by attribute adjacency matrix $\mathbf{A} \in \mathbb{R}^{n \times m}$, dense meta-feature by attribute matrix $\mathbf{M} \in \mathbb{R}^{k \times m}$, sparse user by visual-configuration adjacency matrix $\mathbf{C} \in \mathbb{R}^{n \times h}$, and sparse attribute by visual-configuration adjacency matrix $\mathbf{D} \in \mathbb{R}^{m \times h}$, the goal is to find the rank-d embedding matrices **U**, **V**, **Z**, and **Y** that minimize the following objective function:

$$f(\mathbf{U},\mathbf{V},\mathbf{Z},\mathbf{Y}) = \|\mathbf{A} - \mathbf{U}\mathbf{V}^\top\|^2 + \|\mathbf{M} - \mathbf{Y}\mathbf{V}^\top\|^2 + \|\mathbf{C} - \mathbf{U}\mathbf{Z}^\top\|^2 + \|\mathbf{D} - \mathbf{V}\mathbf{Z}^\top\|^2 \quad (29)$$

where $\mathbf{U} \in \mathbb{R}^{n \times d}$, $\mathbf{V} \in \mathbb{R}^{m \times d}$, $\mathbf{Z} \in \mathbb{R}^{h \times d}$, $\mathbf{Y} \in \mathbb{R}^{k \times d}$ are low-rank d-dimensional embeddings of the users, attributes (across all datasets), visual-configurations, and meta-features. Further, the formulation above uses squared error, though other loss functions can also be used (e.g., Bregman divergences) [Singh and Gordon 2008]. We can solve Eq. 29 by computing the gradient and then using a first-order optimization method [Schenker et al. 2021]. Afterwards, we have

$$\mathbf{A} \approx \mathbf{A}' = \mathbf{U}\mathbf{V}^\top = \sum_{r=1}^{d} \mathbf{u}_r \mathbf{v}_r^\top \quad (30)$$

$$\mathbf{M} \approx \mathbf{M}' = \mathbf{Y}\mathbf{V}^\top = \sum_{r=1}^{d} \mathbf{y}_r \mathbf{v}_r^\top \quad (31)$$

$$\mathbf{C} \approx \mathbf{C}' = \mathbf{U}\mathbf{Z}^\top = \sum_{r=1}^{d} \mathbf{u}_r \mathbf{z}_r^\top \quad (32)$$

$$\mathbf{D} \approx \mathbf{D}' = \mathbf{V}\mathbf{Z}^\top = \sum_{r=1}^{d} \mathbf{v}_r \mathbf{z}_r^\top \quad (33)$$

Solving Eq. 29 corresponds to the *PVisRec* model investigated later in Section 6. We also investigate a few different variants of the PVisRec model from Eq. 29 later in Section 6. In particular, the model variants of PVisRec use only a subset of the graph representations $\{\mathbf{A}, \mathbf{C}, \mathbf{D}\}$ and/or dense meta-feature matrix **M** introduced previously in Section 3.1-3.3.

*3.4.2 PVisRec (**A**,**C**,**M** only):* Given the user by attribute matrix $\mathbf{A} \in \mathbb{R}^{n \times m}$, meta-feature by attribute matrix $\mathbf{M} \in \mathbb{R}^{k \times m}$, and user by visual-configuration matrix $\mathbf{C} \in \mathbb{R}^{n \times h}$, the goal is to find the rank-d embedding matrices **U**, **V**, **Z**, and **Y** that minimize the following objective function:

$$f(\mathbf{U},\mathbf{V},\mathbf{Z},\mathbf{Y}) = \|\mathbf{A} - \mathbf{U}\mathbf{V}^\top\|^2 + \|\mathbf{M} - \mathbf{Y}\mathbf{V}^\top\|^2 + \|\mathbf{C} - \mathbf{U}\mathbf{Z}^\top\|^2 \quad (34)$$

*3.4.3 PVisRec (**A**,**C**,**D** only).* Besides Eq. 34 that uses only **A**, **M**, and **C**, we also investigate another personalized visualization recommendation model that uses **A**, **C**, and **D** (without meta-features). More formally, given **A**, **C**, and **D**, then the problem is to learn low-dimensional rank-d embedding matrices **U**, **V**, and **Z** that minimize the following:

$$f(\mathbf{U},\mathbf{V},\mathbf{Z}) = \|\mathbf{A} - \mathbf{U}\mathbf{V}^\top\|^2 + \|\mathbf{C} - \mathbf{U}\mathbf{Z}^\top\|^2 + \|\mathbf{D} - \mathbf{V}\mathbf{Z}^\top\|^2 \tag{35}$$

In this work, we used an ALS-based optimizer to solve Eq. 29 and the simpler variants shown in Eq. 34 and Eq. 35. However, we can also leverage a variety of different optimization schemes including cyclic/block coordinate descent [Kim et al. 2014; Rossi and Zhou 2016], stochastic gradient descent [Oh et al. 2015; Yun et al. 2014], among others [Balasubramaniam et al. 2020; Bouchard et al. 2013; Choi et al. 2019; Schenker et al. 2021; Singh and Gordon 2008].

3.5 Inferring Personalized Visualization Recommendations for Individual Users

We first discuss using the personalized visualization recommendation model for recommending attributes to users as well as visual-configurations. Then we discuss the fundamentally more challenging task of personalized visualization recommendation.

3.5.1 Personalized Attribute Recommendation. The ranking of attributes for user i is induced by $\mathbf{U}_{i,:}\mathbf{V}^\top$ where $\mathbf{U}_{i,:}$ is the embedding of user i. Let $\pi_1(\mathbf{U}_{i,:}\mathbf{V}^\top)$ denote the largest attribute weight for user i. Therefore, the top-k attribute weights for user i are denoted as:

$$\pi_1(\mathbf{U}_{i,:}\mathbf{V}^\top), \pi_2(\mathbf{U}_{i,:}\mathbf{V}^\top), \ldots, \pi_k(\mathbf{U}_{i,:}\mathbf{V}^\top)$$

3.5.2 Personalized Visual-Configuration Recommendation. The personalized ranking of the visual-configurations for user i is inferred by $\mathbf{U}_{i,:}\mathbf{Z}^\top$ where $\mathbf{U}_{i,:}$ is the embedding of user i and **Z** is the matrix of visual-configuration embeddings. Hence, $\mathbf{U}_{i,:}\mathbf{Z}^\top \in \mathbb{R}^h$ is an h-dimensional vector of weights indicating the likelihood/importance of each visual-configuration for that specific user i. Let $\pi_1(\mathbf{U}_{i,:}\mathbf{Z}^\top)$ denote the largest visual-configuration weight for user i. Therefore, the top-k visual-configuration weights for user i is denoted as:

$$\pi_1(\mathbf{U}_{i,:}\mathbf{Z}^\top), \pi_2(\mathbf{U}_{i,:}\mathbf{Z}^\top), \ldots, \pi_k(\mathbf{U}_{i,:}\mathbf{Z}^\top)$$

3.5.3 Personalized Visualization Recommendation. We now focus on the most complex and challenging problem of recommending complete visualizations personalized for a specific user $i \in [n]$. A recommended visualization for user $i \in [n]$ consists of both the subset of *attributes* $\mathbf{X}^{(k)}$ from some dataset **X** and the *design choices* C_t (a visual-configuration) for those attributes. Given user i along with an arbitrary visualization $\mathcal{V} = (\mathbf{X}^{(k)}, C_t)$ generated from some dataset **X** of interest to user i, we derive a personalized user-specific score for visualization $\mathcal{V}$ (for user i) as,

$$\widehat{y}(\mathcal{V}) = \mathbf{U}_{i,:}\mathbf{Z}_{t,:}^\top \prod_{\mathbf{x}_j \in \mathbf{X}^{(k)}} \mathbf{U}_{i,:}\mathbf{V}_{j,:}^\top \tag{36}$$

where $\mathbf{X}^{(k)}$ is the subset of attributes from the users dataset **X** (hence, $|\mathbf{X}^{(k)}| \leq |\mathbf{X}|$) used in the visualization $\mathcal{V}$ and $C_t \in \mathcal{C}$ is the visual-configuration of the visualization $\mathcal{V}$ being scored for user i. Using Eq. 36, we can predict the personalized visualization score $\widehat{y}(\mathcal{V})$ for any arbitrary visualization $\mathcal{V}$ (for any dataset) and user $i \in [n]$. For evaluation in Section 6.1, we use Eq. 36 to score relevant and non-relevant visualizations for a specific user and dataset of interest.

4 DEEP PERSONALIZED VISUALIZATION RECOMMENDATION MODELS

We now introduce a deep neural network architecture for personalized visualization recommendation. For this, we combine the previously proposed model with a deep multilayer neural network component to learn non-linear functions that capture complex dependencies and patterns between users and their visualization preferences.

4.0.1 Neural PVisRec. Given an arbitrary user i and a visualization $\mathcal{V} = (\mathbf{X}_{ij}^{(k)}, C_t)$ to score from some new dataset of interest to that user, we first must decide on the input representation. In this work, we leverage the user personalized embeddings learned in Section 3.4 by concatenating the embedding of user i, visual configuration t, along with the embeddings for each attribute used in the visualization. More formally,

$$\phi(\mathcal{V} = \langle \mathbf{X}_{ij}^{(k)}, C_t \rangle) = \begin{bmatrix} \mathbf{u}_i \\ \mathbf{z}_t \\ \mathbf{v}_{r_1} \\ \vdots \\ \mathbf{v}_{r_s} \end{bmatrix} \tag{37}$$

where $\mathbf{u}_i$ is the embedding of user i, $\mathbf{z}_t$ is the embedding of the visual-configuration C_t, and $\mathbf{v}_{r_1}, \ldots, \mathbf{v}_{r_s}$ are the embeddings of the attributes used in the visualization being scored for user i. This can be written as,

$$\phi(\mathcal{V} = \langle \mathbf{X}_{ij}^{(k)}, C_t \rangle) = \begin{bmatrix} \mathbf{U}^\top \mathbf{e}_i & \mathbf{Z}^\top \mathbf{e}_t & \mathbf{V}^\top \mathbf{e}_{r_1} & \cdots & \mathbf{V}^\top \mathbf{e}_{r_s} \end{bmatrix}^\top \tag{38}$$

where $\mathbf{e}_i \in \mathbb{R}^n$ (user i), $\mathbf{e}_t \in \mathbb{R}^h$ (visual-configuration C_t), and $\mathbf{e}_{r_1} \in \mathbb{R}^m$ (attribute r_1) are the one-hot encodings of the user i, visual-configuration t, and attributes $r_1, \ldots, r_s$ used in the visualization. Note that $\mathbf{U} \in \mathbb{R}^{n \times d}$, $\mathbf{V} \in \mathbb{R}^{m \times d}$, $\mathbf{Z} \in \mathbb{R}^{h \times d}$, $\mathbf{Y} \in \mathbb{R}^{k \times d}$.

The first *neural personalized visualization recommendation* architecture that we introduce called *Neural PVisRec* leverages the user, visual-configuration, and attribute embeddings from the PVisRec model in Section 3.4 as input into a deep multilayer neural network with L fully-connected layers,

$$\phi(\mathcal{V} = \langle \mathbf{X}_{ij}^{(k)}, C_t \rangle) = \begin{bmatrix} \mathbf{U}^\top \mathbf{e}_i & \mathbf{Z}^\top \mathbf{e}_t & \mathbf{V}^\top \mathbf{e}_{r_1} & \cdots & \mathbf{V}^\top \mathbf{e}_{r_s} \end{bmatrix}^\top \tag{39}$$

$$\mathbf{q}_1 = \sigma_1(\mathbf{W}_1 \phi(\mathcal{V}) + \mathbf{b}_1) \tag{40}$$

$$\mathbf{q}_2 = \sigma_2(\mathbf{W}_2 \mathbf{q}_1 + \mathbf{b}_2) \tag{41}$$

$$\vdots$$

$$\mathbf{q}_L = \sigma_L(\mathbf{W}_L \mathbf{q}_{L-1} + \mathbf{b}_L) \tag{42}$$

$$\widehat{y} = \sigma(\mathbf{h}^\top \mathbf{q}_L) \tag{43}$$

where $\mathbf{W}_L$, $\mathbf{b}_L$, and σ_L are the weight matrix, bias vector, and activation function for layer L. Further, $\widehat{y} = \sigma(\mathbf{h}^\top \mathbf{q}_L)$ (Eq. 43) is the output layer where σ is the output activation function and $\mathbf{h}^\top$ denotes the edge weights of the output function. For the hidden layers, we used ReLU as the activation function. Note that if the visualization does not use all s attributes, then we can pad the remaining unused attributes with zeros. This enables the multi-layer neural network architecture to be flexible for visualizations with any number of attributes. Eq. 39-43 can be written more succinctly as

$$\widehat{y} = \sigma\big(\mathbf{h}^\top \sigma_L(\mathbf{W}_L(\ldots\sigma_1(\mathbf{W}_1 \begin{bmatrix} \mathbf{U}^\top \mathbf{e}_i & \mathbf{Z}^\top \mathbf{e}_j & \mathbf{V}^\top \mathbf{e}_{r_1} \cdots \mathbf{V}^\top \mathbf{e}_{r_s} \end{bmatrix}^\top + \mathbf{b}_1)\ldots) + \mathbf{b}_L)\big) \tag{44}$$

where $\widehat{y}$ is the predicted visualization score for user i.

4.0.2 Neural PVisRec-CMF. We also investigated a second neural approach for the personalized visualization recommendation problem. This approach combines scores from PVisRec and Eq. 44. More formally, given user i along with an arbitrary visualization $\mathcal{V} = (\mathbf{X}_{ij}^{(k)}, C_t)$ generated from some dataset $\mathbf{X}_{ij}$ of interest to user i, we derive a personalized user-specific score for visualization

$\mathcal{V}$ (for user i) as $\widehat{y}_{\text{PVisRec}} = \mathbf{U}_{i,:}\mathbf{Z}_{t,:}^{\top}\prod_{\mathbf{x}_j \in \mathbf{X}^{(k)}} \mathbf{U}_{i,:}\mathbf{V}_{j,:}^{\top}$ where $\mathbf{X}_{ij}^{(k)}$ is a subset of attributes used in the visualization $\mathcal{V}$ from the users dataset $\mathbf{X}_{ij}$ (hence, $|\mathbf{X}^{(k)}| \leq |\mathbf{X}_{ij}|$) and $C_t \in \boldsymbol{C}$ is the visual-configuration for visualization $\mathcal{V}$. Then, we have

$$\widehat{y} = (1-\alpha)\left(\mathbf{U}_i\mathbf{Z}_t^{\top}\prod_{\mathbf{x}_j \in \mathbf{X}^{(k)}} \mathbf{U}_i\mathbf{V}_j^{\top}\right) + \alpha\widehat{y}_{\text{dnn}} \tag{45}$$

where $\widehat{y}_{\text{dnn}} = \sigma\big(\mathbf{h}^{\top}\sigma_L(\mathbf{W}_L(\ldots\sigma_1(\mathbf{W}_1\phi(\mathcal{V})+\mathbf{b}_1)\ldots)+\mathbf{b}_L)\big)$ with $\phi(\mathcal{V}) = \begin{bmatrix}\mathbf{U}^{\top}\mathbf{e}_i & \mathbf{Z}^{\top}\mathbf{e}_t & \mathbf{V}^{\top}\mathbf{e}_{r_1} & \cdots & \mathbf{V}^{\top}\mathbf{e}_{r_s}\end{bmatrix}^{\top}$ and $\alpha \in (0,1)$ is a hyperparameter that controls the influence of the models on the final predicted score of the visualization for user i.

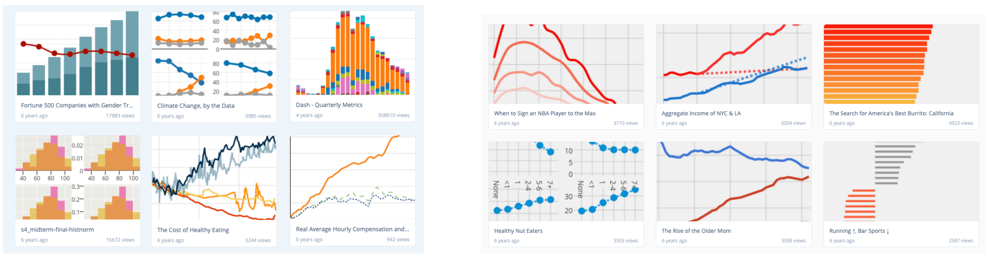
All layers of the various neural architectures for our personalized visualization recommendation problem use ReLU nonlinear activation. Unless otherwise mentioned, we used three hidden layers and optimized model parameters using mini-batch Adam with a learning rate of 0.001. We designed the neural network structure such that the bottom layers are the widest and each successive layer has 1/2 the number of neurons. For fairness, the last hidden layer is set to the embedding size. Hence, if the embedding size is 8, then the architecture of the layers is $32 \longrightarrow 16 \longrightarrow 8$.

4.0.3 Training. The user-centric visualization training corpus $\boldsymbol{\mathcal{D}} = \{\boldsymbol{\mathcal{X}}_i, \mathbb{V}_i\}_{i=1}^{n}$ for *personalized* visualization recommendation consists of user-level training data for n users where for each user $i \in [n]$ we have a set of datasets $\boldsymbol{\mathcal{X}}_i = \{\mathbf{X}_{i1}, \ldots, \mathbf{X}_{ij}, \ldots\}$ of interest to that user along with user i's "relevant" (generated, liked, clicked-on) visualizations $\mathbb{V}_i = \{\boldsymbol{\mathcal{V}}_{i1}, \ldots, \boldsymbol{\mathcal{V}}_{ij}, \ldots\}$ for each of those datasets. For each user $i \in [n]$ and dataset $\mathbf{X}_{ij} \in \boldsymbol{\mathcal{X}}_i$ of interest to user i, there is a set $\boldsymbol{\mathcal{V}}_{ij} = \{\ldots, \mathcal{V} = (\mathbf{X}_{ij}^{(k)}, C_{ijk}), \ldots\}$ of relevant (positive) visualizations for that user, and we also leverage a sampled set of non-relevant (negative) visualizations $\boldsymbol{\mathcal{V}}_{ij}^{-}$ for that user i and dataset $\mathbf{X}_{ij} \in \boldsymbol{\mathcal{X}}_i$. Therefore, the set of training visualizations for user $i \in [n]$ and dataset $\mathbf{X}_{ij} \in \boldsymbol{\mathcal{X}}_i$ is $\boldsymbol{\mathcal{V}}_{ij} \cup \boldsymbol{\mathcal{V}}_{ij}^{-}$ and $Y_{ijk} \in \{0,1\}$ denotes the ground-truth label of visualization $\mathcal{V} = (\mathbf{X}_{ij}^{(k)}, C_{ijk}) \in \boldsymbol{\mathcal{V}}_{ij} \cup \boldsymbol{\mathcal{V}}_{ij}^{-}$. Hence, $Y_{ijk} = 1$ indicates a user-relevant (positive) visualization for user i whereas $Y_{ijk} = 0$ indicates a non-relevant visualization for that user, *i.e.*, $\mathcal{V} = (\mathbf{X}_{ij}^{(k)}, C_{ijk}) \in \boldsymbol{\mathcal{V}}_{ij}^{-}$. The goal is to have the model score $\widehat{Y}_{ijk} \in [0,1]$ each training visualization $\mathcal{V} = (\mathbf{X}_{ij}^{(k)}, C_{ijk}) \in \boldsymbol{\mathcal{V}}_{ij} \cup \boldsymbol{\mathcal{V}}_{ij}^{-}$ for a user i as close as possible to the ground-truth label Y_{ijk}. The neural personalized visualization recommendation model is learned by optimizing the likelihood of model scores for all visualizations of each user. Given a user $i \in [n]$ and the model parameters Θ, the likelihood is

$$\mathrm{P}(\widehat{\mathbb{V}}_i^{-}, \mathbb{V}_i|\Theta) = \prod_{j=1}^{|\boldsymbol{\mathcal{X}}_i|} \prod_{(\mathbf{X}_{ij}^{(k)}, C_{ijk}) \in \boldsymbol{\mathcal{V}}_{ij}} \widehat{Y}_{ijk} \prod_{(\mathbf{X}_{ij}^{(k)}, C_{ijk}) \in \widehat{\boldsymbol{\mathcal{V}}}_{ij}^{-}} \left(1 - \widehat{Y}_{ijk}\right), \quad \text{for } i = 1, \ldots, n \tag{46}$$

where $\widehat{Y}_{ijk}$ is the predicted score of a visualization $\mathcal{V} = (\mathbf{X}_{ij}^{(k)}, C_{ijk})$ for user i and dataset j ($\mathbf{X}_{ij} \in \boldsymbol{\mathcal{X}}_i$). Naturally, the goal is to obtain $\widehat{Y}_{\mathcal{V}}$ such that it is as close as possible to the actual ground-truth $Y_{\mathcal{V}}$. Taking the negative log of the likelihood in Eq. 46 and summing over all n users and their sets of relevant visualizations $\boldsymbol{\mathcal{V}}_{ij}$ from $|\boldsymbol{\mathcal{X}}_i|$ different datasets give us the total loss $\mathbb{L}$.

$$\begin{aligned}
\mathbb{L} &= \sum_{i=1}^{n}\sum_{j=1}^{|\boldsymbol{\mathcal{X}}_i|}\left(-\sum_{(\mathbf{X}_{ij}^{(k)}, C_{ijk}) \in \boldsymbol{\mathcal{V}}_{ij}} \log \widehat{Y}_{ijk} - \sum_{(\mathbf{X}_{ij}^{(k)}, C_{ijk}) \in \widehat{\boldsymbol{\mathcal{V}}}_{ij}^{-}} \log(1 - \widehat{Y}_{ijk})\right) \\
&= -\sum_{i=1}^{n}\sum_{j=1}^{|\boldsymbol{\mathcal{X}}_i|} \sum_{(\mathbf{X}_{ij}^{(k)}, C_{ijk}) \in \boldsymbol{\mathcal{V}}_{ij} \cup \widehat{\boldsymbol{\mathcal{V}}}_{ij}^{-}} Y_{ijk} \log \widehat{Y}_{ijk} + (1 - Y_{ijk}) \log(1 - \widehat{Y}_{ijk})
\end{aligned} \tag{47}$$


Fortune 500 Companies with Gender Tr...
Climate Change, by the Data
Dash - Quarterly Metrics
s4_midterm-final-histnorm
The Cost of Healthy Eating
Real Average Hourly Compensation and...
When to Sign an NBA Player to the Max
Aggregate Income of NYC & LA
The Search for America's Best Burrito: California
Healthy Nut Eaters
The Rise of the Older Mom
Running ↑, Bar Sports ↓

(a) Community Feed (b) User Feed

Fig. 5. Plot.ly Community and User Feed Examples. We provide an example of the global community feed and a specific users visualization feed.

where the objective function above is minimized via stochastic gradient descent (SGD) to update the model parameters Θ in $\mathcal{M}$.

5 BENCHMARK DATA FOR PERSONALIZED VISUALIZATION RECOMMENDATION

Since this is the first work that addresses the *personalized* visualization recommendation problem, there were not any existing public datasets that could be used directly for our problem. Recent works have ignored the user information [Hu et al. 2019a; Qian et al. 2020] that details the "author" of the visualization, which is required in this work for user-level personalization. As an aside, VizML [Hu et al. 2019a] discarded all user information and only kept the attributes used in an actual visualization (and therefore did not consider datasets as well). Both decisions were made based on the underlying problem focused on in that work, which was classifying the chart type of a visualization, as well as a few other simple design choice classification tasks.

In this work, since we focus on the personalized visualization recommendation problem, we derive a user-centered dataset where for each user we know their datasets, visualizations, attributes, and visualization-configurations used. We started from the raw Plot.ly community feed data [Plotly 2018]. The community feed is a place where users can post their visualizations to share with others. An example of the global community feed (along with an example of an individual users feed) is provided in Figure 5. For the personalized visualization recommendation problem, we first extract the set of all n users in the visualization corpus. For each user $i \in [n]$, we then extract the set of datasets $\boldsymbol{\mathcal{X}}_i$ of interest to that user. These are the datasets that user i has generated at least one visualization. Depending on the visualization corpus data, this could also be other types of user feedback such as a visualization that a user liked or clicked. Next, we extract the set of user-preferred visualizations $\boldsymbol{\mathcal{V}}_{ij}$ for each of the datasets $\mathbf{X}_{ij} \in \boldsymbol{\mathcal{X}}_i$ of interest to user i. Hence, $\boldsymbol{\mathcal{V}}_{ij}$ is the set of visualizations generated (or liked, clicked, ...) by user i for dataset j ($\mathbf{X}_{ij}$). Every visualization $\mathcal{V} \in \boldsymbol{\mathcal{V}}_{ij}$ preferred by user i also obviously contains the attributes from dataset $\mathbf{X}_{ij} \in \boldsymbol{\mathcal{X}}_i$ used in the visualization (*i.e.*, the attributes that map to the x, y, binning, color, and so on).

In Table 4, we report statistics about the personalized visualization corpus used in our work, including the number of users, attributes, datasets, visualizations, and visualization-configurations extracted from all the user-generated visualizations, and so on. The corpus $\boldsymbol{\mathcal{D}} = \{\boldsymbol{\mathcal{X}}_i, \mathbb{V}_i\}_{i=1}^{n}$ for learning individual personalized visualization recommendation models consists of a total of $n = 17,469$ users with $|\bigcup_{i=1}^{n} \boldsymbol{\mathcal{X}}_i| = 94,419$ datasets used by those users. Further, there are $m = 2,303,033$ attributes among the $94,419$ datasets of interest by the $17.4k$ users. Our user-centric visualization training corpus $\boldsymbol{\mathcal{D}}$ has a total of $|\bigcup_{i=1}^{n} \mathbb{V}_i| = 32{,}318$ relevant visualizations generated by the $n = 17.4k$ users with an average of 1.85 relevant visualizations per user. Each user in the

Table 4. Personalized visualization recommendation corpus. These user-centric dataset is used for learning personalized visualization recommendation models for individual users.

# Users	17,469
# Datasets	94,419
# Attributes	2,303,033
# Visualizations	32,318
# Vis. Configs	686
# Meta-features	1006
mean # attr. per dataset	24.39
mean # attr. per user	51.63
mean # vis. per user	1.85
mean # datasets per user	5.41
Density (**A**)	<0.0001
Density (**C**)	<0.0001
Density (**D**)	<0.0001
Density (**M**)	0.4130

corpus has an average of 5.41 datasets and each dataset has an average of 24.39 attributes. From the 32.3k user-relevant visualizations from the 17.4k users, we extracted a total of $|\mathcal{C}| = 686$ unique visual-configurations. To further advance research on personalized visualization recommender systems, we have made the user-level plot.ly data that we used for studying the personalized visualization recommendation problem (introduced in Section 2.4) publicly accessible at:

https://networkrepository.com/personalized-vis-rec

We have also made the graph representations used in our personalized visualization recommendation framework publicly accessible[2]

6 EXPERIMENTS

To investigate the effectiveness of the *personalized visualization recommendation* approach, we design experiments to answer the following research questions:

- **RQ1:** Given a user and a new dataset of interest to that user, can we accurately recommend the top most relevant visualizations for that specific user (Section 6.1)?
- **RQ2:** How does our user-level personalized visualization recommendations compare to the non-personalized global recommendations (Section 6.2)?
- **RQ3:** Can we significantly reduce the space requirements of our approach by trading off a small amount of accuracy for a large improvement in space (Section 6.3)?
- **RQ4:** Does the neural personalized visualization recommendation models further improve the performance when incorporating a multilayer deep neural network component (Section 6.4)?

6.1 Personalized Visualization Recommendation Results

6.1.1 Experimental setup. Now we evaluate the system for recommending personalized visualizations to a user. Given an arbitrary user, we know the visualization(s) they preferred for each of the datasets of interest to them, which can serve as ground-truth examples for effective and quantitative evaluation of our approach. Therefore, we can quantitatively evaluate the proposed approach for

[2]https://networkrepository.com/personalized-vis-rec-graphs

Table 5. Personalized Visualization Recommendation Results. Note $d = 10$. See text for discussion.

	HR@K					NDCG@K				
Model	@1	@2	@3	@4	@5	@1	@2	@3	@4	@5
VizRec	N/A	N/A	N/A	N/A	N/A	N/A	N/A	N/A	N/A	N/A
VisPop	0.186	0.235	0.255	0.271	0.289	0.181	0.214	0.224	0.231	0.238
VisConfigKNN	0.026	0.030	0.038	0.055	0.089	0.016	0.021	0.026	0.034	0.048
VisKNN	0.147	0.230	0.297	0.372	0.449	0.143	0.195	0.227	0.257	0.286
eALS	0.304	0.395	0.426	0.441	0.449	0.302	0.360	0.376	0.382	0.385
MLP	0.218	0.452	0.601	0.671	0.715	0.211	0.357	0.435	0.465	0.483
PVisRec	**0.630**	**0.815**	**0.876**	**0.906**	**0.928**	**0.624**	**0.743**	**0.775**	**0.788**	**0.796**

personalized visualization recommendation by holding out a set of user-relevant/ground-truth visualizations, training the model with the remaining data, and then evaluating whether our approach can recover the actual ground-truth visualizations that a user preferred. For each user, we randomly select one of their datasets where the user has manually created at least two visualizations (treated as positive examples), and randomly select one of those positive visualizations to use for testing, and the other positive instances are used for training and validation. This is similar to leave-one-out evaluation which is widely used in traditional user-item recommender systems [He et al. 2016]. However, in our case, we have thousands of datasets, and for each dataset there are a large and completely *disjoint* set of possible visualizations to recommend to that user.[3] Thus, we randomly sample 19 visualizations that were not created by the user. To obtain these visualizations, we use a rule-based visualization generator called CompassQL [Wongsuphasawat et al. 2016a] to generate grammatically correct candidate visualizations.[4] This gives us a total of 20 visualizations per user (1 relevant + 19 non-relevant visualizations) to use for evaluation of the personalized visualization recommendations from our proposed models. Using this held-out ground-truth set of relevant visualizations that a user found interesting/preferred, we evaluate the ability of the proposed approach to recommend these held-out relevant visualizations to the user (which are visualizations the user actually created), among the exponential amount of alternative visualizations (that arise for a single dataset of interest) from a set of attributes and sets of design choices (*e.g.*, chart-types, ...). In particular, given a user i and a dataset of interest to that user, we use the proposed approach to recommend the top-k visualizations personalized for that specific user and dataset. To quantitatively evaluate the personalized ranking of visualizations given by the proposed personalized visualization recommendation models, we use rank-based evaluation metrics including Hit Ratio at K (HR@K) and Normalized Discounted Cumulative Gain (NDCG@K) [He et al. 2016]. Intuitively, HR@K quantifies whether the held-out relevant (user generated) visualization appears in the top-K ranked visualizations or not. Similarly, NDCG@K takes into account the position of the relevant (user generated) visualization in the top-K ranked list of visualizations, by assigning larger scores to visualizations ranked more highly in the list. For both HR@K and NDCG@K, we report $K = 1, \ldots, 5$ unless otherwise mentioned.

Therefore, given a user i along with the set of relevant and non-relevant visualizations $\boldsymbol{\mathcal{V}}_{ij} \cup \boldsymbol{\mathcal{V}}_{ij}^{-}$ for that user and their dataset $\mathbf{X}_{ij} \in \boldsymbol{\mathcal{X}}_i$ of interest, we derive a score for each of the visualizations $\mathcal{V} \in (\boldsymbol{\mathcal{V}}_{ij} \cup \boldsymbol{\mathcal{V}}_{ij}^{-})$ where $|\boldsymbol{\mathcal{V}}_{ij}| + |\boldsymbol{\mathcal{V}}_{ij}^{-}| = 20$. An effective personalized visualization recommender will assign a larger score to the relevant visualizations and smaller scores to the non-relevant

[3]The set of candidate visualizations for a specific dataset are not only disjoint (*i.e.*, completely different from any other set of visualizations generated from another dataset), but the amount of possible visualizations for a given dataset are exponential in the number of attributes, possible design choices, and so on, making this problem unique and fundamentally challenging.
[4]This ensures that visualizations are not grammatically incorrect such as a bar chart with x-axis being continuous.

visualizations, hence, the relevant visualizations will show up first, followed by the non-relevant visualizations (which should appear further down the list). Unless otherwise mentioned, we use $d = 10$ as the embedding size and use the full meta-feature matrix $\mathbf{M}$. For the neural variants of our approach, we use $\alpha = 0.5$.

Table 6. Ablation study results for different variants of our personalized visualization recommendation approach.

	HR@K					NDCG@K				
Model	@1	@2	@3	@4	@5	@1	@2	@3	@4	@5
PVisRec (**A**,**C**,**M** only)	0.307	0.416	0.470	0.488	0.501	0.306	0.374	0.401	0.410	0.415
PVisRec (**A**,**C**,**D** only)	0.414	0.474	0.537	0.610	0.697	0.384	0.435	0.450	0.457	0.460
PVisRec	**0.630**	**0.815**	**0.876**	**0.906**	**0.928**	**0.624**	**0.743**	**0.775**	**0.788**	**0.796**

6.1.2 Baselines. Since the personalized visualization recommendation problem introduced in Section 2 is new, there are not any existing vis. rec. methods that can be directly applied to solve it. For instance, VizRec [Mutlu et al. 2016] is the closest existing approach, though is unable to be used since it explicitly assumes a single dataset where users provide feedback about visualizations pertaining to that dataset of interest. However, in our problem formulation and corpus $\mathcal{D} = \{(\boldsymbol{\mathcal{X}}_i, \mathbb{V}_i)\}_{i=1}^n$, every user $i \in [n]$ can have their own set of datasets $\boldsymbol{\mathcal{X}}_i$ that are not shared by any other user. Nevertheless, we adapted a wide variety of methods to use as baselines for evaluation. We now briefly summarize these methods below:

- **VisPop**: Given a visualization $\mathcal{V}$ with attributes $\mathbf{X}^{(k)}$ and visual-configuration $C \in \boldsymbol{\mathcal{C}}$, the score of visualization $\mathcal{V}$ is $\phi(V) = f(C) \prod_{\mathbf{x} \in \mathbf{X}^{(k)}} f(\mathbf{x})$ where $f(\mathbf{x})$ is the frequency of attribute $\mathbf{x}$ (sum of the columns of $\mathbf{A}$) and $f(C)$ is the frequency of visual-configuration C. Hence, the score given by VisPop is a product of the frequencies of the underlying visualization components, *i.e.*, visual-configuration and attributes used in the visualization being scored.
- **VisKNN**: This is the standard item-based collaborative filtering method adapted for the visualization recommendation problem. Given a visualization $\mathcal{V}$ with attributes $\mathbf{X}^{(k)}$ and visual-configuration $C \in \boldsymbol{\mathcal{C}}$, then we score $\mathcal{V}$ by taking the mean score of the visual configurations most similar to C, along with the mean score of the top attributes most similar to each of the attributes used in the visualization.
- **VisConfigKNN**: This approach is similar to VisKNN, but uses only the visual-configuration matrix to score the visualizations.
- **eALS**: This is an adapted version of the state-of-the-art MF method used for item recommendation in [He et al. 2016]. We adapted it for our visualization recommendation problem by minimizing squared loss while treating all unobserved user iterations between attributes and visual-configurations as negative examples, which are weighted non-uniformly by the frequency of attributes and visual-configurations.
- **MLP**: We used three hidden layers and optimized model parameters using mini-batch Adam with a learning rate of 0.001. For the activation functions of the MLP layers, we used ReLU. For fairness, the last hidden layer is set to the embedding size.
- **VizRec** [Mutlu et al. 2016]: For each dataset, this approach constructs a user-by-visualization matrix and uses it to obtain the average overall rating among the similar users of a visualization, where a user is similar if it has rated a visualization preferred by the active user. VizRec assumes a single dataset and is only applicable when there are a large number of users that have rated visualizations from the *same* dataset.

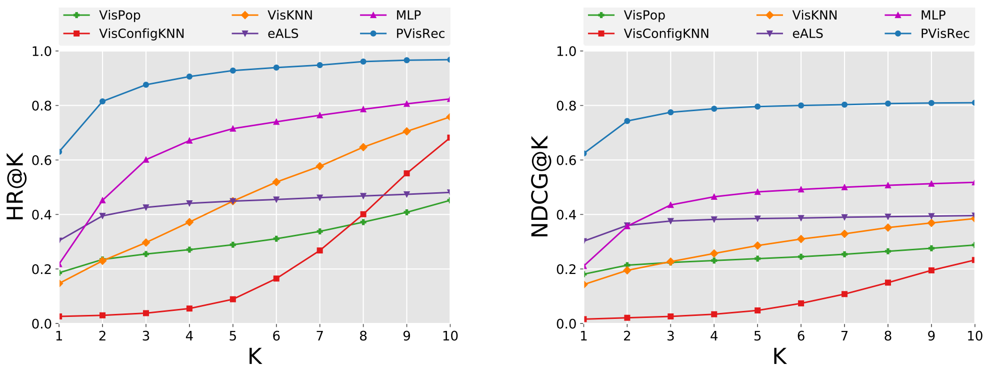
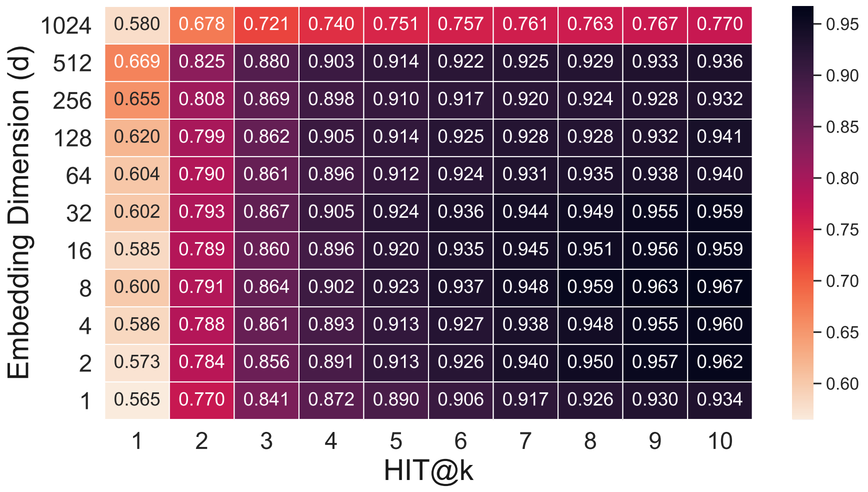


Fig. 6. Evaluation of top-K personalized visualization recommendations.

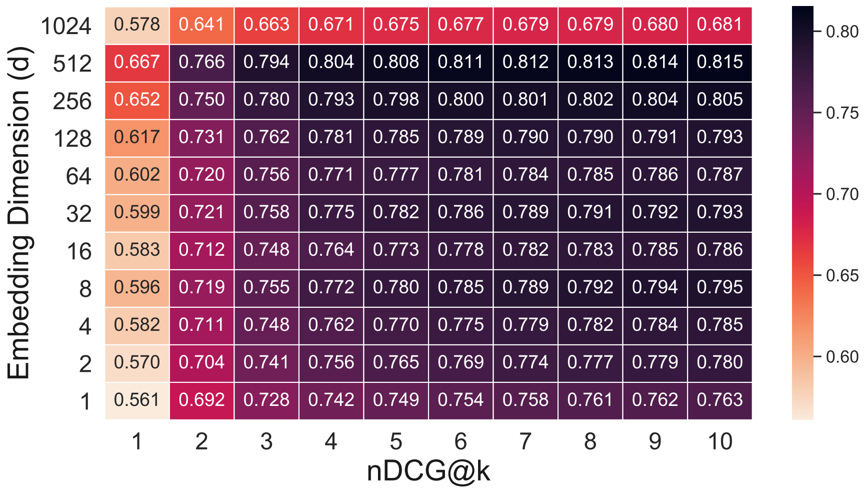
6.1.3 Results. We provide the results in Table 5. Overall, the proposed approach, PVisRec, significantly outperforms the baseline methods by a large margin as shown in Table 5. Strikingly, PVisRec consistently achieves the best HR@K and NDCG@K across all $K = 1, 2, \ldots, 5$. From Table 5, we see that PVisRec achieves a mean relative improvement of 107.2% and 106.6% over the best performing baseline method (eALS) for HR@1 and NDCG@1, respectively. Comparing HR@5 and NDCG@5, PVisRec achieves a mean improvement of 29.8% and 64.9% over the next best performing method (MLP). As an aside, VizRec is the only approach proposed for ranking visualizations. All other methods used in our comparison are new and to the best of our knowledge have never been extended for ranking and recommending visualizations. Recall that VizRec in itself solves a different problem, but we point out the above since it is clearly the closest. As discussed in Section 8, all of the assumptions required by VizRec are unrealistic in practice. This is also true when using VizRec for our problem and corpus $\mathcal{D} = \{(\mathcal{X}_i, \mathbb{V}_i)\}_{i=1}^{n}$ where every user $i \in [n]$ can have their own set of datasets $\mathcal{X}_i$ that are not shared by any other user. In such cases, we use "N/A" to denote this fact. This is due to the VizRec assumption that there is a single dataset of interest by all n users, and every user has given many different preferences on the relevant visualizations generated for that specific dataset. All of these assumptions are violated in our problem. Figure 6 shows the mean performance of the top-K visualization recommendations for $K = 1, 2, \ldots, 10$. These results demonstrate the effectiveness of our user personalized visualization recommendation approach as we are able to successfully recommend users the held-out ground-truth visualizations that they preferred.

6.1.4 Ablation Study Results. Previously, we observed that PVisRec significantly outperforms other methods for the personalized visualization recommendation problem. To understand the importance of the different model components of PVisRec, we investigate a few different variants of our personalized visualization recommendation model. The first variant called PVisRec ($\mathbf{A}, \mathbf{C}, \mathbf{M}$ only) does not use the attribute by visual-configuration graph represented by the sparse adjacency matrix $\mathbf{D}$ whereas the second variant called PVisRec ($\mathbf{A}, \mathbf{C}, \mathbf{D}$ only) does not use the dense meta-feature matrix $\mathbf{M}$ for learning. This is in contrast to PVisRec that uses $\mathbf{A}, \mathbf{C}, \mathbf{D}$ and $\mathbf{M}$. In Table 6, we see that both variants perform worse than PVisRec, indicating the importance of using all the graph representations for learning the personalized visualization recommendation model. Further, PVisRec ($\mathbf{A}, \mathbf{C}, \mathbf{D}$ only) outperforms the other variant across both ranking metrics and across all K. This suggests that $\mathbf{D}$ may be more important for learning than $\mathbf{M}$. Nevertheless, the best personalized visualization recommendation performance is obtained when both $\mathbf{D}$ and $\mathbf{M}$ are



Embedding Dimension (d) \ HIT@k	1	2	3	4	5	6	7	8	9	10
1024	0.580	0.678	0.721	0.740	0.751	0.757	0.761	0.763	0.767	0.770
512	0.669	0.825	0.880	0.903	0.914	0.922	0.925	0.929	0.933	0.936
256	0.655	0.808	0.869	0.898	0.910	0.917	0.920	0.924	0.928	0.932
128	0.620	0.799	0.862	0.905	0.914	0.925	0.928	0.928	0.932	0.941
64	0.604	0.790	0.861	0.896	0.912	0.924	0.931	0.935	0.938	0.940
32	0.602	0.793	0.867	0.905	0.924	0.936	0.944	0.949	0.955	0.959
16	0.585	0.789	0.860	0.896	0.920	0.935	0.945	0.951	0.956	0.959
8	0.600	0.791	0.864	0.902	0.923	0.937	0.948	0.959	0.963	0.967
4	0.586	0.788	0.861	0.893	0.913	0.927	0.938	0.948	0.955	0.960
2	0.573	0.784	0.856	0.891	0.913	0.926	0.940	0.950	0.957	0.962
1	0.565	0.770	0.841	0.872	0.890	0.906	0.917	0.926	0.930	0.934

Fig. 7. Ablation study results for personalized visualization recommendation with varying embedding dimensions $d \in \{2^0, \ldots, 2^{10}\}$ and HR@k for $k = 1, \ldots, 10$. See text for discussion.



Embedding Dimension (d) \ nDCG@k	1	2	3	4	5	6	7	8	9	10
1024	0.578	0.641	0.663	0.671	0.675	0.677	0.679	0.679	0.680	0.681
512	0.667	0.766	0.794	0.804	0.808	0.811	0.812	0.813	0.814	0.815
256	0.652	0.750	0.780	0.793	0.798	0.800	0.801	0.802	0.804	0.805
128	0.617	0.731	0.762	0.781	0.785	0.789	0.790	0.790	0.791	0.793
64	0.602	0.720	0.756	0.771	0.777	0.781	0.784	0.785	0.786	0.787
32	0.599	0.721	0.758	0.775	0.782	0.786	0.789	0.791	0.792	0.793
16	0.583	0.712	0.748	0.764	0.773	0.778	0.782	0.783	0.785	0.786
8	0.596	0.719	0.755	0.772	0.780	0.785	0.789	0.792	0.794	0.795
4	0.582	0.711	0.748	0.762	0.770	0.775	0.779	0.782	0.784	0.785
2	0.570	0.704	0.741	0.756	0.765	0.769	0.774	0.777	0.779	0.780
1	0.561	0.692	0.728	0.742	0.749	0.754	0.758	0.761	0.762	0.763

Fig. 8. Ablation study results for personalized visualization recommendation with varying embedding dimensions $d \in \{2^0, \ldots, 2^{10}\}$ and nDCG@k for $k = 1, \ldots, 10$. See text for discussion.

used along with **A** and **C**. Finally, these two simpler variants still perform better than the baselines for HR@1 and NDCG@1 as shown in Table 5.

To understand the effect of the embedding size d on the performance of our personalized visualization recommendation approach, we vary the dimensionality of the embeddings d from 1 to 1024. In these experiments, we use PVisRec with the full-rank meta-feature matrix **M**. In Figure 7, we show results for the personalized visualization recommendation problem using our PVisRec approach with varying embedding dimensions (size) $d \in \{2^0, \ldots, 2^{10}\}$ and HR@K for $k = 1, \ldots, 10$. In addition, we also provide results in Figure-8 for NDCG@K for $k = 1, \ldots, 10$ while varying the embedding size $d \in \{2^0, \ldots, 2^{10}\}$. This experiment uses the original meta-feature matrix $\mathbf{M}$ and not the compressed meta-feature embedding (MFE) matrix. For both HR@K and NDCG@K, we observe in Figure 7-8 that performance typically increases as a function of the embedding dimension d. We also observe that for HR@1 and nDCG@1, the best performance is achieved when $d = 512$, which is 0.669 and 0.667, respectively. This holds for all k for both HR@K and NDCG@K as shown in Figure 7-8. Furthermore, when d becomes too large, we observe a large drop in performance, which is due to overfitting. For instance, in Figure 7, we see that when $d = 1024$ we have HR@1 of 0.580 compared to 0.669 for $d = 512$.

Table 7. Results comparing Non-personalized vs. Personalized Visualization Recommendation.

	HR					NDCG				
Model	@1	@2	@3	@4	@5	@1	@2	@3	@4	@5
Non-personalized	0.151	0.248	0.319	0.373	0.404	0.145	0.209	0.244	0.268	0.280
Personalized	**0.630**	**0.815**	**0.876**	**0.906**	**0.928**	**0.624**	**0.743**	**0.775**	**0.788**	**0.796**

6.2 Comparing Personalized vs. Non-personalized Visualization Recommendation

To answer RQ2, we compare the personalized visualization recommendation model (PVisRec) to a non-personalized ML model. More specifically, we compare the user-specific personalized visualization recommendation model (PVisRec) to a global non-personalized ML-based method that does not leverage a user-specific personalized model for each user. For fairness, we simply leverage the specific user embedding for the personalized model, and for the non-personalized model we simply derive an aggregate global embedding of a typical user, and leverage this global non-personalized model to rank the visualizations. More formally, the non-personalized ML-based approach uses a global user embedding derived as,

$$\mathbf{u}_g = \frac{1}{n}\sum_{i=1}^{n} \mathbf{U}_i \tag{48}$$

where $\mathbf{u}_g$ is called the global user embedding and represents the centroid of the user embeddings from PVisRec. Everything else remains the same as the personalized visualization recommendation approach. More formally, given a user i along with an arbitrary visualization $\mathcal{V} = (\mathbf{X}^{(k)}, C_t)$ generated from some dataset $\mathbf{X}$, we derive a score for the visualization $\mathcal{V}$ using the global user embedding $\mathbf{u}_g$ from Eq. 48 as follows:

$$\phi_g(V) = \mathbf{u}_g \mathbf{Z}_{t,:}^{\top} \prod_{\mathbf{x}_j \in \mathbf{X}^{(k)}} \mathbf{u}_g \mathbf{V}_{j,:}^{\top} \tag{49}$$

where $\mathbf{X}^{(k)}$ is a subset of attributes used in the visualization $\mathcal{V}$ from the dataset $\mathbf{X}$ (hence, $|\mathbf{X}^{(k)}| \leq |\mathbf{X}|$) and $C_t \in \mathcal{C}$ is the visual-configuration of $\mathcal{V}$. Hence, instead of leveraging user i's personalized visualization recommendation model to obtain a user personalized score for visualization $\mathcal{V}$ (that is $\phi(V) = \mathbf{U}_{i,:}\mathbf{Z}_{t,:}^{\top} \prod_{\mathbf{x}_j \in \mathbf{X}^{(k)}} \mathbf{U}_{i,:}\mathbf{V}_{j,:}^{\top}$), we replace $\mathbf{U}_{i,:}$ with the global user embedding $\mathbf{u}_g$ representing a "typical" user. Results are provided in Table 7. For both models in Table 7, we use the same experimental setup from Section 6.1. This PVisRec model is used for learning $\mathbf{U}$, then Eq. 48 is used for the non-personalized model. Notably, the non-personalized approach that uses the same global user model for all users performs significantly worse (as shown in Table 7) compared to the user-level personalized approach that leverages the appropriate learned model to personalize the ranking of visualizations with respect to the user at hand. This demonstrates the significance of learning individual models for each user that are personalized based on the users attribute/data preferences along with their visual design choice preferences **(RQ2)**.

Table 8. Space vs. Accuracy Trade-off Results using Meta-Feature Embeddings (MFE). Results for the space-efficient variants of our personalized visualization recommendation methods that use meta-feature embeddings. In particular, we set $d = 10$ and vary the MFE dimensions from $\{1, 2, 4, 8, 16\}$. See text for discussion.

		HR@K					NDCG@K				
Model	**MFE dim.**	@1	@2	@3	@4	@5	@1	@2	@3	@4	@5
PVisRec (**A**,**C**,**M** only)	1	0.284	0.413	0.480	0.512	0.529	0.282	0.364	0.398	0.412	0.418
	2	0.245	0.348	0.395	0.417	0.429	0.244	0.308	0.333	0.342	0.346
	4	0.265	0.388	0.444	0.468	0.481	0.263	0.341	0.369	0.380	0.385
	8	0.304	0.419	0.462	0.492	0.506	0.302	0.376	0.397	0.410	0.416
	16	0.294	0.404	0.452	0.471	0.483	0.292	0.362	0.386	0.395	0.399
PVisRec	1	0.467	0.589	0.641	0.667	0.681	0.464	0.542	0.569	0.580	0.585
	2	0.542	0.685	0.744	0.771	0.792	0.539	0.630	0.660	0.672	0.680
	4	0.544	0.713	0.779	0.815	0.829	0.541	0.649	0.682	0.698	0.704
	8	0.608	0.806	0.874	0.906	0.925	0.604	0.731	0.765	0.779	0.787
	16	0.616	0.794	0.865	0.896	0.916	0.613	0.726	0.762	0.776	0.784

6.3 Improving Space-Efficiency via Meta-Feature Embeddings

In this section, we investigate using a low-rank meta-feature embedding matrix to significantly improve the space-efficiency of our proposed approach. In particular, we replace the original meta-feature matrix **M** with a low-rank approximation that captures the most important and meaningful meta-feature signals in the data. In addition to significantly reducing the space requirements of PVisRec, we also investigate the performance when the low-rank meta-feature embeddings are used, and the space and accuracy trade-off as the number of meta-feature embedding dimensions varies from $\{1, 2, 4, 8, 16\}$. We set $d = 10$ and vary the dimensions of the dimensionality of the meta-feature embeddings (MFE) from $\{1, 2, 4, 8, 16\}$ across the different proposed approaches. We provide the results in Table 8 for the space-efficient variants of our personalized visualization recommendation methods that use meta-feature embeddings. Overall, we find that in nearly all cases, we find similar HR@K and NDCG@K compared to the original variants, while obtaining a significantly more compact model with orders of magnitude less space. For instance, when MFE dim. is 16, PvisRec has a HR@1 of 0.616 compared to 0.630 using the original 1006-dimensional meta-feature matrix, which uses roughly 63x more space compared to the 16-dimensional MFE variant. As an aside, since PVisRec (**A**, **C**, **D**) does not use **M**, it does not have a meta-feature embedding (MFE) variant. This implies that we can indeed significantly reduce the space requirements of our approaches by trading off only a tiny amount of accuracy **(RQ3)**.

6.4 Neural Personalized Visualization Recommendation

In this section, we study the performance of the proposed Neural Personalized Visualization Recommendation models (**RQ4**). For these experiments, we use $d = 10$ and $\alpha = 0.5$ for Neural PVisRec-CMF. All models use three layers and ReLU for the activation function. See Section 4 for further details. The results are provided in Table 9. Both neural personalized visualization recommendation models outperform the simpler and faster graph-based PVisRec approach (across both rank-based evaluation metrics and across all top-K personalized visualization recommendations). This is expected since the neural visualization models all leverage the graph-based PVisRec model in some fashion. Neural PVisRec uses the learned low-dimensional embeddings of the users, visual-configurations, attributes, and meta-features of the attributes as input into the first layer whereas Neural PVisRec-CMF also uses the learned low-dimensional embeddings, but also uses the predicted visualization scores from the PVisRec model for each user and combines these with the predicted scores from the neural component. Notably, both neural personalized visualization recommendation models outperform the simpler and faster graph-based approach. In Table 9, Neural PVisRec-CMF outperforms the simpler Neural PVisRec network. This holds for HR@K and NDCG@K, and across all top-K personalized visualization recommendations where $K \in \{1, ..., 5\}$.

Table 9. Results for the *Neural* Personalized Visualization Recommendation Models.

	HR@K					NDCG@K				
Model	@1	@2	@3	@4	@5	@1	@2	@3	@4	@5
Neural PVisRec	0.656	0.825	0.889	0.923	0.946	0.652	0.761	0.793	0.808	0.817
Neural PVisRec-CMF	0.762	0.879	0.922	0.944	0.961	0.729	0.822	0.845	0.855	0.861

6.4.1 *Nonlinear activation function.*

Neural PVisRec is flexible and can leverage any nonlinear activation functions for the fully-connected layers of our multilayer neural network architecture for personalized visualization recommendation. In Table 10, we compare three non-linear activation functions σ for learning a personalized visualization recommendation model including hyperbolic tangent (tanh) $\sigma(\mathbf{x}) = \tanh(\mathbf{x})$, sigmoid $\sigma(\mathbf{x}) = 1/(1 + \exp[-\mathbf{x}])$, and ReLU $\sigma(\mathbf{x}) = \max(0, \mathbf{x})$. The results in Table 10 show that ReLU performs best by a large margin followed by sigmoid and then tanh. ReLU likely performs well due to its ability to avoid saturation, handle sparse data and be less likely to overfit.

Table 10. Ablation study results of Neural PVisRec with different nonlinear activation functions. We report HR@1 for brevity. All results use $d = 10$ and $\alpha = 0.5$.

	nonlinear activation σ		
Model	tanh	sigmoid	ReLU
Neural PVisRec	0.615	0.624	0.656
Neural PVisRec-CMF	0.613	0.640	0.762

6.4.2 *Hidden layers.*

To understand the impact of the number of layers on the performance of the neural personalized visualization recommendation models, we vary the number of hidden layers from $L \in \{1, 2, 3, 4\}$. In Table 11, the performance increases as additional hidden layers are included, and begins to decrease at $L = 4$. The best performance is achieved with three hidden layers. This result indicates the benefit of deep learning for personalized visualization recommendation.

Table 11. Comparing performance of Neural PVisRec with different number of hidden layers.

	HR@K					NDCG@K				
# Hidden Layers	@1	@2	@3	@4	@5	@1	@2	@3	@4	@5
1	0.579	0.773	0.844	0.880	0.896	0.578	0.701	0.737	0.752	0.758
2	0.618	0.801	0.865	0.892	0.907	0.618	0.733	0.765	0.777	0.783
3	0.656	0.825	0.889	0.923	0.946	0.652	0.761	0.793	0.808	0.817
4	0.646	0.754	0.813	0.842	0.869	0.499	0.639	0.680	0.694	0.705

6.4.3 Layer size. Recall that our network structure followed a tower pattern where the layer size of each successive layer is halved. In this experiment, we investigate larger layer sizes while fixing the final output embedding size to be 8 and using 4 hidden layers. In Table 12, we observe a significant improvement in the visualization ranking when using larger layer sizes.

Table 12. Varying the layer sizes used in the deep personalized visualization recommendation model (Neural PVisRec). We vary the layer sizes used in the neural architecture tower structure by a multiple of $\{2, 4, 6\}$.

	HR@K				
Layer Sizes	@1	@2	@3	@4	@5
8-16-32-64	0.701	0.790	0.832	0.865	0.883
8-32-128-512	0.734	0.797	0.846	0.874	0.886
8-48-288-1728	0.752	0.822	0.869	0.895	0.913

6.4.4 Runtime performance. Neural PVisRec is also fast, taking on average 10.85 seconds to train using the large personalized visualization corpus from Section 5. The other neural visualization recommender is nearly as fast, as it contains only an additional step that is linear in the output embedding size. For these experiments, we used a 2017 MacBook Pro with 16GB memory and 3.1GHz Intel Core i7 processor.

7 DISCUSSION & FUTURE DIRECTIONS

Our approach can easily be integrated and leveraged by interactive visualization exploration tools like Voyager [Wongsuphasawat et al. 2015, 2017]. These tools allow the user to select or upload a dataset of interest, and then the list of attributes are typically displayed in alphabetical order to the user as shown in Figure 9 (A). In contrast, we can use the personalized attribute recommender to obtain a ranking of the attributes in the users dataset and then display them in the order most relevant to the user. As shown in Figure 9 (A), some attribute names may also not be very useful (*e.g.*, "eVars_eVar10-15"), despite being highly related to previous attributes that were of interest to the user. One can also display the actual personalized score of each attribute to the user. This would improve the user experience, making it faster for them to find interesting and relevant data attributes, especially for datasets with hundreds of thousands of potential attributes to select. The second way it can be used is when the user selects an attribute of interest as shown in Figure 9 (B), we can then use the personalized embeddings for that user to find and recommend similar attributes that may also be interesting to explore, or even attributes that they may select to use for the attribute of the y-encoding as highlighted in Figure 9 (B).

Additionally, one can use the personalized weights specific to the user from our approach and combine them with the rule-based weights from an interactive visualization exploration tool such as Voyager to obtain a better more personalized ranking of the visualizations. More specifically, we can add a hyperparameter that can be adjusted by the specific user of the system, depending on whether they want to give more weight to the personalized score compared to the rule-based score, and vice-versa. There are also side cases where the personalized visualization recommendation approach may not perform effectively, and thus, one can simply identify such cases and give less weight to the personalized visualization score. While we provide a comprehensive and systematic quantitative evaluation in Section 6, conducting a more qualitative user study is left for future work. In this work, we also sampled relevant and non-relevant visualizations for evaluation. Potential limitations of sampled metrics for recommendation were recently analyzed in [Krichene and Rendle 2020]. Future work should investigate the impact of such sampled metrics and explore other sampling strategies and metrics for the personalized visualization recommendation problem.

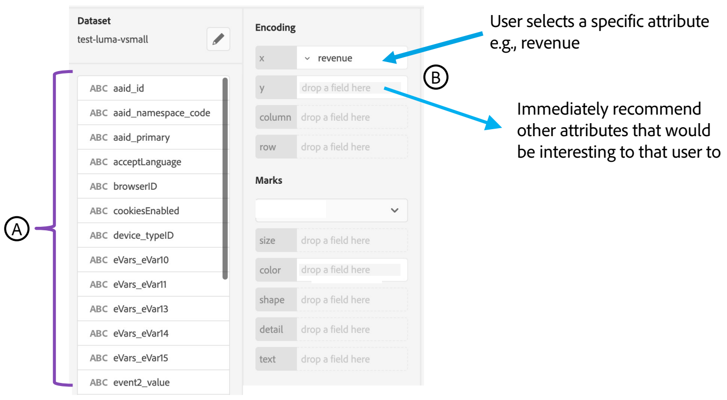


Fig. 9. Attribute recommendation use cases in visualization exploration tools like Voyager. See text for discussion.

While the proposed techniques were shown to work well in practice, there remain a few challenging cases that require additional handling when deployed in an actual system for some downstream application task. One such case is when there is a new user (and thus, no previous feedback about their preferences) and they are also interested in a new dataset that has never been seen before by any other user. Further, the data attributes in this new dataset are also not similar to any of the other thousands of datasets (and their attributes) that have been previously explored by users. In this specific extreme case, the approach would likely not perform well since there is no user feedback, and the dataset attributes of interest to them are not similar to any of the other datasets used by other users, and thus, the approach cannot even leverage the similarity between the attributes of the uploaded dataset and those previously seen by other users. To handle this case, one can simply check if the user and dataset of interest are new, and there does not exist an attribute in the users dataset that has similarity above some threshold with another attribute in one of the other datasets previously used by other users. Once this side case is detected by the system, it can then rank the visualizations using the visual rule-based approach from one of the previous works [Wongsuphasawat et al. 2017]. While this side case is unlikely, it becomes even less likely over time as the number of datasets uploaded to the system increases, and thus the proposed model becomes more familiar with other types of datasets that may have not been observed until now. As an aside, when a new user arrives with a new dataset, we simply treat all such attributes as having equal preference, since that is all the information known to us at this stage. Using this fact, we search for attributes in other datasets that are similar to those in the users new dataset, and score them accordingly. Finally, future work should investigate other side cases that may arise when such techniques are deployed in practice for a specific application task, and develop heuristics and strategies to appropriately handle them.

8 RELATED WORK

8.1 Visualization Recommendation

Rule-based visualization recommendation systems such as Voyager [Vartak et al. 2017; Wongsuphasawat et al. 2015, 2017], VizDeck [Perry et al. 2013], and DIVE [Hu et al. 2018] use a large set of rules defined manually by domain experts to recommend appropriate visualizations that satisfy the rules [Casner 1991; Derthick et al. 1997; Feiner 1985; Lee 2020; Mackinlay 1986; Mackinlay et al. 2007a; Roth et al. 1994; Seo and Shneiderman 2005; Stolte et al. 2002]. Such rule-based systems do not leverage any training data for learning or user personalization. There have been a few "hybrid" approaches that combine some form of learning with manually defined rules for visualization recommendation [Moritz et al. 2018], *e.g.*, Draco learns weights for rules (constraints) [Moritz et al. 2018]. Recently, there has been work that focused on the end-to-end ML-based visualization recommendation problem [Dibia and Demiralp 2019; Qian et al. 2020, 2021]. However, this work learns a global visualization recommendation model that is agnostic of the user, and thus not able to be used for the personalized visualization recommendation problem studied in our work.

All of the existing rule-based [Hu et al. 2018; Perry et al. 2013; Vartak et al. 2017; Wongsuphasawat et al. 2015, 2017], hybrid [Moritz et al. 2018], and pure ML-based visualization recommendation [Qian et al. 2021] approaches are unable to recommend personalized visualizations for specific users. These approaches do not model users, but focus entirely on learning or manually defining visualization rules that capture the notion of an effective visualization [Cui et al. 2019; Dang and Wilkinson 2014; Demiralp et al. 2017; Harris et al. 2021; Key et al. 2012; Kim et al. 2021; Lee et al. 2019a; Lin et al. 2020; Mackinlay et al. 2007b; Siddiqui et al. 2016; van den Elzen and van Wijk 2013; Vartak et al. 2015; Wilkinson and Wills 2008; Wills and Wilkinson 2010]. Therefore, no matter the user, the model always gives the same recommendations. The closest existing work is VizRec [Mutlu et al. 2016]. However, VizRec is only applicable when there is a single dataset shared by all users (and therefore a single small set of visualizations that the users have explicitly liked and tagged). This problem is unrealistic with many impractical assumptions that are not aligned with practice. Nevertheless, the problem solved by that prior work is a simple special case of the personalized visualization recommendation problem introduced in our paper.

Other recent work has focused on creating a large-scale benchmark dataset for visualization recommendation tasks [Hu et al. 2019b]. In particular, VizNet [Hu et al. 2019b] combines four large-scale corpora that come from web tables, online visualization tools, and open data portals. The vast majority of datasets in VizNet are from WebTables. However, this benchmark does not include crucial information such as the users, and the user-relevant datasets and visualizations (from those datasets) that the users actually preferred. Therefore, such data cannot be used for the personalized visualization recommendation problem studied in this paper. In this work, we have created a benchmark repository for this new problem, and make it publicly accessible to the wider research community to further advance research on this important and challenging problem of personalized visualization recommendation. See Section 5 for further details.

There are many important downstream applications and systems that can leverage the proposed personalized visualization recommendation techniques developed in this paper. For instance, visualization exploration systems like Voyager [Wongsuphasawat et al. 2017] use visualization recommendation algorithms to score and rank the visualizations to display for exploration, and thus they can be seen as one potential application of the actual visualization recommendation algorithms, and specifically, the personalized visualization recommendation techniques developed in our work. Deploying the personalized visualization recommendation techniques proposed in this paper in visualization exploration systems such as Voyager are left for future work as they lie outside the scope of this paper. Nevertheless, such systems have mainly used simple rule-based approaches to score and rank visualizations for exploration, and thus, applying our work on personalized visualization recommendation for this important application can bring new insights and directions for future research on improving such techniques for use in actual visualization exploration systems.

8.2 Simpler Design and Data Tasks

Besides visualization recommendation, there are methods that solve simpler sub-tasks such as improving expressiveness, improving perceptual effectiveness, matching task types, etc. These simpler sub-tasks can generally be divided two categories [Lee 2020; Wongsuphasawat et al. 2016b]: whether the solution focuses on recommending data (*what data to visualize*), such as Discovery-driven Data Cubes [Sarawagi et al. 1998], Scagnostics [Wilkinson et al. 2005], AutoVis [Wills and Wilkinson 2010], and MuVE [Ehsan et al. 2016]) or recommending encoding (*how to design and visually encode the data*), such as APT [Mackinlay 1986], ShowMe [Mackinlay et al. 2007a], and Draco–learn [Moritz et al. 2018]). While some of those are ML-based, none are able to recommend entire visualizations nor are they personalized, which is the focus of this work. For example, VizML [Hu et al. 2019a] predicts the type of a chart (e.g., bar, scatter, etc.) instead of complete visualization. Draco [Moritz et al. 2018] infers weights for a set of manually defined rules. VisPilot [Lee et al. 2019b] recommended different drill-down data subsets from datasets. As an aside, not only does these works not solve the visualization recommendation problem, they are also not personalized for individual users. Instead of solving simple sub-tasks such as predicting the chart type of a visualization, we focus on the *end-to-end personalized visualization recommendation problem (Sec. 2)*: given a dataset of interest to user i, the goal is to *automatically recommend the top-k most effective visualizations personalized for that individual user.* This paper fills the gap by proposing the first *personalized* visualization recommendation approach that is completely automatic, data-driven, and most importantly recommends personalized visualizations based on a users previous feedback, behavior, and interactions with the system.

8.3 Traditional Recommender Systems

In traditional item-based recommender systems [Adomavicius and Tuzhilin 2005; Noel et al. 2012; Ricci et al. 2011; Zhang et al. 2017; Zhao et al. 2020], there is a *single shared set of items* (*i.e.*, movies [Bennett et al. 2007; Covington et al. 2016; Harper and Konstan 2015], products [Linden et al. 2003], hashtags [Sigurbjörnsson and Van Zwol 2008; Wang et al. 2020], documents [Kanakia et al. 2019; Xu et al. 2020], news [Ge et al. 2020], books [Liu et al. 2014], and location [Bennett et al. 2011; Ye et al. 2011; Zhou et al. 2019]). However, in the personalized visualization recommendation problem studied in this work, since visualizations are dataset dependent, there is not a shared set of visualizations to recommend users. Therefore, given N datasets, there are N completely disjoint sets of visualizations that can be recommended. Every dataset consists of its own completely separate set of relevant visualizations that are exclusive to the dataset. Therefore, in contrast to the goal of traditional item recommender systems, the goal of personalized visualization recommendation is to learn a personalized vis. rec. model for each individual user, which is capable of scoring and ultimately recommending personalized visualizations to that user from any unseen dataset in the future. Some recent works have adapted various deep learning approaches for collaborative filtering [Chen et al. 2020; Guan et al. 2019; He et al. 2017; Li et al. 2020; Sedhain et al. 2015]. However, none of these works have focused on the problem of *personalized visualization recommendation* studied in this work. The personalized visualization recommender problem has a few similarities with cross-domain recommendation [Gao et al. 2013; Hu et al. 2013; Man et al. 2017; Shapira et al. 2013; Tang et al. 2012]. In cross-domain item recommendation, there is only a few datasets as opposed to tens of thousands of different datasets in our problem [Zhao et al. 2020]. More importantly, in cross-domain item recommendation, the different datasets are assumed to share at least one mode between each other, whereas in personalized visualization recommendation, each new dataset gives rise to a completely different set of visualizations to recommend.

9 CONCLUSION

In this work, we introduced the problem of user-specific personalized visualization recommendation and proposed an approach for solving it. The approach learns individual personalized visualization recommendation models for each user. In particular, the personalized vis. rec. models for each user are learned by taking into account the user feedback including both implicit and explicit feedback regarding the visual and data preferences of the users, as well as users whom have also explored similar datasets and visualizations. We overcome the issues with data sparsity and limited user feedback by leveraging the data and visualization preferences of users whom are similar, despite that the visualizations from such users are from completely different datasets. The models are able to learn better visualization recommendation models for each individual user by leveraging the data and visualization preferences of users whom are similar. In addition, we proposed a deep neural network architecture for *neural* personalized visualization recommendation that can learn complex non-linear relationships between the users, their attributes of interest, and visualization preferences. This paper is a first step in the direction of learning personalized visualization recommendation models for individual users based on their data and visualization feedback, and the data and visual preferences of users with similar data and visual preferences. Future work should investigate and develop better machine learning models and learning techniques to further improve the personalized visualization recommendation models and the visualization recommendations for individual users.

REFERENCES

Gediminas Adomavicius and Alexander Tuzhilin. 2005. Toward the next generation of recommender systems: A survey of the state-of-the-art and possible extensions. *TKDE* 17, 6 (2005), 734–749.

Thirunavukarasu Balasubramaniam, Richi Nayak, Chau Yuen, and Yu-Chu Tian. 2020. Column-wise element selection for computationally efficient nonnegative coupled matrix tensor factorization. *IEEE Transactions on Knowledge and Data Engineering* (2020).

James Bennett, Stan Lanning, et al. 2007. The netflix prize. In *KDD Cup*. 35.

Paul N Bennett, Filip Radlinski, Ryen W White, and Emine Yilmaz. 2011. Inferring and using location metadata to personalize web search. In *SIGIR*. 135–144.

Guillaume Bouchard, Dawei Yin, and Shengbo Guo. 2013. Convex collective matrix factorization. In *AISTATS*. PMLR, 144–152.

Stephen M Casner. 1991. Task-Analytic Approach to the Automated Design of Graphic Presentations. *ACM Transactions on Graphics (ToG)* 10, 2 (1991), 111–151.

Chong Chen, Min Zhang, Yongfeng Zhang, Yiqun Liu, and Shaoping Ma. 2020. Efficient neural matrix factorization without sampling for recommendation. *ACM Transactions on Information Systems (TOIS)* 38, 2 (2020), 1–28.

Dongjin Choi, Jun-Gi Jang, and U Kang. 2019. S3 CMTF: Fast, accurate, and scalable method for incomplete coupled matrix-tensor factorization. *PloS one* 14, 6 (2019), e0217316.

Paul Covington, Jay Adams, and Emre Sargin. 2016. Deep neural networks for youtube recommendations. In *RecSys*. 191–198.

Zhe Cui, Sriram Karthik Badam, M Adil Yalçin, and Niklas Elmqvist. 2019. Datasite: Proactive Visual Data Exploration With Computation of Insight-Based Recommendations. *Information Visualization* 18, 2 (2019), 251–267.

Tuan Nhon Dang and Leland Wilkinson. 2014. ScagExplorer: Exploring Scatterplots by Their Scagnostics. In *2014 IEEE Pacific visualization symposium*. IEEE, 73–80.

Çağatay Demiralp, Peter J Haas, Srinivasan Parthasarathy, and Tejaswini Pedapati. 2017. Foresight: Recommending Visual Insights. In *Proceedings of the VLDB Endowment International Conference on Very Large Data Bases*, Vol. 10.

Mark Derthick, John Kolojejchick, and Steven F Roth. 1997. An interactive visualization environment for data exploration. In *KDD*. 2–9.

Victor Dibia and Çağatay Demiralp. 2019. Data2vis: Automatic generation of data visualizations using sequence-to-sequence recurrent neural networks. *IEEE computer graphics and applications* 39, 5 (2019), 33–46.

Humaira Ehsan, Mohamed Sharaf, and Panos Chrysanthis. 2016. Muve: Efficient multi-objective view recommendation for visual data exploration. In *ICDE*.

Steven Feiner. 1985. APEX: An Experiment in the Automated Creation of Pictorial Explanations. *IEEE Computer Graphics and Applications* 5, 11 (1985), 29–37.

Sheng Gao, Hao Luo, Da Chen, Shantao Li, Patrick Gallinari, and Jun Guo. 2013. Cross-domain recommendation via cluster-level latent factor model. In *Joint European conference on machine learning and knowledge discovery in databases*. Springer, 161–176.

Suyu Ge, Chuhan Wu, Fangzhao Wu, Tao Qi, and Yongfeng Huang. 2020. Graph Enhanced Representation Learning for News Recommendation. In *WWW*.

Xinyu Guan, Zhiyong Cheng, Xiangnan He, Yongfeng Zhang, Zhibo Zhu, Qinke Peng, and Tat-Seng Chua. 2019. Attentive aspect modeling for review-aware recommendation. *ACM Transactions on Information Systems (TOIS)* 37, 3 (2019), 1–27.

F Maxwell Harper and Joseph A Konstan. 2015. The movielens datasets: History and context. *TIIS* 5, 4 (2015), 1–19.

Camille Harris, Ryan A. Rossi, Sana Malik, Jane Hoffswell, Fan Du, Tak Yeon Lee, Eunyee Koh, and Handong Zhao. 2021. Insight-centric Visualization Recommendation. *arXiv:2103.11297* (2021).

Xiangnan He, Lizi Liao, Hanwang Zhang, Liqiang Nie, Xia Hu, and Tat-Seng Chua. 2017. Neural collaborative filtering. In *WWW*. 173–182.

Xiangnan He, Hanwang Zhang, Min-Yen Kan, and Tat-Seng Chua. 2016. Fast matrix factorization for online recommendation with implicit feedback. In *Proceedings of the 39th International ACM SIGIR conference on Research and Development in Information Retrieval*. 549–558.

Kevin Hu, Michiel A Bakker, Stephen Li, Tim Kraska, and César Hidalgo. 2019a. Vizml: A machine learning approach to visualization recommendation. In *Proceedings of the 2019 CHI Conference on Human Factors in Computing Systems*. 1–12.

Kevin Hu, Snehalkumar'Neil'S Gaikwad, Madelon Hulsebos, Michiel A Bakker, Emanuel Zgraggen, César Hidalgo, Tim Kraska, Guoliang Li, Arvind Satyanarayan, and Çağatay Demiralp. 2019b. Viznet: Towards a large-scale visualization learning and benchmarking repository. In *CHI*. 1–12.

Kevin Hu, Diana Orghian, and César Hidalgo. 2018. Dive: A mixed-initiative system supporting integrated data exploration workflows. In *Workshop on Human-In-the-Loop Data Anal.* 1–7.

Liang Hu, Jian Cao, Guandong Xu, Longbing Cao, Zhiping Gu, and Can Zhu. 2013. Personalized recommendation via cross-domain triadic factorization. In *Proceedings of the 22nd International Conference on World Wide Web*. 595–606.

Anshul Kanakia, Zhihong Shen, Darrin Eide, and Kuansan Wang. 2019. A scalable hybrid research paper recommender system for microsoft academic. In *WWW*.

Alicia Key, Bill Howe, Daniel Perry, and Cecilia Aragon. 2012. VizDeck: Self-Organizing Dashboards for Visual Analytics. In *Proceedings of the 2012 ACM SIGMOD International Conference on Management of Data*. 681–684.

Hyeok Kim, Ryan Rossi, Abhraneel Sarma, Dominik Moritz, and Jessica Hullman. 2021. An Automated Approach to Reasoning About Task-Oriented Insights in Responsive Visualization. *IEEE Transactions on Visualization and Computer Graphics* 28, 1 (2021), 129–139.

Jingu Kim, Yunlong He, and Haesun Park. 2014. Algorithms for nonnegative matrix and tensor factorizations: A unified view based on block coordinate descent framework. *Journal of Global Optimization* 58, 2 (2014), 285–319.

Walid Krichene and Steffen Rendle. 2020. On sampled metrics for item recommendation. In *Proceedings of the 26th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*. 1748–1757.

Doris Jung-Lin Lee. 2020. *Insight Machines: The Past, Present, and Future of Visualization Recommendation.*

Doris Jung-Lin Lee, Himel Dev, Huizi Hu, Hazem Elmeleegy, and Aditya Parameswaran. 2019a. Avoiding Drill-Down Fallacies With VisPilot: Assisted Exploration of Data Subsets. In *Proceedings of the 24th International Conference on Intelligent User Interfaces*. 186–196.

Doris Jung-Lin Lee, Himel Dev, Huizi Hu, Hazem Elmeleegy, and Aditya Parameswaran. 2019b. Avoiding drill-down fallacies with VisPilot: assisted exploration of data subsets. In *IUI*. 186–196.

Xiangsheng Li, Maarten de Rijke, Yiqun Liu, Jiaxin Mao, Weizhi Ma, Min Zhang, and Shaoping Ma. 2020. Learning Better Representations for Neural Information Retrieval with Graph Information. In *Proceedings of the 29th ACM International Conference on Information & Knowledge Management*. 795–804.

Halden Lin, Dominik Moritz, and Jeffrey Heer. 2020. Dziban: Balancing Agency & Automation in Visualization Design via Anchored Recommendations. In *Proceedings of the 2020 CHI Conference on Human Factors in Computing Systems*. 1–12.

Greg Linden, Brent Smith, and Jeremy York. 2003. Amazon.com recommendations: Item-to-item collaborative filtering. *Internet Computing* 7, 1 (2003), 76–80.

Yidan Liu, Min Xie, and Laks VS Lakshmanan. 2014. Recommending user generated item lists. In *RecSys*. 185–192.

Jock Mackinlay. 1986. Automating the design of graphical presentations of relational information. *ACM Trans. Graph.* 5, 2 (1986), 110–141.

Jock Mackinlay, Pat Hanrahan, and Chris Stolte. 2007a. Show Me: Automatic presentation for visual analysis. *TVCG* 13, 6 (2007), 1137–1144.

Jock Mackinlay, Pat Hanrahan, and Chris Stolte. 2007b. Show Me: Automatic Presentation for Visual Analysis. *IEEE transactions on visualization and computer graphics* 13, 6 (2007), 1137–1144.

Tong Man, Huawei Shen, Xiaolong Jin, and Xueqi Cheng. 2017. Cross-Domain Recommendation: An Embedding and Mapping Approach.. In *IJCAI*. 2464–2470.

Dominik Moritz, Chenglong Wang, Greg L Nelson, Halden Lin, Adam M Smith, Bill Howe, and Jeffrey Heer. 2018. Formalizing visualization design knowledge as constraints: Actionable and extensible models in draco. *IEEE transactions on visualization and computer graphics* 25, 1 (2018), 438–448.

Belgin Mutlu, Eduardo Veas, and Christoph Trattner. 2016. Vizrec: Recommending personalized visualizations. *ACM Transactions on Interactive Intelligent Systems (TIIS)* 6, 4 (2016), 1–39.

Joseph Noel, Scott Sanner, Khoi-Nguyen Tran, Peter Christen, Lexing Xie, Edwin V Bonilla, Ehsan Abbasnejad, and Nicolás Della Penna. 2012. New objective functions for social collaborative filtering. In *Proceedings of the 21st International Conference on World Wide Web*. 859–868.

Jinoh Oh, Wook-Shin Han, Hwanjo Yu, and Xiaoqian Jiang. 2015. Fast and robust parallel SGD matrix factorization. In *SIGKDD*. ACM, 865–874.

Daniel B Perry, Bill Howe, Alicia MF Key, and Cecilia Aragon. 2013. VizDeck: Streamlining exploratory visual analytics of scientific data. (2013).

Plotly. 2018. Plotly Community Feed. http://plot.ly/feed.

Xin Qian, Ryan A. Rossi, Fan Du, Sungchul Kim, Eunyee Koh, Sana Malik, Tak Yeon Lee, and Joel Chan. 2020. ML-based Visualization Recommendation: Learning to Recommend Visualizations from Data. *arXiv:2009.12316* (2020).

Xin Qian, Ryan A Rossi, Fan Du, Sungchul Kim, Eunyee Koh, Sana Malik, Tak Yeon Lee, and Joel Chan. 2021. Learning to Recommend Visualizations from Data. In *Proceedings of the 27th ACM SIGKDD Conference on Knowledge Discovery & Data Mining*. 1359–1369.

Francesco Ricci, Lior Rokach, and Bracha Shapira. 2011. Introduction to recommender systems handbook. In *Rec. Sys. handbook*. 1–35.

Ryan A. Rossi and Rong Zhou. 2016. Parallel collective factorization for modeling large heterogeneous networks. *Social Network Analysis and Mining* 6, 1 (2016), 1–30.

Steven F Roth, John Kolojejchick, Joe Mattis, and Jade Goldstein. 1994. Interactive graphic design using automatic presentation knowledge. In *CHI*. 112–117.

Sunita Sarawagi, Rakesh Agrawal, and Nimrod Megiddo. 1998. Discovery-driven exploration of OLAP data cubes. In *Extending Database Tech.* 168–182.

Arvind Satyanarayan, Dominik Moritz, Kanit Wongsuphasawat, and Jeffrey Heer. 2016. Vega-lite: A grammar of interactive graphics. *IEEE Transactions on Visualization and Computer Graphics* 23, 1 (2016), 341–350.

Carla Schenker, Jeremy E Cohen, and Evrim Acar. 2021. An optimization framework for regularized linearly coupled matrix-tensor factorization. In *28th European Signal Processing Conference (EUSIPCO)*. 985–989.

Suvash Sedhain, Aditya Krishna Menon, Scott Sanner, and Lexing Xie. 2015. Autorec: Autoencoders meet collaborative filtering. In *Proceedings of the 24th International Conference on World Wide Web*. 111–112.

Jinwook Seo and Ben Shneiderman. 2005. A Rank-by-Feature Framework for Interactive Exploration of Multidimensional Data. *Information visualization* 4, 2 (2005), 96–113.

Bracha Shapira, Lior Rokach, and Shirley Freilikhman. 2013. Facebook single and cross domain data for recommendation systems. *User Modeling and User-Adapted Interaction* 23, 2-3 (2013), 211–247.

Tarique Siddiqui, Albert Kim, John Lee, Karrie Karahalios, and Aditya Parameswaran. 2016. Effortless Data Exploration With zenvisage: An Expressive and Interactive Visual Analytics System. *arXiv preprint arXiv:1604.03583* (2016).

Börkur Sigurbjörnsson and Roelof Van Zwol. 2008. Flickr tag recommendation based on collective knowledge. In *WWW*. 327–336.

Ajit P Singh and Geoffrey J Gordon. 2008. Relational learning via collective matrix factorization. In *KDD*. 650–658.

Chris Stolte, Diane Tang, and Pat Hanrahan. 2002. Polaris: A system for query, analysis, and visualization of multidimensional relational databases. *TVCG* 8, 1 (2002), 52–65.

Jie Tang, Sen Wu, Jimeng Sun, and Hang Su. 2012. Cross-domain collaboration recommendation. In *Proceedings of the 18th ACM SIGKDD international conference on Knowledge discovery and data mining*. 1285–1293.

Stef van den Elzen and Jarke J van Wijk. 2013. Small multiples, large singles: A new approach for visual data exploration. In *Computer Graphics Forum*, Vol. 32. Wiley Online Library, 191–200.

Manasi Vartak, Silu Huang, Tarique Siddiqui, Samuel Madden, and Aditya Parameswaran. 2017. Towards visualization recommendation systems. *ACM SIGMOD* 45, 4 (2017), 34–39.

Manasi Vartak, Sajjadur Rahman, Samuel Madden, Aditya Parameswaran, and Neoklis Polyzotis. 2015. Seedb: Efficient data-driven visualization recommendations to support visual analytics. In *Proceedings of the VLDB Endowment International Conference on Very Large Data Bases*, Vol. 8. NIH Public Access, 2182.

Xueting Wang, Yiwei Zhang, and Toshihiko Yamasaki. 2020. Earn More Social Attention: User Popularity Based Tag Recommendation System. In *WWW*.

Leland Wilkinson, Anushka Anand, and Robert Grossman. 2005. Graph-theoretic scagnostics. In *IEEE Symposium on Information Visualization*. 157–164.

Leland Wilkinson and Graham Wills. 2008. Scagnostics Distributions. *Journal of Computational and Graphical Statistics* 17, 2 (2008), 473–491.

Graham Wills and Leland Wilkinson. 2010. Autovis: automatic visualization. *Information Visualization* 9, 1 (2010), 47–69.

Kanit Wongsuphasawat, Dominik Moritz, Anushka Anand, Jock Mackinlay, Bill Howe, and Jeffrey Heer. 2015. Voyager: Exploratory analysis via faceted browsing of visualization recommendations. *IEEE transactions on visualization and computer graphics* 22, 1 (2015), 649–658.

Kanit Wongsuphasawat, Dominik Moritz, Anushka Anand, Jock Mackinlay, Bill Howe, and Jeffrey Heer. 2016a. Towards a General-Purpose Query Language for Visualization Recommendation. In *Proceedings of the Workshop on Human-In-the-Loop Data Analytics*. ACM, 4.

Kanit Wongsuphasawat, Dominik Moritz, Anushka Anand, Jock Mackinlay, Bill Howe, and Jeffrey Heer. 2016b. Towards a general-purpose query language for visualization recommendation. In *Workshop on Human-In-the-Loop Data Anal.*

Kanit Wongsuphasawat, Zening Qu, Dominik Moritz, Riley Chang, Felix Ouk, Anushka Anand, Jock Mackinlay, Bill Howe, and Jeffrey Heer. 2017. Voyager 2: Augmenting visual analysis with partial view specifications. In *Proceedings of the 2017 CHI Conference on Human Factors in Computing Systems*. 2648–2659.

Xuhai Xu, Ahmed Hassan Awadallah, Susan T. Dumais, Farheen Omar, Bogdan Popp, Robert Rounthwaite, and Farnaz Jahanbakhsh. 2020. Understanding User Behavior For Document Recommendation. In *WWW*. 3012–3018.

Mao Ye, Peifeng Yin, Wang-Chien Lee, and Dik-Lun Lee. 2011. Exploiting Geo. Influence for Collaborative Point-of-Interest Recommendation. In *SIGIR*.

Hyokun Yun, Hsiang-Fu Yu, Cho-Jui Hsieh, SVN Vishwanathan, and Inderjit Dhillon. 2014. NOMAD: Non-locking, stOchastic Multi-machine algorithm for Asynchronous and Decentralized matrix completion. *VLDB* 7, 11 (2014), 975–986.

Yongfeng Zhang, Qingyao Ai, Xu Chen, and W Bruce Croft. 2017. Joint representation learning for top-n recommendation with heterogeneous information sources. In *Proceedings of the 2017 ACM on Conference on Information and Knowledge Management*. 1449–1458.

Cheng Zhao, Chenliang Li, Rong Xiao, Hongbo Deng, and Aixin Sun. 2020. CATN: Cross-domain recommendation for cold-start users via aspect transfer network. In *SIGIR*. 229–238.

Fan Zhou, Ruiyang Yin, Kunpeng Zhang, Goce Trajcevski, Ting Zhong, and Jin Wu. 2019. Adversarial point-of-interest recommendation. In *WWW*. 3462–34618.

APPENDIX

A EXPERIMENTS FOR OTHER RELATED PERSONALIZED TASKS

In this section, we present some additional results using a few of the proposed approaches for personalized visual-configuration recommendation and personalized attribute recommendation. As an aside, the task of personalized visual-configuration recommendation is a more general and useful task compared to the design choice prediction problem focused on in other recent works [Hu et al. 2019a] since to solve the visual-configuration recommendation task, our approach predicts all such design choices at once, as opposed to scoring/predicting a single design choice like chart-type. To investigate the effectiveness of our *personalized visualization recommendation* approach for these simpler tasks, we design experiments to answer the following research questions:

- **RQ1:** How does the personalized model perform for *personalized attribute recommendation* (Section A.2)?
- **RQ2:** How does the personalized model perform for *personalized visualization-configuration recommendation* (Section A.3)?

A.1 Experimental Setup

For each of the applications, we randomly hold-out 5% of the nonzero values that correspond to observed preferences. As an example, in the case of personalized visualization configuration recommendation, 5% of the nonzero values in the user-by-configuration preference matrix **C** are held-out uniformly at randomly for evaluation of the learned models. We repeat this 10 times and average the results. Each sample corresponds to a train and test split. This is performed for both the attribute and configuration problems.

A.2 Personalized Attribute Recommendation Results

For this application, we use the same experimental setup discussed in Section A.1 unless otherwise mentioned. The ranking is on the specific dataset where an attribute appears in. For attribute recommendation, we compare to a random baseline that selects an attribute uniformly at random from the test dataset. The probability of correctly recommending the attribute in the test dataset j for user i is $\frac{1}{|\mathbf{X}_{ij}|}$ where $|\mathbf{X}_{ij}|$ denotes the number of attributes (columns) in the data matrix $\mathbf{X}_{ij}$. For *personalized* attribute recommendation, it is not as straightforward to directly apply HR@K due to the unique data characteristics around this novel problem and its formalization. For instance, every user has their own datasets with a completely different number of attributes. Furthermore, every user has their own set of attributes that are *not* shared by any other user. Hence, the set of attributes for any user is completely disjoint with the set of attributes used by any other user. This makes it difficult to report average HR@k over all users, since each user has a different number of attributes. The problem arises for large k as every user has a different upper bound on the HR@k and difficulty. Therefore, we compute HR@K for attributes whose datasets have at least five unused attributes. This ensures that we can interpret hit rate up to rank five where each model variant has a sufficient number of attributes to rank over before achieving a score of 1 at a lower rank. Results

Table 13. Personalized Attribute Recommendation Results (HR@k).

Model	HR@1	@2	@3
Random	0.089	0.168	0.262
PVisRec (**A**, **M**)	0.223	0.289	0.343
PVisRec (**A**, **M**, **C**)	0.263	0.322	0.365
PVisRec	**0.275**	**0.355**	**0.408**

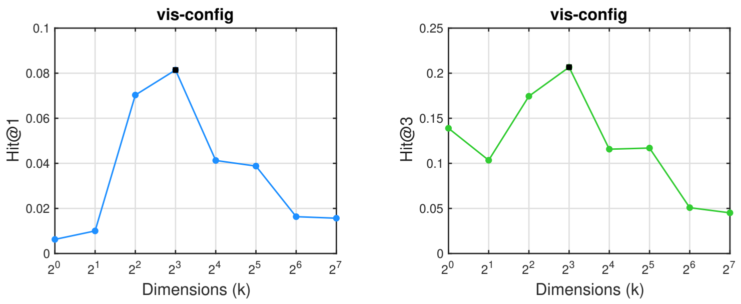


Fig. 10. Results for personalized visualization configuration recommendation with varying embedding dimensions $k \in \{2^0, \ldots, 2^7\}$. See text for discussion.

are provided in Table 13. Overall, PVisRec consistently achieves the best HR@K across all K as shown in Table 13. Comparing the PVisRec variants, we see that PVisRec always outperforms the other variants, followed by PVisRec (**A**, **M**, **C**), which performs better than PVisRec (**A**, **M**). These findings indicate that our approach is able to recommend highly relevant personalized attributes to users.

A.3 Personalized Visualization Configuration Recommendation Results

For personalized visualization configuration recommendation, since all users share the same set of possible visualization-configurations (sets of design choices), we can simply use the classic HR@k evaluation metric. In this case, we report HR@k where $k \in \{1, 2, ...5, 10, 15, 20\}$. Results are provided in Table 14. Overall, PVisRec outperforms all other methods. Notice that PVisRec (**A**, **C**) performs better than PVisRec (**A**, **M**, **C**) for $K \geq 10$. This indicates that for larger K, using the additional meta-feature matrix **M** for training can lead to a worse model. This is reasonable as **M** is not as important for this specific personalized recommendation task, and may actually degrade performance. These results indicate the effectiveness of the proposed approach for personalized visual-configuration recommendation.

Table 14. Personalized Visualization Configuration Recommendation Results (HR@k).

Model	HR@1	@2	@5	@10	@15	@20
Random	0.000	0.000	0.000	0.000	0.000	0.050
PVisRec (**C** only)	0.050	0.050	0.050	0.050	0.050	0.050
PVisRec (**A**, **C**)	0.000	0.050	**0.100**	0.200	0.250	0.250
PVisRec (**A**, **M**, **C**)	0.000	0.050	**0.100**	0.100	0.150	0.200
PVisRec	**0.100**	**0.100**	**0.100**	**0.250**	**0.350**	**0.450**

A.4 Ablation Study for Personalized Visual-Config. and Attribute Recommendation

To understand the effect of the number of embedding dimensions k on recommendations, we vary the number of dimensions $k \in \{2^0, 2^1, 2^2, 2^3, 2^4, 2^5, 2^6, 2^7\}$. Results are provided for both personalized visualization-configuration recommendation as well as personalized attribute recommendation. In these experiments, we show results using the full model that also includes the attribute by visual-configuration matrix **D**.

A.4.1 Personalized Visual-Configuration Recommendation. In Figure 10, we show results for personalized visual-configuration recommendation using different embedding dimensions $k \in \{2^0, 2^1, \ldots, 2^7\}$.

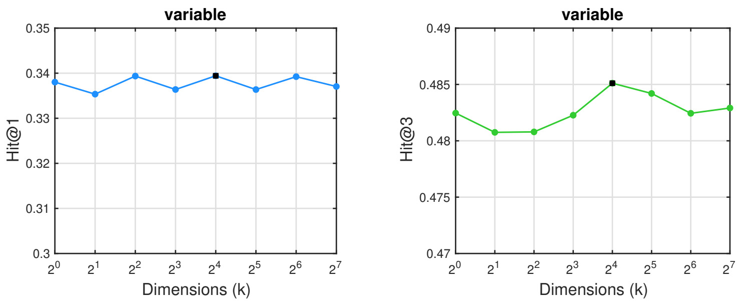


Fig. 11. Results for personalized variable recommendation with varying embedding dimensions $k \in \{2^0, \ldots, 2^7\}$. See text for discussion.

For both Hit@1 and Hit@3, we find the best performance when using a modest $k = 8$ dimensional embeddings. We expect that a small k performs best due to the sparsity and difficulty of this task.

A.4.2 Personalized Attribute Recommendation. In Figure 11, we show results for personalized attribute recommendation using different embedding dimensions $k \in \{2^0, 2^1, \ldots, 2^7\}$. For both Hit@1 and Hit@3, we find the best performance when using a modest $k = 16$ dimensional embeddings.

A.5 Results on Synthetic Data

In this section, we validate the properties and advantages of the framework on intuitive synthetic toy examples where everything can be easily controlled. This enables us to quantitatively evaluate the framework. We investigate the approach using synthetic data with known ground-truth. The synthetic data is shown in Figure 12. To quantitatively evaluate the framework, we hold-out an actual visual-configuration that a user preferred, then use the personalized model to predict it. In particular, given a user and their trained model, we apply it to score all visual-configurations, and then derive a ranking from the user-specific personalized scores, and evaluate whether the actual ground-truth visual-configuration that was held-out for evaluation is recovered as the top-1 ranked visual-configuration for that user. In this example, we hold-out one of the visual-configurations for the third user, which is shown in Figure 12 as the red dotted edge connecting the third user to the held-out visual-configuration. The user is also connected to two data-attributes for which positive user-feedback was observed (that is, they were used in a visualization generated by the user). The results are shown at the bottom in Figure 12. On the left, we see the weights for each of the visual-configurations for the third user. Notably, the proposed model assigns the largest weight to the held-out visual-configuration that was actually preferred by that user. Furthermore, the second visual-configuration is assigned the next largest weight. This makes sense since the third user shares a visual-configuration with the second user, whom also prefers the 2nd and 4th visual-configuration. Intuitively, without the meta-features of the data-attributes, the personalized model for user 3 scores these visual-configurations to be roughly equal weight, since user 2 is connected to user 1 and 3 in identical ways. However, the joint model trained with meta-features of the data-attributes, gives a larger weight to the 4th visual-configuration (which is the actual held-out visual-configuration) due to the similar meta-features across the different datasets used by the two users.

In Figure 12 (bottom right), we use the model to infer personalized scores for every data-attribute across all datasets. As expected, the actual attributes preferred by the third user are assigned

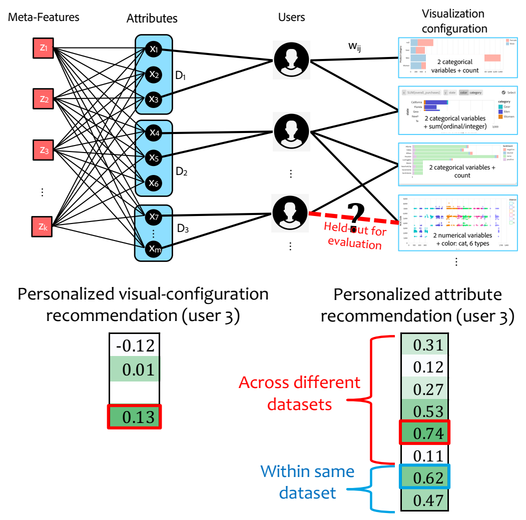

Meta-Features
Attributes
Users
Visualization configuration
w_{ij}
D_1
D_2
D_3
2 categorical variables + count
2 categorical variables + sum(ordinal/integer)
2 categorical variables + count
2 numerical variables + color: cat, 6 types
Held-out for evaluation
Personalized visual-configuration recommendation (user 3)
-0.12
0.01
0.13
Personalized attribute recommendation (user 3)
Across different datasets
Within same dataset
0.31
0.12
0.27
0.53
0.74
0.11
0.62
0.47

Fig. 12. Results on a synthetic toy dataset with known ground-truth. See text for detailed discussion.

relatively large weights (these are the data-attributes that the third user interacted with). For the across-dataset personalized attribute recommendation task that seeks to recommend individual users data-attributes that are across different datasets. For instance, in this synthetic example, there are two other datasets that the third user has not interacted with or explored, $D_1 = \{\mathbf{x}_1, \mathbf{x}_2, \mathbf{x}_3\}$ and $D_2 = \{\mathbf{x}_4, \mathbf{x}_5, \mathbf{x}_6\}$. However, the personalized model can still infer scores for the attributes across different datasets (not seen by the user). As expected, the data attributes in dataset D_1 are assigned lower scores overall compared to data-attributes in dataset D_2. Intuitively, $\{\mathbf{x}_4, \mathbf{x}_5\} \in D_2$ are closer in the graph (fewer hops) to user 3, and thus, receive larger weights (which is also moderated by the similarity of the meta-features). We also observe that attributes $\mathbf{x}_2 \in D_1$ and $\mathbf{x}_6 \in D_2$ are assigned the lowest scores. This is due to the attributes not being used/preferred by any of the users. Interestingly, the attribute $\mathbf{x}_2$ in dataset D_1 is assigned a lower score, which also makes sense since it is further from user 3 in the graph.