Frontiers of Information Technology & Electronic Engineering
www.jzus.zju.edu.cn; engineering.cae.cn; www.springerlink.com
ISSN 2095-9184 (print); ISSN 2095-9230 (online)
E-mail: jzus@zju.edu.cn

Explainable data transformation recommendation for automatic visualization*

Ziliang WU†1, Wei CHEN†‡1, Yuxin MA2, Tong XU1, Fan YAN1,
Lei LV1, Zhonghao QIAN1, Jiazhi XIA3

[1]*State Key Lab of CAD & CG, Zhejiang University, Hangzhou 310058, China*

[2]*Department of Computer Science and Engineering, Southern University of Science and Technology, Shenzhen 518055, China*

[3]*School of Computer Science and Engineering, Central South University, Changsha 410083, China*

†E-mail: wzlzju@zju.edu.cn; chenvis@zju.edu.cn

Received Sept. 23, 2022; Revision accepted Dec. 12, 2022; Crosschecked Dec. 13, 2022; Published online Dec. 23, 2022

Abstract: Automatic visualization generates meaningful visualizations to support data analysis and pattern finding for novice or casual users who are not familiar with visualization design. Current automatic visualization approaches adopt mainly aggregation and filtering to extract patterns from the original data. However, these limited data transformations fail to capture complex patterns such as clusters and correlations. Although recent advances in feature engineering provide the potential for more kinds of automatic data transformations, the auto-generated transformations lack explainability concerning how patterns are connected with the original features. To tackle these challenges, we propose a novel explainable recommendation approach for extended kinds of data transformations in automatic visualization. We summarize the space of feasible data transformations and measures on explainability of transformation operations with a literature review and a pilot study, respectively. A recommendation algorithm is designed to compute optimal transformations, which can reveal specified types of patterns and maintain explainability. We demonstrate the effectiveness of our approach through two cases and a user study.

Key words: Data transformation; Data transformation recommendation; Automatic visualization; Explainability
https://doi.org/10.1631/FITEE.2200409 **CLC number:** TP391

1 Introduction

Automatic visualization, also known as visualization recommendation, has been brought into the limelight in recent years. Existing works on automatic visualization focus mainly on data domain, chart type, encoding, and simple data transformation (e.g., filtering and aggregation). Tools such as Draco (Moritz et al., 2019), Voyager (Wongsuphasawat et al., 2016, 2017), and Dziban (Lin et al., 2020) generate attractive visualizations based on the user's requirements. However, they support only simple data formats and common visualization requirements, and patterns in complex datasets (e.g., high-dimensional data) can hardly be discovered directly without targeted feature transformations. To address this problem, data mining researchers proposed feature engineering (Zöller and Huber, 2021) techniques that derive optimal transformations under certain metrics, allowing advanced mining tasks such as clustering and outlier detection. Inspired by the successful application of feature engineering techniques, we believe that by introducing automatic data transformation in visualization recommendation, insights that used to be imperceptible can be

‡ Corresponding author

* Project supported by the National Natural Science Foundation of China (No. 62132017) and the Fundamental Research Funds for the Central Universities, China (No. 226202200235)

ORCID: Ziliang WU, https://orcid.org/0000-0002-2548-3788; Wei CHEN, https://orcid.org/0000-0002-8365-4741

discovered more easily.

In this study, we focus on data transformation of tabular data. Through a comprehensive survey of the past IEEE Visualization Conference (VIS) papers, we noticed that all data transformations can be categorized into three types based on their application scenarios, i.e., data preparation (DP), data mining (DM), and data wrangling (DW). Among them, dimension reduction and clustering are the most common transformations used for identifying insights.

To achieve data transformation recommendation, two main schemes are identified. First, we expand the design space of existing automatic visualization tools to support data transformation. Second, we leverage feature engineering approaches to obtain representative data attributes. However, both schemes suffer from poor explainability. The explainability problem first arises in the domain of machine learning, including explaining the model input, output, training process, and feature performance (Burkart and Huber, 2021). For data transformation in automatic visualization, explainability is equally significant. It requires users to relate the visualization result to the patterns of the original data.

Explainable data transformation recommendation faces three major challenges: (C1) This topic is barely studied, resulting in a lack of prior knowledge about the factors that may influence explainability and a lack of quantified models to handle these factors. (C2) It is difficult to find an explainable transformation efficiently, which requires us to design a specific searching method. (C3) Data transformation is task- and domain-related. To make the approach adaptable to different applications, internal modules of the approach should be reusable and configurable.

In this study, we first conduct a pilot study about data transformation explainability. Based on the results of the study, we select the three most important factors that cause difficulty in understanding data transformations, namely, over-transformation, dimension matching, and semantic information. We propose the concept of comprehension load to quantify the three factors (C1). All load metrics can be computed for each continuous transformation sequence. Because the loads are difficult to normalize, we leverage the Pareto-optimal concept to synthesize and compare the three metrics in one consistent framework from a multi-objective optimization perspective. To realize optimization and complete explainable searching, we introduce the Pareto-optimal pruning method, which modifies the subtree pruning in G-Skyline methods by keeping and scoring only a Pareto-optimal explainable set of transformations (C2). Particularly, our searching method is built as a flexible framework, in which most of the components are replaceable, making the framework applicable for diverse scenarios (C3).

To evaluate our method, we implement a preliminary prototype system. We present two use cases to demonstrate how our approach facilitates pattern discovery in automatic visualization. Recommendation results of our approach are compared with visualizations provided by experts. We also conduct a user study to evaluate the efficiency and effectiveness of our approach.

The contributions of this paper include: (1) a survey that summarizes existing transformation methods and a pilot study that reveals important explainability factors; (2) three comprehension load metrics, a synthesizing scheme, and a Pareto-optimal explainable pruning searching method for transformation recommendation; (3) a prototype transformation recommendation system to validate and evaluate our approach.

2 Related works

2.1 Automatic data transformation

Data transformation appears in many domains. In this study, we discuss automatic data transformation in databases, data mining, and visualization research.

In database research, data transformation aims at data preparation (data wrangling) and is about string reforming and table transforming (He et al., 2018a, 2018b), the latter of which relates to our work. To automate transformation, recent research focuses on examples and patterns from users and data, respectively. For transforming by examples, approaches proposed by He et al. (2018a) match user-created input–output pairs to several (typically one) transformation functions. To generate such functions, Foofah (Jin et al., 2017) employs an efficient search approach, while transform-data-by-example (TDE) (He et al., 2018a, 2018b) searches targets in a rich expert code library. BlinkFill (Singh,

2016) employs a graph structure and leverages semi-supervised learning to address scalability issues. For transforming by patterns (Jin et al., 2020), approaches infer potential transformation functions from data. A typical approach is learning patterns from existing transforming collections (Jin et al., 2020; Yan and He, 2020) and storing knowledge in models like recurrent neural networks (RNNs). Auto-Join (Zhu EK et al., 2017) employs q-gram matching to identify joinable rows in tables. In addition, branch and merge strategies can be leveraged to accelerate pattern learning (Ilyas et al., 2018). Recently, knowledge graph-based recommendation was developed for numeric data, using the matching between patterns and statistics (Natani and Watanabe, 2021).

In data mining research, automatic data transformation, also known as feature engineering (FE), serves as a part of automatic machine learning (AutoML) to augment features (Yao et al., 2018). Generally, the automatic process contains generation and selection, and is regarded as an optimization problem with definite objective functions (Zöller and Huber, 2021). Diverse models have been employed to optimize user-specified metrics, like machine learning (Kanter and Veeramachaneni, 2015; Katz et al., 2016), genetic programming (Tran et al., 2016), evolutionary algorithms (Chen BY et al., 2018), and ensemble learning (Dong et al., 2020). Also, researchers seek to learn transformation policies from numerous past records by regression algorithms (Kaul et al., 2017) and neural networks (Nargesian et al., 2017). To deal with such an exponential problem, greedy methods (Khurana et al., 2016) based on a tree-structure formulation (Lam et al., 2017) and reinforcement learning methods (Khurana et al., 2018) have been employed for acceleration.

Concerning visualization, researchers primarily transform data manually for these reasons: (1) the scope of transformation targets is significant and difficult to unify in one framework; (2) transformation also serves as an exploration stage for researchers to understand data (Lu et al., 2017). There also exist interactive data transformation tools (Ingram et al., 2010; Gleicher, 2013). Little research has focused on automatic data transformation for visualization. A preliminary effort evaluates common transformations in visualization, including ordering, sampling, and cleaning, provides usage guidelines (Wen and Zhou, 2008a), and implements a visual system to tailor visualization results automatically (Wen and Zhou, 2008b) for aesthetics and intuitiveness.

Compared to general visualization research, current automatic visual assistance for users is more fixed, direct, monotonous, and relatively simple. As a result, only a small number of transformations and visualizations (which are also the most basic ones) can be supported. Most automatic visualization tools provide grouping and aggregation as data transformation (Wongsuphasawat et al., 2016, 2017; Demiralp et al., 2017; Ding et al., 2019; Lin et al., 2020). DataSite (Cui et al., 2019) provides more diverse transformations, such as K-means and DBSCAN (density-based spatial clustering of applications with noise), as timely assistance for users' visual analysis. We leverage transformations and automation approaches in the above three domains to extend supported transformation categories and numbers in automatic visualization.

2.2 Automatic visualization

Automatic visualization seeks to assist common users in creating high-quality visualization without learning professional knowledge (Zhu SJ et al., 2020). Generally, recommendation approaches involve two stages: enumerating and ranking (Zeng et al., 2022). In the enumerating stage, approaches collect all possible visualizations in the enumeration space, which is significant and contains many meaningless visualizations (Qin et al., 2020). To prune such a meaningless subspace, SeeDB (Vartak et al., 2014) and DeepEye (Luo et al., 2018; Qin et al., 2018) use user-specified constraints, Draco (Moritz et al., 2019) employs expert-provided constraints, and Voyager (Wongsuphasawat et al., 2016, 2017) employs two types of constraints. In the ranking stage, approaches evaluate candidate visualizations to recommend the top-k results. Most research does not require complete user specification for desired results (Qin et al., 2020). Both rule-based ranking (Wongsuphasawat et al., 2016, 2017; Hu et al., 2018; Qin et al., 2018) and learning-based ranking (Luo et al., 2018; Dibia and Demiralp, 2019; Hu et al., 2019; Moritz et al., 2019; Wu et al., 2022) are commonly employed; e.g., Voyager (Wongsuphasawat et al., 2016, 2017) considers perceptual effectiveness rules, Draco and other works based on it (Moritz et al., 2019; Lin et al., 2020; Shen et al.,

2021) learn soft constraints provided by human beings, KG4Vis (Li HT et al., 2022) leverages a knowledge graph, and MultiVision (Wu et al., 2022) uses learning-to-rank models to learn partial orders in a tabular dataset. There are also recommendation approaches that accept user specifications; e.g., SeeDB (Vartak et al., 2014) recommends interesting visualizations based on user-provided queries, zenvisage (Siddiqui et al., 2017) recommends visualizations based on user-desired patterns, and GenoREC (Pandey et al., 2022) develops an interactive recommendation system. Recently, end-to-end models have also been employed to recommend visualizations (Qian et al., 2021; Zhou et al., 2021), and they learn recommending strategies from numerous data and avoid adjusting internal details manually.

Our work enhances traditional automatic visualization by employing data transformations before visualizing, which uncovers insights in complex datasets. We inherit the enumerating and ranking framework from visualization recommendation and fit it into the transformation context.

2.3 Machine understanding of insights

Tabular data contain rich information and play a significant role in data analysis. Much research exists to discover such information automatically (Law et al., 2020), namely insight discovery. Many researchers have developed novel metrics to judge different categories of insight significance. Interestingness is a popular metric (Geng and Hamilton, 2006; Tang et al., 2017; Ding et al., 2019), while QuickInsights (Ding et al., 2019) uses impact to measure the significance relative to an entire dataset (Wang Y et al., 2019). Statistical values are also common metrics (Demiralp et al., 2017; Cui et al., 2019) due to their scalability. Calliope (Shi et al., 2021) uses self-information to measure the significance of data facts and synthesize a data story. ScagExplorer (Dang and Wilkinson, 2014) leverages graph scagnostics (Wilkinson et al., 2005) to score the quality of scatters. Also, learning-based models can be used to evaluate insights (Zhou et al., 2020; Du et al., 2021).

Insight evaluation is relevant to, but different from, ranking of visualization recommendations. Scores of insights depend on the nature of data and are independent of visual encoding and presentation forms. Therefore, such scores are more appropriate to evaluate data transformation results. We leverage and integrate existing achievements for our recommendation approaches.

3 Design rationale

We aim to develop a data transformation recommendation approach that addresses the complex transformation problem in automatic visualization.

3.1 Data transformation for visualization

To understand the role that data transformations play in the visualization domain, we have surveyed IEEE VIS papers from 2010 to 2021. We follow the transformation definition in Wen and Zhou (2008a) and expand it into all operations that transform the original data into new forms. We focus on tabular data, and thus skip the discussion of other data types like graph and volume data. Depending on diverse transforming characteristics, transformations are employed in various usage scenarios. For transforming purposes, we summarize the transformations in three categories (Liu SX et al., 2018): data preparation (DP), data mining (DM), and data wrangling (DW).

3.1.1 Data preparation

Transformations in this category generate preliminary attributes and are often applied to prepare raw data for further operations. Most research adopts data preparation transformations, but rarely mentions this fact explicitly. The transformations for this category are selection, deletion, ranking, numeric calculation, aggregation, and string reforming.

3.1.2 Data mining

Transformations in this category employ data mining algorithms to discover data patterns. Compared to transformations in the DP category, DM transformations employ more complex computations and extract information from data at a deeper level. According to the survey results, we can categorize DM transformations into two types: projection and categorization. Projection converts the original data dimensions to several (two in general) new projection dimensions, e.g., t-distributed stochastic neighbor embedding (t-SNE) and principal component analysis (PCA). Categorization assigns each data item

a specific label, e.g., K-means and topic modeling approaches. Statistical results of DM transformations are shown in Fig. 1, where "One time" contains transformations that appear only once, and "Other" contains transformations that are modified or unstated in the study. The results indicate that dimension reduction and clustering are the most common DM transformations.

3.1.3 Data wrangling

Transformations in this category do not create new attributes, but rather form, cleanse, and revise the data. We regard them as DW transformations. Compared to DP and DM transformations, they are not used to discover insights but aim to improve data performance and visualization. Similar to DP, DW transformations are commonly used but not described in most literature. The transformations for this category are filtering, filling, sampling, normalizing, ordering, and grouping & aggregation.

Because our recommendation goal is pattern discovery, we further reduce the transformation space. We remove string reforming and DM transformations that are rarely used. The resulting transformation space is shown in Table 1. In the space,

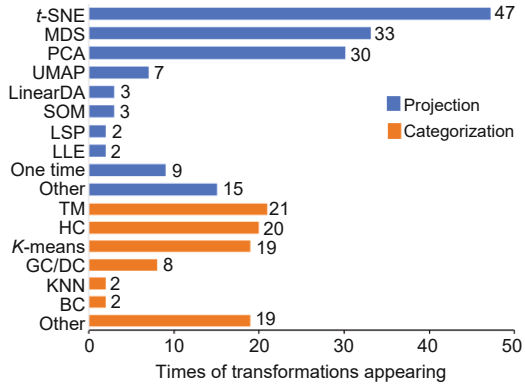

Fig. 1 Survey results of data mining (DM) transformation in IEEE Visualization Conference (VIS) papers

t-SNE: t-distributed stochastic neighbor embedding; MDS: multidimensional scaling; PCA: principal component analysis; UMAP: uniform manifold approximation and projection; LinearDA: linear discriminant analysis; SOM: self-organizing map; LSP: least square projection; LLE: locally linear embedding; TM: topic modeling; HC: hierarchical clustering; K-means: K-means clustering; GC/DC: grid- or density-based clustering; KNN: K-nearest neighbors algorithm; BC: biclustering. "One time" contains transformations that appear only once, and "Other" contains transformations that are modified or unstated in the study

DP and DM transformations influence table columns and DW transformations influence table rows. Note that AGGR in DP aggregates columns to create new attributes for a data table, which can be involved in subsequent transformations. G&A in DW groups data and aggregates rows to generate aggregation values for visualizations like a bar chart.

3.2 Problem formulation

In this subsection, we formulate the transformation recommendation problem and propose the main scheme for solving the problem.

A user-input data table contains items (rows) and attributes (columns). Given a data table D_i, a transformation T_j operates on some attribute(s) of D_i and generates new attribute(s) to derive a new table D_j (new attributes are appended to the original table). In Fig. 2, the transformations start from a source data table D_0 and are then applied progressively to subsequent tables to form a transformation tree, in which the nodes represent data tables D's and the edges represent transformations T's. A transformation path TP_k is a transformation sequence $(T_{k_1}, T_{k_2}, \ldots, T_{k_m}, T_k)$ denoting the path from root node D_0 to node D_k, representing that D_0 is transformed by m intermediate transformations (i.e., T_{k_1} to T_{k_m}) and the final transformation (i.e., T_k) to generate D_k (m can be 0). For example, in Fig. 2, TP_3 denotes the sequence (T_1, T_3), where T_1 is the intermediate transformation and T_3 is the final transformation. The tree extends until it reaches the maximum depth, and then visualization nodes (V) are added to the tree via traversal. To form a visualization, data tables must obey two constraints: (1) the data types must match channels of V; (2) the tables must have the same row transformations (e.g., in Fig. 2, if T_5 filters some data rows

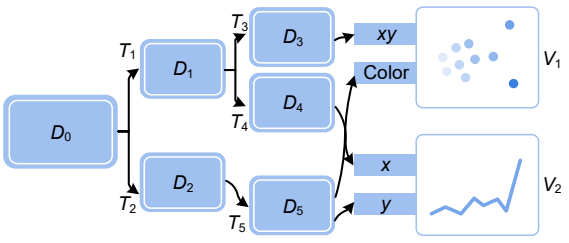

Fig. 2 A transformation tree. There are six data nodes and two visualization nodes in the tree. T_i transforms the original attributes to generate D_i. The visualization nodes select data from data nodes

Table 1 Transformations in the proposed space

Category	T	Name	Input type	Input num	Output type	Output num	Description
DP	$+-\times\div$	Numeric calculation	N	2*	N	1	*: for + and ×, operations are allowed with more than two inputs
	SEL	Selection	Any	Any	Any	Any	–
	DEL	Deletion	Any	Any	Null	Null	–
	RANK	Ranking	N	Any	N	Any	Outputting the ranking value as new columns
	AGGR	Aggregation	C/N	Any	N	Any	Aggregating columns
DM	⟨PRO⟩	Projection	N	≥ 2	N	2	Including PCA, t-SNE, MDS, and UMAP
	⟨CAT⟩	Categorization	N	≥ 2	C	1	Including K-means, DBSCAN, HC, and LDA
DW	FILT	Filtering	Any	Any	Any	Any	Filtering rows
	FILL	Filling	Any	Any	Null	Null	Filling values
	SAM	Sampling	Any	Any	Any	Any	Sampling rows
	NOR	Normalizing	Any	Any	Null	Null	Normalizing and updating values
	ORD	Ordering	Any	Any	Null	Null	Updating the order of rows
	G&A	Grouping & aggregation	Any**	Any**	Any	Any	Grouping and then aggregating rows; **: requiring at least one categorical input as the grouping flag

Data preparation (DP) and data mining (DM) transformations influence table columns, and data wrangling (DW) transformations influence rows. Column "T" lists the symbols to represent the transformations and can be used as abbreviations; input/output type stands for the input/output data type of a transformation; "input num" stands for the acceptable number of columns or rows of input data of a transformation; "output num" stands for the possible output number of columns or rows. "N" and "C" represent numerical and categorical, respectively; "any" means that the transformation has no input/output type/number constraints; "null" means that the transformation has no column/row output. PCA: principal component analysis; t-SNE: t-distributed stochastic neighbor embedding; MDS: multidimensional scaling; UMAP: uniform manifold approximation and projection; K-means: K-means clustering; DBSCAN: density-based spatial clustering of applications with noise; HC: hierarchical clustering; LDA: latent Dirichlet allocation

in D_5, it cannot match the rows of D_4 to form a visualization). The constraints mean that V nodes can be built on partial D node combinations and the transformation tree is not dense. An edge from D to V means that some channel of V comes from D. All TPs to obtain a visualization can form a transformation pipeline. For the parameters involved in transformations, a grid search can be employed. We ignore potential node duplication caused by the commutative property of some transformations, because the duplication is uncommon and has no influence on visualization nodes (Zöller and Huber, 2021).

Based on the tree-structure formulation, the recommendation can be seen as searching TPs to derive the optimal V's. As traversal space and objective are both confirmed, a natural scheme is extending the search space of general automatic visualization (Zeng et al., 2022) to cover the transformation space, then leveraging enumerating and ranking to realize searching. An alternative scheme is leveraging automatic machine learning (AutoML) approaches (Heffetz et al., 2020) by setting visualization rules as objective functions.

However, both schemes face an explainability problem. A typical transformation pipeline in the AutoML approaches is: normalization–feature selection (FS)–data enrichment (DE)/dimensionality reduction (DR). The FS and DE depend on rule-based objective functions only. Therefore, even if the pipeline can generate high-score visualizations, the result is unexplainable for users, because users cannot understand the patterns or connect the patterns to the source data. The extended automatic visualization method faces the same problem. Unexplainable examples are shown in Section 3.3.

To study the explainability, we implement a simple version of the extended searching method and use its results to conduct a pilot study.

3.3 Pilot study

3.3.1 Data and participants

We applied a simple transformation searching method (a breadth-first traversing method to find all available transformation paths) on Iris and country datasets. As searching complexity is exponential,

we limited path length to be smaller than 4. We employed the methods in Ding et al. (2019) to score insight significance including correlation, clustering, and outstandingness. Fifty results with top scores were collected and visualized for a pilot study. We recruited 24 participants including data analysts and computer science graduate students who have data analysis experience and fundamental computer science knowledge.

3.3.2 Process

The pilot study contained three stages including transformation evaluation, explainability interview, and requirement collection. First, participants observed visualizations with transformation paths and told us whether they could understand the patterns. In particular, we asked them to provide the knowledge they could learn or why it was not understandable. Second, we interviewed participants about explainability of each transformation in the T space. Finally, we asked their requirements for data transformation recommendation.

3.3.3 Results and analysis

The participants' evaluation results showed that low-explainability transformations existed widely in searching results. Two typical examples are shown in Fig. 3. The example of Iris data (Fig. 3a) showed that $(\mathrm{sl}+\mathrm{pl})\times\mathrm{pw}$ had better separation ability and correlation than $\mathrm{sl}+\mathrm{sw}$. However, participants all claimed that $(\mathrm{sl}+\mathrm{pl})\times\mathrm{pw}$ was not an explainable transformation because such an area-like quantity does not have realistic meaning. The example of country data (Figs. 3b and 3c) showed that replacing gdpp with an economy-unrelated attribute life_expec did not generate results whose significance score decreased, but the latter confused participants. Based on the pilot study, we summarize the low explainability problem as follows:

P1: The transformation path contains numerous calculations, thereby hindering users' understanding.

P2: The transformation generates a quantity by complex but meaningless calculations.

P3: Attributes that are transformed together cannot match.

Interviews in the second stage showed that the explainability of a specific transformation varied widely among users and tasks. However, all

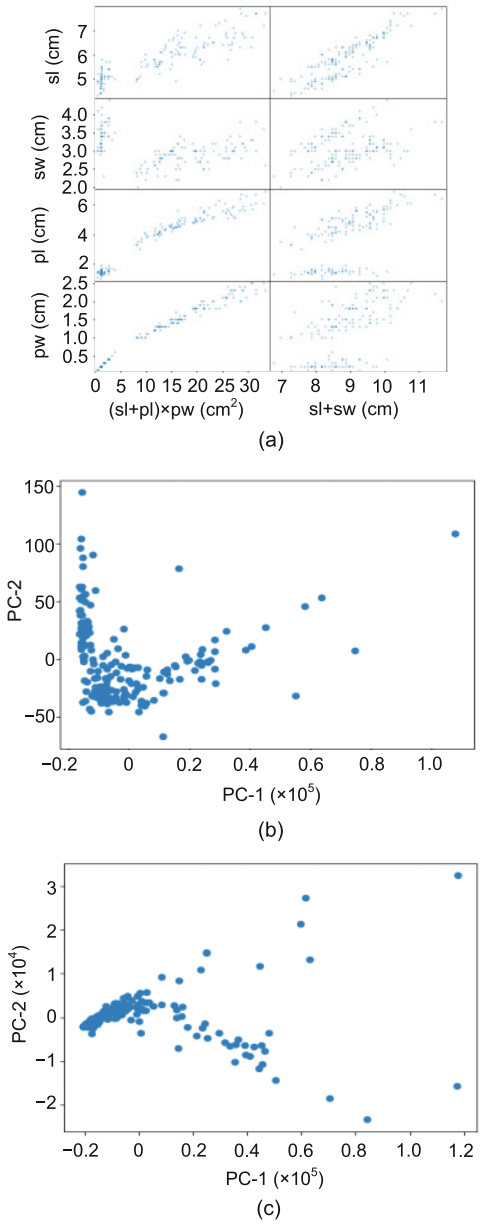

Fig. 3 Unexplainable examples in our pilot study: (a) using basic calculation transformations to study the Iris dataset; (b) principal component analysis (PCA) results with child_mort, exports, imports, income, inflation, total_fer, life_expec; (c) PCA results with child_mort, exports, imports, income, inflation, total_fer, gdpp. (a) demonstrates that a combination of basic calculations may generate unexplainable results containing striking patterns. (b) and (c) demonstrate that PCA may select country attributes related to economics as input, but exceptions also exist, e.g., including life_expec and excluding gdpp. sl: sepal length; sw: sepal width; pl: petal length; pw: petal width. PC-1 and PC-2 represent the first and second principal components, respectively

participants agreed that complex DM transformations do not mean low explainability, because the algorithms all have definite statistical meanings and enable users to connect patterns in transformed data with source data. For example, clustering in two-dimensional PCA results also implies clustering in high-dimensional data. In this way, transformation results can be regarded as the feature of all input attributes. This time, the explainability relates to whether input attributes can be understood as a whole by users.

Finally, we collected requirements supplied by participants for a transformation recommendation tool. In addition to explainability, participants expected the recommending approach to be configurable so that users can replace modules based on tasks. Recommendations need to be diverse to enable user explorations; otherwise, users can accept that the computation of recommendations as transformations is time-consuming. However, recommending still needs to be completed within acceptable time.

4 Search-based transformation recommendation

We follow the formulation in Section 3.2 and regard the recommendation as searching optimal TPs. For optimal TPs, currently, we have two goals: (G1) explainability for users, to derive understandable V's; (G2) pattern discovery, to derive insightful V's.

4.1 Explainability

The pilot study shows user difficulties in understanding unexplainable transformations. According to the pilot results, we summarize the underlying problem in three factors: over-transformation, dimension matching, and semantic information. We define three comprehension load metrics to describe the above factors.

4.1.1 Over-transformation

Over-transformation means that a series of transformations, typically numeric calculations, are applied on the original data and produce a result in which users cannot understand the intent and meaning of the transformations and cannot identify connections between the transformed and original data. This corresponds to P1 and P2 in the pilot study. According to the pilot study, users usually expect succinct but powerful transformations. Only when a specific domain meaning exists can one complex transformation path be explainable. Therefore, for our automatic recommendation, it is necessary to shorten transformation paths to improve explainability.

In the T space, calculation, ranking, and aggregation are relevant transformations. We therefore define a comprehension load metric, $\text{load}_{\text{over}}$, to represent the transforming degree that a transformation path brings to the original data. Every relevant transformation T is assigned a weight w_T. Each time an attribute is involved in the transformation, the load increases by w_T. We set a higher weight for multiplication and division according to the pilot study. Weights are adjustable to ensure that the load fits different scenarios.

Under the definition, the calculation formula of the load is

$$\text{load}_{\text{over}}(\text{TP}) = \sum_{T \in \text{TP}} \text{load}_{\text{over}}(T) = \sum_{T \in \text{TP}} w_T. \quad (1)$$

A low load represents less difficulty in explaining the TP.

4.1.2 Dimension matching

The explainability of a transformation also closely relates to the attributes on which it operates. For example, addition and subtraction can apply only on quantities with the same unit measure. For DM transformations like dimension reduction, although input dimensions are not strictly constrained, the pilot study shows that the matching of attributes influences explainability. We call this explainability component dimension matching, which corresponds to P3.

We first apply transformations on attributes with identical unit measures if explicit units are detected in table headers. Then we borrow methods from recent data management research that use statistics, including min/max, field length, and distribution, to match data attributes (Natani and Watanabe, 2021) and discover computable columns (Zhu EK et al., 2017; Golfarelli and Rizzi, 2018). Inspired by these methods, we use statistics to infer dimension matching by calculating a confidence. For two attributes i and j, we use only their

distributions as statistics because min/max and field length information is included. Their distributions are denoted as d_i and d_j, respectively, and the distribution matching degree is the distance between d_i and d_j. We choose the Wasserstein distance as the distance measure. Other distance metrics, e.g., Jensen–Shannon (JS) divergence, can be used, but the Wasserstein distance can give high-quality difference measures when distributions are supported on non-overlapping domains (Kolouri et al., 2018); e.g., for three data columns (1, 2, 3, 2, 1), (7, 8, 9, 8, 7), (100, 200, 300, 200, 100), the Wasserstein distance can recognize that the first two columns have more closed distributions, but JS divergence will give identical distribution distances between them. We regard the distance closeness as confidence, meaning that we have higher confidence to apply transformations on closer attributes. Therefore, the comprehension load metric can be defined as

$$\mathrm{load}_{\mathrm{dim}}(T)=\begin{cases}0, \text{ for unit measure matching},\\ \mathrm{Wass}(d_i,d_j), \text{ otherwise},\end{cases} \tag{2}$$

where T operates i and j, and $\mathrm{Wass}(\cdot)$ denotes the Wasserstein distance function. For transformation involving more than two attributes (attribute set A, where $|A|>2$), the load is defined as the maximum distance between them:

$$\mathrm{load}_{\mathrm{dim}}(T)=\max_{i,j\in A}\{\mathrm{Wass}(d_i,d_j)\}.$$

For a TP, its dimension-matching load is the average load of its T's:

$$\mathrm{load}_{\mathrm{dim}}(\mathrm{TP})=\frac{1}{\mathrm{length}(\mathrm{TP})}\sum_{T\in\mathrm{TP}}\mathrm{load}_{\mathrm{dim}}(T). \tag{3}$$

4.1.3 Semantic information

Semantic information is an underlying relationship among data attributes. For a given data table, we consider column names as potential semantic information, and propose to apply transformations on semantically similar attributes to improve explainability (P3). To parse natural language and obtain numerical representations of column names, word-to-vector (Mikolov et al., 2013; Wu et al., 2022) can be used as an effective mapping approach. Existing word embedding models trained on large datasets can map words to a continuous vector space in which distance represents a similarity measurement.

Using pre-trained word-to-vector models, we represent the column name as an embedding vector $\boldsymbol{e}_i$ for each attribute i. Specifically, we calculate the average embedding vector of all words in the name string as $\boldsymbol{e}_i$ (Dey et al., 2017; Fu et al., 2018). For stop words and those column names containing no available words (e.g., i, j, t_0, and Idx), we set their embedding $\mathbf{0}$ vector, because no semantic information can be detected. The data attributes newly created by transformations do not have column names, so we use their parent attributes' average embedding vector.

The comprehension load metric can be defined as the normalized cosine distance between embedding vectors of transformed attributes:

$$\mathrm{load}_{\mathrm{sem}}(T)=N(\mathrm{dist}_{\cos}(\boldsymbol{e}_i,\boldsymbol{e}_j))\in[0,1],$$

where $\mathrm{dist}_{\cos}$ is the cosine distance function and $N(\cdot)$ denotes linear normalization. Similarly, we use the maximum cosine distance as the load when transforming more than two attributes:

$$\mathrm{load}_{\mathrm{sem}}(T)=\max_{i,j\in A}\{N(\mathrm{dist}_{\cos}(\boldsymbol{e}_i,\boldsymbol{e}_j))\},$$

where A is the transformed attribute set. If some attribute embedded as a zero vector is involved in a transformation, we set the corresponding $\mathrm{load}_{\mathrm{sem}}(T)$ as the upper bound of the cosine distance, i.e., 1. As a result, two completely semantically irrelevant attributes or two attributes whose relevance cannot be detected will have the maximum load. The detected semantic relevance will facilitate understanding, thereby decreasing the load to be smaller than 1. Also, the load of a path is defined as the mean value:

$$\mathrm{load}_{\mathrm{sem}}(\mathrm{TP})=\frac{1}{\mathrm{length}(\mathrm{TP})}\sum_{T\in\mathrm{TP}}\mathrm{load}_{\mathrm{sem}}(T). \tag{4}$$

In addition to inferring semantic relevance from column name embedding, matching of explicit column names serves as available semantic information. The details are shown in Section 5.1.

4.1.4 Synthesis

The three metrics ($\mathrm{load}_{\mathrm{over}}$, $\mathrm{load}_{\mathrm{dim}}$, and $\mathrm{load}_{\mathrm{sem}}$) are calculated independently in different measurement spaces. Therefore, we need an appropriate approach to synthesize the total comprehension load. A weighted summation can be used, but the weights are difficult to confirm. An alternative

scheme is to collect considerable datasets to calculate average values of the three load components and use averages for normalization. However, even if the amount of data can be satisfied, comprehensiveness is difficult to guarantee. Specifically, data collected can reflect only a limited data domain, and the normalization fails to transform loads of unseen TPs into a consistent measurement.

We propose to address synthesis from a multi-objective perspective and leverage skyline to synthesize an explainable set (Borzsony et al., 2001; Ngatchou et al., 2005). Skyline, also known as Pareto-optimal, is a subset of data in which every data point is not dominated by others. Meanwhile, each point outside skyline is dominated by at least one other point. Point p_1 dominating p_2 means that all dimensions of p_1 are better than or equal to that of p_2, and at least one dimension is "strictly better than." Skyline considers multiple dimensions independently and provides a Pareto-optimal set, which meets our requirements.

However, our scenario causes us to adapt the original method. First, a single line of points may be insufficient. We need the resulting set to have diversity (A1) to guarantee recommendations that are insightful enough according to G2. Second, each node can be seen as an update of its ancestors and does not need to be compared with them (A2). Finally, we can develop a pruning search for acceleration (A3). Our explainable searching method is discussed in Section 4.2.2.

4.2 Explainable transformation path search

We divide searching into three parts:

1. space: TP space based on T space;

2. searching: searching method to generate an explainable TP set;

3. scoring: scoring functions to evaluate visualizations derived by the resulting TPs.

Based on the three parts, our method framework is as shown in Fig. 4. The method constructs a tree-structure search space, searches explainable transformation paths, and evaluates the resulting visualizations.

4.2.1 Space

According to the T space (Table 1), we further organize the TP space. First, the transformation survey and pilot study show that for the most part, only one DM T can appear in a TP. Therefore, we establish the limit that a TP can include at most one DM T out of explainability considerations. Second, DW T's do not generate new attributes, which allows us to consider DW T's as visualization decorating operations and attempt to apply them at the end of TPs (Wen and Zhou, 2008a). We add filtering, normalizing, and tokenizing for DM T's as internal preprocessing to separate DW T's. Consequently, DW T's can be skipped in the searching stage and added in the visualization scoring stage. These two simplifications allow us to develop a DM-centric TP space (Fig. 4). DP transformations are performed first as preparation, and DM transformations (including null) follow up to obtain core attributes. DW transformations are then performed according to the visualization type, e.g., grouping & aggregation for bar charts and outlier filtering for scatter plots. For each input channel of visualization, every DM T is checked to see whether its output type matches the input type of V (numerical/categorical). By confirming a DM T, the searching can focus on incomplete TPs consisting of DP T's. In this way, we constrain the search space in a controllable but still considerable range.

4.2.2 Searching

In this subsection, we describe our searching method to generate an explainable set. We adopt a pruning search according to A3, because the pruning frees us from exploring the entire TP space. The searching runs in the space of incomplete TPs consisting of DP T's.

Load space: We regard three load metrics as three optimizing objectives and analyze the transformation tree in the load metric space. Based on the load definition, three loads can be calculated for all nodes along with transformation tree extension. Each node of the tree is represented as a tri-tuple ($\text{load}_{\text{over}}$, load_{dim}, load_{sem}), which can be seen as three-dimensional coordinates that correspond to a point in the load metric space. The root represents untransformed source data, thereby being $(0, 0, 0)$. The tree extends in the first octant because all loads increase positively, and a two-dimensional version is shown in Fig. 5 for illustration. As formulated in Section 3.1, each point, corresponding to a D node, can represent a T and a TP. The three loads belong

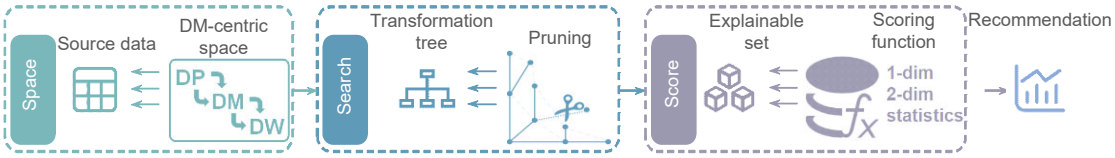

Fig. 4 Our approach consists of three parts, including searching space, pruning search, and scoring functions. A DM-centric searching space is first constructed based on the source data. As the transformation tree expands, nodes with low explainability are pruned to generate an explainable set. Insight scoring functions are applied on the set to rank candidates and decide recommendations. DP: data preparation; DM: data mining; DW: data wrangling

to the TP as well.

We first define an "easy" region in the load space (blue region in Fig. 5). In the region, no points or corresponding nodes will be pruned or compared with other points, because TPs on them are short enough and intuitive to understand. Setting the region is necessary to protect straightforward TPs from pruning. In general, the easy region is defined according to node depth. In our experiments, nodes with a depth less than 1 or 2 are appropriate.

Pruning: Because we want more diverse points than skyline (A1), a group-based skyline (G-Skyline) (Liu JF et al., 2015) can be used instead. G-Skyline provides several l-sized Pareto groups to replace a single skyline, where each group is undominated. This time, our resulting set is the union of all Pareto groups. However, to leverage G-Skyline, a set enumeration tree must be constructed, which is computation-intensive. Considering that we do not need the grouping information, we leverage layered skyline and subtree pruning in G-Skyline to develop an alternative simplified method.

Based on the transformation tree, we partition points into layers, where each layer consists of points with the same depth. We conduct subtree pruning in each layer to form a skyline layer (red dash in Fig. 5). In detail, we check all points in the layer whether they are dominated or not. Once a dominated point occurs, it and its subtree are pruned. In the tree, every node is compared only with nodes in the same layer, thereby avoiding comparison with its ancestors (A2). In Fig. 5, an example two-layer pruning is shown. The pruning continues with tree extension until it reaches the size limit l. The remaining points are used to generate an explainable set replacing the Pareto groups.

It is easy to prove that given the same l, the G-Skyline result is a subset of the simplified method

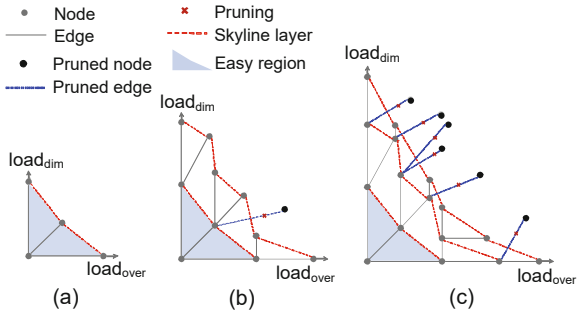

Fig. 5 The pruning process: (a) easy region; (b) layer-2; (c) layer-3. In layers out of the easy region, a skyline operator is executed to prune dominated points. The remaining points form the skyline layer and generate the next layer of points. References to color refer to the online version of this figure

result because we check only dominations in the same layer and bypass cross-layer comparison. In our scenario, it is reasonable to reserve TPs dominated by only shorter TPs because these TPs are still Pareto-optimal in their layers and can enrich the resulting diversity.

Synthesis: The resulting TP set is obtained by traversing from the root to each of the remaining points. In the set, each TP consists of DP T only. A complete transformation path defined in Section 3.2 is obtained by connecting a resulting TP in the set, a DM T, and an optional DW T by matching input and output. By feeding source data into available transformation paths of each visualization, we obtain the final V set.

4.2.3 Scoring

We provide three types of visualizations, including scatter, line chart, and bar chart. To accomplish recommendations, we leverage existing scoring methods for both visualization and data to evaluate insightfulness. Any significance metric can be used, including statistics, user-specified, expert-provided,

and learning-based metrics. Note that the scoring module is user-configurable, and that all metrics can be replaced by others.

All supported DW T's are checked concerning whether they need to be applied based on the design guidelines proposed by Wen and Zhou (2008a), where grouping and aggregation are necessary for bar charts, and the others are optional. In detail, we first check whether most scores increase by the transformation (for visual legibility and visual pattern recognizability) and then check whether the transformed data are still close to the source data in Wasserstein distance measurement (for visual fidelity).

We sort the resulting visualizations by their average scores because all scores are normalized. Top-k visualizations and their corresponding transformation paths are recommended.

5 System

We implement the transformation recommendation approach and a prototype system for verification and evaluation.

5.1 Implementation details and optimizations

We implement DM transformations by Scikit-learn and execute a preprocessing step in each transforming process, including normalization in dimension reduction (Abdi and Williams, 2010; McInnes et al., 2018) and tokenization in topic modeling (Chen SM et al., 2020).

5.1.1 Attribute grouping

To further accelerate the searching procedures and enable users to control computational time, we pre-compute some attribute groups in the distribution space and column name embedding space. In particular, we employ DBSCAN clustering in the two vector spaces to generate groups once the number of attributes exceeds N. We also leverage substring matching (SM) proposed by Warren and Tompa (2006) to acquire some explicit groups as supplements. The attribute groups are pre-extracted information of relevance between attributes. In subsequent explainable TP searching, the system will preferentially check whether it is possible to transform attributes in one group together. In this way, potential explainable groups will have high priority in search. If users set an insufficient time for a large dataset, explainable searching is still available. An online incremental DBSCAN (Chakraborty and Nagwani, 2014) is employed to group newly created attributes dynamically and efficiently. In our experiments, 20 is appropriate for N. In addition, we expose grouping information to users and enable them to revise the grouping. By doing this, users can delete unsuited groups or add expected groups.

5.1.2 Parallelization

DM T's in the projection or categorization class have similar properties and identical requirements. Therefore, we search TPs for one class together, and T's in the same class share the searching results. The two parts are independent and we begin searching processes in parallel for them. Searching processes store TPs in two pools in memory; meanwhile, a synthesis process checks pools and assembles complete transformation pipelines for visualization channels. Three processes run simultaneously to advance the scoring stage and reduce the heavy scoring time created by transformation computation. In addition, if searching and assembling cost is unacceptable for users, a dynamic presentation scheme is adopted, where early results are presented while the computation is running. In particular, once computational time reaches its limit, assembled visualizations are evaluated to generate early results and computational time is reset.

5.1.3 Scoring functions

For ease of use, we categorize metrics as one-dimensional, two-dimensional, and statistics.

One-dimensional: We use hypothetical tests to evaluate significance like salience, linearness, and correlation in line charts and bar charts according to QuickInsights (Ding et al., 2019). If the target visualization presents more than one variable, the score is the average value of all variables.

Two-dimensional: We provide two types of functions for two-dimensional data visualization (scatter plot). First, the scagnostics method (Wilkinson et al., 2005) can capture interestingness of point distribution, like outlying, skewed, and straight. Second, we provide functions to evaluate grouping quality in scatter plots, including inter-group comparison (class density measure (CDM) (Tatu et al., 2009))

and intra-group quality scores from scagnostics.

Statistics: We provide some statistical metrics from existing transformation recommendation research (Demiralp et al., 2017), such as dispersion and heavy tails, for users to constrain general properties of transformed data.

We also provide four pre-defined scoring function configurations based on categories mentioned in QuickInsights (Ding et al., 2019) for discovering distribution-wise insights, point-wise insights, compound insights, and all insights. For example, distribution-wise configuration contains all DM T's and all two-dimensional scoring functions. Users can further edit configuration based on pre-defined results.

5.2 Interface

We design and implement an experimental system for user study (Cao et al., 2020; Pan et al., 2020; Xia et al., 2020). The system interface shown in Fig. 6 contains five views: data table, attribute group, visualization navigation, transformation overview, and transformation path.

5.2.1 Data table

This view receives user-uploaded data and shows basic attribute information including name, type, domain, min, max, iskey, and values, where type and iskey are editable for users.

5.2.2 Attribute group

The system computes grouping automatically and presents results in foldable boxes. Users can add new groups and delete existing groups using the "Edit groups" button. Below the "attribute group" view, a "Configure" button can expand a parameter configuration panel that includes visualization type, transformations, scoring functions, and system parameters such as the maximum recommendation number. A "Query" button triggers a recommendation.

5.2.3 Visualization navigation

After obtaining recommendation results, the system shows a visualization preview in one list for navigation. All the visualizations are generated by a charts library ECharts (Li DQ et al., 2018) and we use the preset encoding configurations. By default,

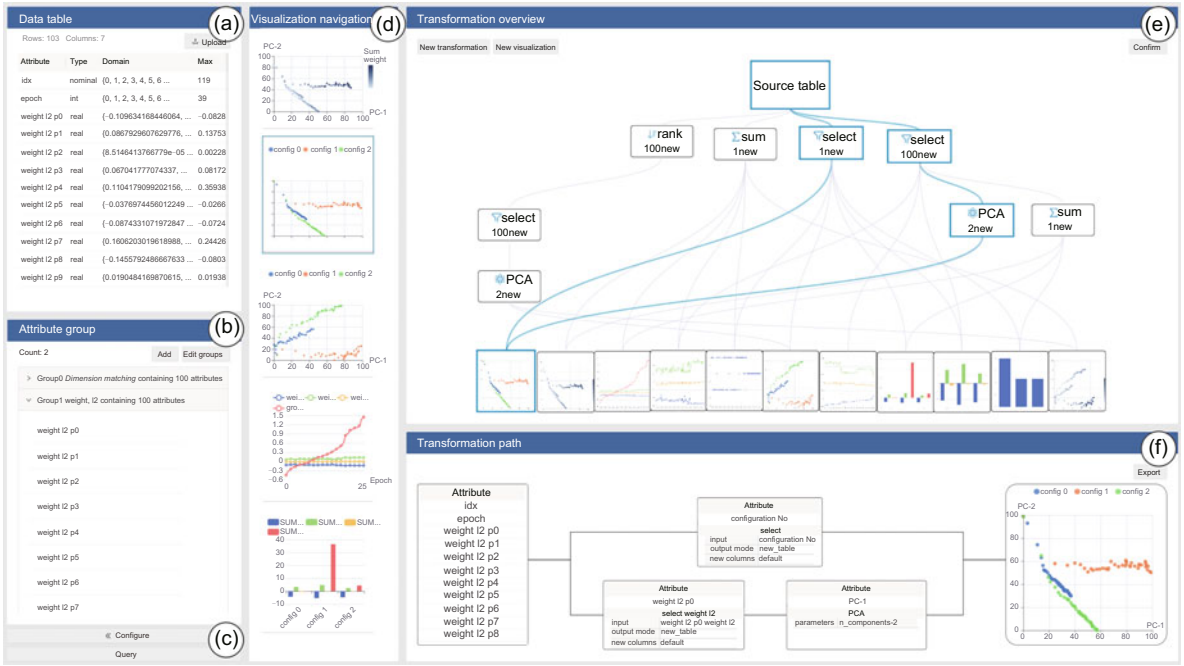

Fig. 6 Our visual interface. The data view (a) shows basic information. The attribute group view (b) presents groups automatically detected by the system and enables users to edit grouping situations. Two buttons (c) allow users to configure system parameters and trigger recommendations. A visualization navigation list (d) presents previews of recommendation. Overall transformation recommendations are presented in a tree structure with visualization thumbnails (e) and detailed information is shown below (f)

top-10 results of each chart type are shown. Operations like filtering and highlighting are available in preview charts. By double-clicking a chart, the transformation pipeline from the root node to the result is highlighted in the "transformation overview," and detailed path information is presented in the "transformation path" view. The visualization generation module is replaceable and other automatic encoding recommendations can be used.

5.2.4 Transformation overview

All nodes and edges in the transformation tree are shown. Each data node presents the transformation type, an icon, and the number of newly created attributes, and each visualization node presents a thumbnail of the result. The view supports zoom and panning. We add interactions to help users create transformation and visualization nodes independently. By clicking buttons in the top-left corner, users can call out the function.

5.2.5 Transformation path

After users select a visualization, the view shows detailed node and edge information in the pipeline. In particular, attributes contained in each node and parameters of corresponding transformation are shown.

6 Evaluation

This section describes how our approach facilitates pattern discovery in automatic visualization through two use cases on real-world datasets, a quantitative comparison with expert-designed transformations, and a user study. All experiments are conducted with an easy region depth of one and a maximum TP length of five.

6.1 Use cases

We demonstrate two usage cases of the system based on two real-world datasets.

The first dataset is about international trade. The data record basic information and trade values of 238 countries or regions. In particular, export value and import value of 10 industries form 20 attributes in the table, e.g., textiles export. To study relationships between countries, a data analyst selects distribution-wise configuration and filters out several groups. Among recommendations, the top-1 scatter plot attracts the analyst's attention as point colors vary along the horizontal axis (Fig. 7a). In the scatter plot, point coordinates are PCA projections of all export and import attributes, and the color encodes the sum of these attributes. The analyst finds that although China and the USA are both top trade powers, China has a more similar trade mode to other countries because it is closer to them. Next, the analyst expects to study the continent-wise trade situation, but the same PCA transformation is not appropriate because most of countries appear on the left. The system also provides a PCA transformation path that transforms trade values into ranks (Fig. 7b). The ranking operation homogenizes intervals between countries. Therefore, the analyst is able to compare continents definitely, e.g., Africa and Oceania (Figs. 7b1 and 7b2). Another recommendation (Fig. 7c1) indicates that the PC-1 axis of such PCA transformation relates to total export and import values. Then the analyst enables latent Dirichlet allocation (LDA) to study the trade industries in the various countries. The analyst first observes a bar chart with an outstanding No. 1 category, where the total export sum and import sum of each trade topic are shown (Fig. 7c). The bar chart indicates that topic-1 countries have significant advantages in international trade, and that topic-1 is distributed mainly on textiles, electronics, and machinery exports. The analyst further checks a scatter of trade topic categories on PCA (Fig. 7c2) and realizes that the PC-2 axis actually relates to the country trade topics of LDA.

The second dataset is collected during the training process of a machine learning model under three different configurations. Each data item consists of 100 weights and hyperparameters. The system detects a significant clustering in dimension reduction results, where each cluster is stringy. Therefore, results containing PCA and t-SNE are recommended as top results. The analyst selects recommendations containing PCA (Fig. 7d) because the analyst expects to check model weights in a linear transformation space. Results show that "config 1" is notably different from the two other configurations, because its weight gradually separates from those of the others during the training process. Fig. 7e shows a comparison of the aggregated weight sum, which illustrates that "config 1" differs from the others because

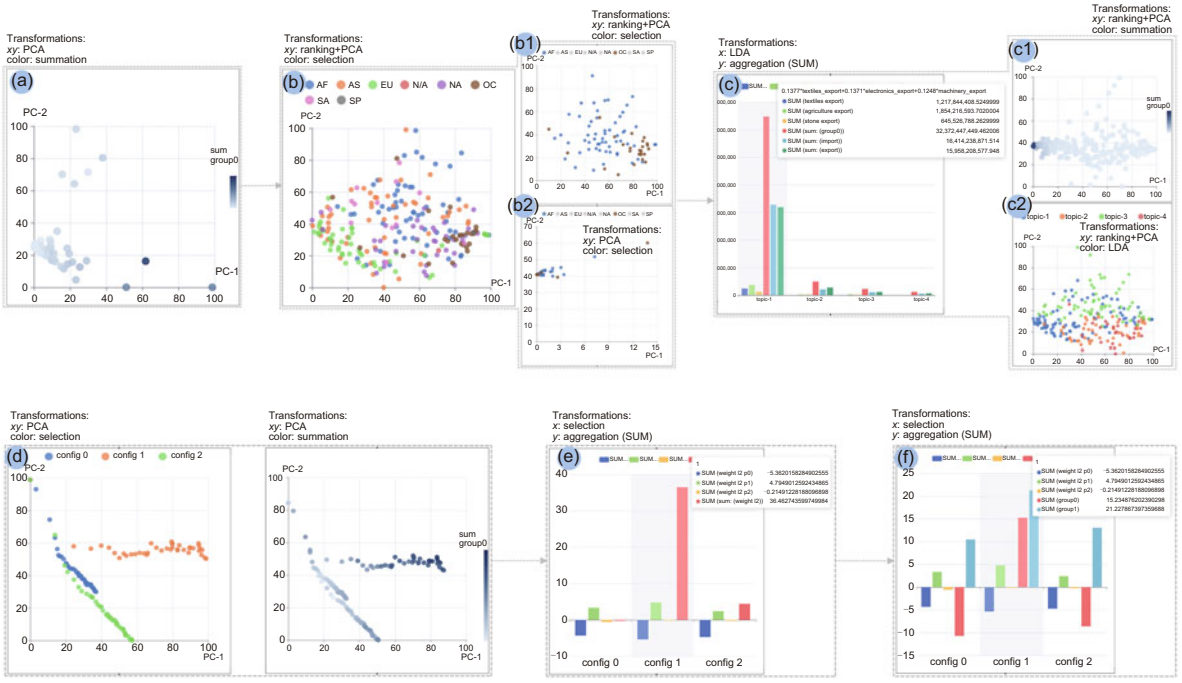

Fig. 7 Two recommendation cases: (a–c) recommended based on an international trade dataset; (d–f) generated based on a model training dataset

it results in a large weight sum. Later, the analyst notices two attribute groups clustered by DBSCAN, which are weights with low and high index numbers. The analyst then checks the corresponding results and finds an unbalanced weight distribution based on index numbers. Therefore, the analyst deletes the original groups and creates two groups containing the first and last halves of the weights, and then queries recommendation. The resulting bar chart shows weight sums of the two newly added groups, which indicates that the difference in the three configurations is affected mainly by the first half of the weights (Fig. 7f).

6.2 Evaluation of the recommendation approach

We designed experiments to evaluate the usefulness of the recommendation approach by comparison with expert-designed transformation pipelines.

We collected 24 data tables from Kaggle. The data domain includes economics, medicine, culture, vehicles, entertainment, and machine learning. The number of rows of data ranges from 24 to 6000 and that of columns ranges from 10 to 138. We also collected the data transformation pipelines presented on Kaggle that have data tables, which discover patterns in data. In this way, we collected 12 pipelines. To extend the dataset, we recruited three data analysis experts to design 38 additional transformation pipelines to generate visualizations containing discovered patterns. The number of pipelines for each table ranges from 1 to 4 and one pipeline may have 1 or 2 paths. All transformation pipelines consist of T in our space and visualization types consist of scatter plots, bar charts, and line charts, which are convenient and fair to compare with our approach. The resulting 50 pipelines form our ground-truth set.

For each table, we applied our recommendation approach with four pre-defined configurations and merged results to record top-k recommendations. Then we checked how many expert-designed transformation pipelines appeared in our top-k recommendations. Two transformation operations are considered identical if they have the same transformation type and input. We denote the number as $S@k$ (success at top k). The $S@k$ results are shown in Table 2. Note that a recommendation with two transformation paths is counted in $S@k$ only when both paths match expert-designed transformation pipelines.

Experimental results showed that our approach achieved a comparable recommendation with ground

truth. Top-1 recommendations can produce results in the ground-truth set on 17 datasets, and top-5 recommendations can cover more than 80% of the ground-truth set. We further analyzed examples of positive false and negative true recommendations.

A positive false sample concerned a movie characters data table in a distribution-wise configuration, in which each item represents a character and each attribute represents a feature. The system selects all numeric attributes, applies t-SNE to generate dimension reduction results as two-dimensional scatter coordinates, and selects all rating attributes to calculate a sum encoded as scatter color, while the expert uses a simple PCA to complete dimension reduction. It is intuitive for the expert to select PCA because the dataset is simple, but the system obtains a higher score on t-SNE because it shows more definite clustering. An example of unrecommended ground truth concerns a population data table. The expert calculates the population increase rate for each year and the change speed of the rate by using the difference. Such transformations with specific real meanings are difficult for the approach to discover. The closest recommendation from our approach is the average value of population changes.

Table 2 $S@k$ score of our approach on experimental datasets

k	$S@k$	New	k	$S@k$	New
1	17	17	5	41	6
3	35	18	10	44	3

6.3 User study

We conducted a user study to evaluate whether the recommendation system makes discovering patterns in data easier. We recruited 12 computer science graduate students as study participants (four females and eight males). All participants (U1–U12) have data analysis experience (2–4 years) and have mastered at least one data analysis tool including Python packages (numpy, pandas, scikit, matplotlib) and R.

6.3.1 Datasets and settings

We used the two datasets (D-1, D-2) in use cases for user study, which have suitable complexity and contain relatively rich patterns. The data backgrounds are also easy to understand for participants. We further divided participants into four groups (A-1, A-2, B-1, B-2) randomly to balance the influence of experimental order.

6.3.2 Procedure

We first gave a tutorial to participants about how to use the system to analyze an example Iris dataset (10 min). All views were described in detail to participants. In particular, we explained the function and effect of each scoring metric to ensure that participants were able to configure metrics without confusion. Meanwhile, we encouraged participants to use the pre-defined configurations.

Participants began their exploration and analysis after the tutorial. The goal was to find and collect insights in data and show them in visualizations. Groups A-1 and A-2 used the recommendation system on D-1 and their familiar tool (Python or R) on D-2, while groups B-1 and B-2 did the opposite. Groups A-1 and B-1 analyzed D-1 first (15 min) and D-2 later (15 min), while groups A-2 and B-2 did the opposite. During experiments, we provided Tableau and PowerBI for participants as optional tools to recommend visual encoding and generate visualizations. We recorded queries and interactions of participants.

After finishing both studies, participants were asked to fill a five-point Likert scale (20 min) and interviewed about how they used the system.

6.3.3 Results

We checked insights collected by participants and removed duplicate ones. The resulting numbers of insights are shown in Fig. 8, where the number of insights with recommendation was notably higher. Although the insight definition and boundary were relatively vague for participants, the results suggested a significant improvement brought by recommendations. In addition, we found that using the recommendation system first caused participants to collect more insights in subsequent manual

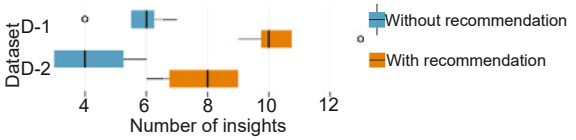

Fig. 8 Numbers of discovered insights in the user study. Each dataset is analyzed by six participants with recommendation and six participants without recommendation

exploration. This may result from inspirations created by the recommendations.

6.3.4 Feedback

Overall, our approach received a positive response from participants. As shown in Fig. 9, participants generally agreed that the approach provides explainable and useful results and that the system assists them in using the approach. All participants commented on the convenience brought by the recommendations. Also, they affirmed the recommendation effectiveness on explainability and diversity. U2, also a participant in the pilot study, said, "The system provides natural recommendations. Unlike pilot study, all visualizations can be understand and I can show them to the audience directly." U6 said, "The system provides many different results. I can compare them. Specifically, I can compare transformation pipelines with an identical path using transformation overview." By asking the operation logs, we found that recommendation provides useful help when participants have no idea about the dataset or their initial ideas have been explored. About the role that recommendation plays, U2 said, "Actually, I complete the study by 'cooperating' with the recommendation system. Recommendations show me a global understanding and help me find patterns in breadth. I inform the system of realistic meanings to find detailed information." Other participants reported similar situations. Such feedbacks suggest how automation promotes pattern discovery. Helping discover ignored details was also mentioned (U1 and U8). In addition, U11 noted an unsatisfactory condition when he wanted to study a specific transformation when the recommendation results did not contain it. "I have to create that transformation through interaction provided by the system." These advantages and limitations of the automatic approach encourage us to design a mixed-initiative system in the future.

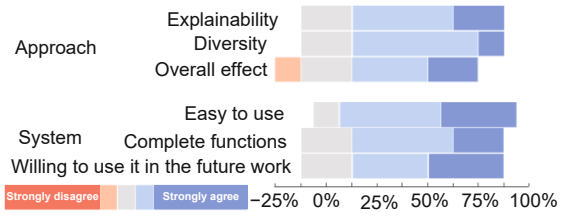

Fig. 9 User ratings for the recommendation approach and system on a five-point Likert scale (12 participants)

7 Discussion

7.1 Transformation recommendation framework

The work in this study can be regarded as a framework for transformation recommendation.

The proposed approach provides a transformation space for the visualization domain, which ranges from data preparation and data mining to data visualization, and covers most of the commonly used data transformation operations. The space benefits further research on automatic visualization by enriching supportable transformations. More transformations, even user-defined transforming code, can be added to the space easily as only input and output information is required to be specified.

The tree-structure formulation offers an enumeration approach for finding feasible transformation pipelines in the space. The pilot study shows factors that influence the explainability of transformation pipelines. Based on the two results, a quantified framework of explainability is proposed by addressing three common aspects of the problem. If other explainability factors appear in specific domain research, specially designed metrics can be added to the framework directly without normalization considerations, because of the proposed multi-objective optimization scheme which synthesizes different metrics.

The three parts of the search approach, i.e., the DM-centric searching space, the pruning search method, and the scoring functions, are all extensive. The searching space is constructed based on the transformation space, and therefore can accept more transformation operations. The pruning search can deal with a number of optimization objectives and allows adjustment of pruning standards. The scoring function is a relatively independent module and is therefore completely replaceable, enabling the evaluation metrics on both data and visualization images.

Otherwise, the proposed approach has high scalability. The DM-centric searching space constrains the search range to accelerate and enable parallel searching. The priority setting of attribute groups enables users to control the time limit of the

approach. For a user-uploaded dataset with many rows and columns, attribute grouping keeps the approach from attempting all combinations of attributes and explainable pruning search keeps the approach from calculating time-consuming transformations to decide on candidates.

By regarding our recommendation approach as a framework, a reduced edition of the approach can be configured according to the requirements and added to practical automatic visualization systems to enhance the performance.

7.2 Future work

7.2.1 Mixed-initiative system

The user study suggests that a mixed-initiative interaction mode fits pattern discovery, where an automatic agent recommends potential transformations and users evaluate and select results. In such a mode, users can ideally ignore the recommending details of the system and do not need halfway adjustments of system parameters while the system decides new recommendations according to users' selection. The mixed-initiative mode provides more convenience for users and balances automation and human efforts. In this study, we propose a recommendation approach for data transformation and implement a prototype system that users can adjust. In future work, we expect to design approaches to infer intentions from user interactions and update recommendations quickly for developing a mixed-initiative pattern discovery system.

In a mixed-initiative system, users need to understand system-recommended transformation pipelines. Sometimes it is also meaningful to compare different pipelines (Giovannangeli et al., 2020), especially when pipelines share one path. The prototype system in this study provides a preliminary form, and more studies of visualization and interaction design are needed (Chegini et al., 2020). In particular, we expect to design visual analysis approaches fitting mixed-initiative systems to realize efficient collaboration between human beings and machines (Collins et al., 2018; Chen W et al., 2021).

7.2.2 Design space

As explainable transformation recommendation is a complex problem, the pilot study in this paper may be preliminary. In the future, we plan to perform more experiments to learn more about explainability, e.g., (1) finding more potential factors of explainability, (2) attempting to collect quantitative indicators in large-scale experiments, and (3) collecting user requirements for mixed-initiative systems. Based on these results, a transformation recommendation design space can be built. Furthermore, we hope to study the recommender representation space (Rattaphun et al., 2022) and identify alternative approaches rather than heuristic approaches to optimize approach details and facilitate evaluations.

7.2.3 Learning-based method

In this study, we leverage skyline operation to solve the multi-objective optimization problem. We hope to use recent achievements in reinforcement learning (Wang HN et al., 2020) to generate explainable transformations in the future. Compared to general machine learning methods, reinforcement learning needs a responsive environment rather than sufficient and comprehensive data. We propose that the mixed-initiative system can serve as an environment that responds to user selections as feedback to the reinforcement learning agent. In this way, the agent learns explainability patterns from user interactions, thus obtaining the recommendation policy.

8 Conclusions

In this paper, we proposed a novel data transformation recommendation approach for enhancing the pattern discovery ability of automatic visualization. We first summarized a space of feasible data transformations by surveying the literature. To deal with the explainability problem appearing in data transformations, we conducted a pilot study to locate three significant influencing factors and proposed the comprehension load to quantify them. A Pareto-optimal concept was introduced to synthesize load metrics from a multi-objective perspective, and pruning searching was designed to realize optimization and explainability. We implemented a prototype system to validate and evaluate our approach. Use cases and user study demonstrated the effectiveness and efficiency of our approach. In the future, we will explore learning-based methods to develop a mixed-initiative recommendation system.

Contributors

Ziliang WU designed the research and drafted the paper. Wei CHEN, Yuxin MA, and Jiazhi XIA helped organize the paper. Ziliang WU, Tong XU, and Lei LV implemented the system. Fan YAN and Zhonghao QIAN collected the data. Wei CHEN revised and finalized the paper.

Compliance with ethics guidelines

Ziliang WU, Wei CHEN, Yuxin MA, Tong XU, Fan YAN, Lei LV, Zhonghao QIAN, and Jiazhi XIA declare that they have no conflict of interest.

Data availability

The data that support the findings of this study are available from the corresponding author upon reasonable request.

References

Abdi H, Williams LJ, 2010. Principal component analysis. *WIRE Comput Stat*, 2(4):433-459. https://doi.org/10.1002/wics.101

Borzsony S, Kossmann D, Stocker K, 2001. The skyline operator. Proc 17th Int Conf on Data Engineering, p.421-430. https://doi.org/10.1109/ICDE.2001.914855

Burkart N, Huber MF, 2021. A survey on the explainability of supervised machine learning. *J Artif Intell Res*, 70:245-317. https://doi.org/10.1613/jair.1.12228

Cao MQ, Liang J, Li MZ, et al., 2020. TDIVis: visual analysis of tourism destination images. *Front Inform Technol Electron Eng*, 21(4):536-557. https://doi.org/10.1631/FITEE.1900631

Chakraborty S, Nagwani NK, 2014. Analysis and study of incremental DBSCAN clustering algorithm. https://arxiv.org/abs/1406.4754

Chegini M, Bernard J, Cui J, et al., 2020. Interactive visual labelling versus active learning: an experimental comparison. *Front Inform Technol Electron Eng*, 21(4):524-535. https://doi.org/10.1631/FITEE.1900549

Chen BY, Wu H, Mo W, et al., 2018. Autostacker: a compositional evolutionary learning system. Proc Genetic and Evolutionary Computation Conf, p.402-409. https://doi.org/10.1145/3205455.3205586

Chen SM, Andrienko N, Andrienko G, et al., 2020. LDA ensembles for interactive exploration and categorization of behaviors. *IEEE Trans Visual Comput Graph*, 26(9):2775-2792. https://doi.org/10.1109/TVCG.2019.2904069

Chen W, Zhang TY, Zhu HY, et al., 2021. Perspectives on cross-domain visual analysis of cyber-physical-social big data. *Front Inform Technol Electron Eng*, 22(12):1559-1564. https://doi.org/10.1631/FITEE.2100553

Collins C, Andrienko N, Schreck T, et al., 2018. Guidance in the human-machine analytics process. *Vis Inform*, 2(3):166-180. https://doi.org/10.1016/j.visinf.2018.09.003

Cui Z, Badam SK, Yalçin MA, et al., 2019. DataSite: proactive visual data exploration with computation of insight-based recommendations. *Inform Visual*, 18(2):251-267. https://doi.org/10.1177/1473871618806555

Dang TN, Wilkinson L, 2014. ScagExplorer: exploring scatterplots by their scagnostics. Proc IEEE Pacific Visualization Symp, p.73-80. https://doi.org/10.1109/PacificVis.2014.42

Demiralp Ç, Haas PJ, Parthasarathy S, et al., 2017. Foresight: recommending visual insights. *Proc VLDB Endow*, 10(12):1937-1940. https://doi.org/10.14778/3137765.3137813

Dey K, Shrivastava R, Kaushik S, et al., 2017. EmTaggeR: a word embedding based novel method for hashtag recommendation on Twitter. Proc IEEE Int Conf on Data Mining Workshops, p.1025-1032. https://doi.org/10.1109/ICDMW.2017.145

Dibia V, Demiralp Ç, 2019. Data2Vis: automatic generation of data visualizations using sequence-to-sequence recurrent neural networks. *IEEE Comput Graph Appl*, 39(5):33-46. https://doi.org/10.1109/MCG.2019.2924636

Ding R, Han S, Xu Y, et al., 2019. QuickInsights: quick and automatic discovery of insights from multi-dimensional data. Proc ACM SIGMOD Int Conf on Management of Data, p.317-332. https://doi.org/10.1145/3299869.3314037

Dong XB, Yu ZW, Cao WM, et al., 2020. A survey on ensemble learning. *Front Comput Sci*, 14(2):241-258. https://doi.org/10.1007/s11704-019-8208-z

Du L, Gao F, Chen X, et al., 2021. TabularNet: a neural network architecture for understanding semantic structures of tabular data. Proc 27th ACM SIGKDD Conf on Knowledge Discovery & Data Mining, p.322-331. https://doi.org/10.1145/3447548.3467228

Fu P, Lin Z, Yuan FC, et al., 2018. Learning sentiment-specific word embedding via global sentiment representation. Proc AAAI Conf on Artificial Intelligence, p.4808-4815. https://doi.org/10.1609/aaai.v32i1.11916

Geng LQ, Hamilton HJ, 2006. Interestingness measures for data mining: a survey. *ACM Comput Surv*, 38(3):9. https://doi.org/10.1145/1132960.1132963

Giovannangeli L, Bourqui R, Giot R, et al., 2020. Toward automatic comparison of visualization techniques: application to graph visualization. *Vis Inform*, 4(2):86-98. https://doi.org/10.1016/j.visinf.2020.04.002

Gleicher M, 2013. Explainers: expert explorations with crafted projections. *IEEE Trans Visual Comput Graph*, 19(12):2042-2051. https://doi.org/10.1109/TVCG.2013.157

Golfarelli M, Rizzi S, 2018. From star schemas to big data: 20+ years of data warehouse research. In: Flesca S, Greco S, Masciari E, et al. (Eds.), A Comprehensive Guide Through the Italian Database Research Over the Last 25 Years. Springer, Cham, p.93-107. https://doi.org/10.1007/978-3-319-61893-7_6

He YY, Ganjam K, Lee K, et al., 2018a. Transform-data-by-example (TDE): extensible data transformation in Excel. Proc ACM SIGMOD Int Conf on Management of Data, p.1785-1788. https://doi.org/10.1145/3183713.3193539

He YY, Chu X, Ganjam K, et al., 2018b. Transform-data-by-example (TDE): an extensible search engine for data transformations. *Proc VLDB Endow*, 11(10):1165-1177. https://doi.org/10.14778/3231751.3231766

Heffetz Y, Vainshtein R, Katz G, et al., 2020. DeepLine: AutoML tool for pipelines generation using deep reinforcement learning and hierarchical actions filtering. Proc 26th ACM SIGKDD Conf on Knowledge Discovery & Data Mining, p.2103-2113. https://doi.org/10.1145/3394486.3403261

Hu K, Orghian D, Hidalgo CA, 2018. DIVE: a mixed-initiative system supporting integrated data exploration workflows. Proc Workshop on Human-in-the-Loop Data Analytics, Article 5. https://doi.org/10.1145/3209900.3209910

Hu K, Bakker MA, Li S, et al., 2019. VizML: a machine learning approach to visualization recommendation. Proc CHI Conf on Human Factors in Computing Systems, Article 128. https://doi.org/10.1145/3290605.3300358

Ilyas A, da Trindade JMF, Fernandez RC, et al., 2018. Extracting syntactical patterns from databases. Proc 34th IEEE Int Conf on Data Engineering, p.41-52. https://doi.org/10.1109/ICDE.2018.00014

Ingram S, Munzner T, Irvine V, et al., 2010. DimStiller: workflows for dimensional analysis and reduction. Proc IEEE Symp on Visual Analytics Science and Technology, p.3-10. https://doi.org/10.1109/VAST.2010.5652392

Jin ZJ, Anderson MR, Cafarella M, et al., 2017. Foofah: transforming data by example. Proc ACM Int Conf on Management of Data, p.683-698. https://doi.org/10.1145/3035918.3064034

Jin ZJ, He YY, Chauduri S, 2020. Auto-transform: learning-to-transform by patterns. *Proc VLDB Endow*, 13(12):2368-2381. https://doi.org/10.14778/3407790.3407831

Kanter JM, Veeramachaneni K, 2015. Deep feature synthesis: towards automating data science endeavors. Proc IEEE Int Conf on Data Science and Advanced Analytics, p.1-10. https://doi.org/10.1109/DSAA.2015.7344858

Katz G, Shin ECR, Song D, 2016. ExploreKit: automatic feature generation and selection. Proc 16th IEEE Int Conf on Data Mining, p.979-984. https://doi.org/10.1109/ICDM.2016.0123

Kaul A, Maheshwary S, Pudi V, 2017. AutoLearn—automated feature generation and selection. Proc IEEE Int Conf on Data Mining, p.217-226. https://doi.org/10.1109/ICDM.2017.31

Khurana U, Turaga D, Samulowitz H, et al., 2016. Cognito: automated feature engineering for supervised learning. Proc 16th IEEE Int Conf on Data Mining Workshops, p.1304-1307. https://doi.org/10.1109/ICDMW.2016.0190

Khurana U, Samulowitz H, Turaga D, 2018. Ensembles with automated feature engineering. ICML AutoML Workshop.

Kolouri S, Pope PE, Martin CE, et al., 2018. Sliced-Wasserstein auto-encoders. Proc 17th Int Conf on Learning Representations.

Lam HT, Thiebaut JM, Sinn M, et al., 2017. One button machine for automating feature engineering in relational databases. https://arxiv.org/abs/1706.00327

Law PM, Endert A, Stasko J, 2020. Characterizing automated data insights. Proc IEEE Visualization Conf, p.171-175. https://doi.org/10.1109/VIS47514.2020.00041

Li DQ, Mei HH, Shen Y, et al., 2018. ECharts: a declarative framework for rapid construction of web-based visualization. *Vis Inform*, 2(2):136-146. https://doi.org/10.1016/j.visinf.2018.04.011

Li HT, Wang Y, Zhang SH, et al., 2022. KG4Vis: a knowledge graph-based approach for visualization recommendation. *IEEE Trans Vis Comput Graph*, 28(1):195-205. https://doi.org/10.1109/TVCG.2021.3114863

Lin H, Moritz D, Heer J, 2020. Dziban: balancing agency & automation in visualization design via anchored recommendations. Proc CHI Conf on Human Factors in Computing Systems, p.1-12. https://doi.org/10.1145/3313831.3376880

Liu JF, Xiong L, Pei J, et al., 2015. Finding Pareto optimal groups: group-based skyline. *Proc VLDB Endow*, 8(13):2086-2097. https://doi.org/10.14778/2831360.2831363

Liu SX, Andrienko G, Wu YC, et al., 2018. Steering data quality with visual analytics: the complexity challenge. *Vis Inform*, 2(4):191-197. https://doi.org/10.1016/j.visinf.2018.12.001

Lu JH, Chen W, Ma YX, et al., 2017. Recent progress and trends in predictive visual analytics. *Front Comput Sci*, 11(2):192-207. https://doi.org/10.1007/s11704-016-6028-y

Luo YY, Qin XD, Tang N, et al., 2018. DeepEye: towards automatic data visualization. Proc 34th IEEE Int Conf on Data Engineering, p.101-112. https://doi.org/10.1109/ICDE.2018.00019

McInnes L, Healy J, Melville J, 2018. UMAP: uniform manifold approximation and projection for dimension reduction. https://arxiv.org/abs/1802.03426v2

Mikolov T, Chen K, Corrado G, et al., 2013. Efficient estimation of word representations in vector space. Proc 1st Int Conf on Learning Representations.

Moritz D, Wang CL, Nelson GL, et al., 2019. Formalizing visualization design knowledge as constraints: actionable and extensible models in Draco. *IEEE Trans Visual Comput Graph*, 25(1):438-448. https://doi.org/10.1109/TVCG.2018.2865240

Nargesian F, Samulowitz H, Khurana U, et al., 2017. Learning feature engineering for classification. Proc 26th Int Joint Conf on Artificial Intelligence, p.2529-2535. https://doi.org/10.24963/ijcai.2017/352

Natani G, Watanabe S, 2021. Knowledge graph-based data transformation recommendation engine. Proc IEEE Int Conf on Big Data, p.4617-4623. https://doi.org/10.1109/BigData52589.2021.9671905

Ngatchou P, Zarei A, El-Sharkawi A, 2005. Pareto multi objective optimization. Proc 13th Int Conf on Intelligent Systems Application to Power Systems, p.84-91. https://doi.org/10.1109/ISAP.2005.1599245

Pan JC, Han DM, Guo FZ, et al., 2020. RCAnalyzer: visual analytics of rare categories in dynamic networks. *Front Inform Technol Electron Eng*, 21(4):491-506. https://doi.org/10.1631/FITEE.1900310

Pandey A, L'Yi S, Wang QW, et al., 2022. GenoREC: a recommendation system for interactive genomics data visualization. *IEEE Trans Visual Comput Graph*, early access. https://doi.org/10.1109/TVCG.2022.3209407

Qian X, Rossi RA, Du F, et al., 2021. Learning to recommend visualizations from data. Proc 27th ACM SIGKDD Conf on Knowledge Discovery & Data Mining, p.1359-1369. https://doi.org/10.1145/3447548.3467224

Qin XD, Luo YY, Tang N, et al., 2018. DeepEye: an automatic big data visualization framework. *Big Data Min Anal*, 1(1):75-82. https://doi.org/10.26599/BDMA.2018.9020007

Qin XD, Luo YY, Tang N, et al., 2020. Making data visualization more efficient and effective: a survey. *VLDB J*, 29(1):93-117. https://doi.org/10.1007/s00778-019-00588-3

Rattaphun M, Fang WC, Chiu CY, 2022. Attention on global-local representation spaces in recommender systems. *IEEE Trans Comput Soc Syst*, 9(5):1394-1405. https://doi.org/10.1109/TCSS.2021.3129482

Shen LX, Shen EY, Tai ZW, et al., 2021. TaskVis: task-oriented visualization recommendation. Proc Eurographics Conf on Visualization. https://doi.org/10.2312/evs.20211061

Shi DQ, Xu XY, Sun FL, et al., 2021. Calliope: automatic visual data story generation from a spreadsheet. *IEEE Trans Visual Comput Graph*, 27(2):453-463. https://doi.org/10.1109/TVCG.2020.3030403

Siddiqui T, Lee J, Kim A, et al., 2017. Fast-forwarding to desired visualizations with zenvisage. Proc 8th Biennial Conf on Innovative Data Systems Research.

Singh R, 2016. BlinkFill: semi-supervised programming by example for syntactic string transformations. *Proc VLDB Endow*, 9(10):816-827. https://doi.org/10.14778/2977797.2977807

Tang B, Han S, Yiu ML, et al., 2017. Extracting top-k insights from multi-dimensional data. Proc ACM Int Conf on Management of Data, p.1509-1524. https://doi.org/10.1145/3035918.3035922

Tatu A, Albuquerque G, Eisemann M, et al., 2009. Combining automated analysis and visualization techniques for effective exploration of high-dimensional data. Proc IEEE Symp on Visual Analytics Science and Technology, p.59-66. https://doi.org/10.1109/VAST.2009.5332628

Tran B, Xue B, Zhang MJ, 2016. Genetic programming for feature construction and selection in classification on high-dimensional data. *Memet Comput*, 8(1):3-15. https://doi.org/10.1007/s12293-015-0173-y

Vartak M, Madden S, Parameswaran A, et al., 2014. SeeDB: automatically generating query visualizations. *Proc VLDB Endow*, 7(13):1581-1584. https://doi.org/10.14778/2733004.2733035

Wang HN, Liu N, Zhang YY, et al., 2020. Deep reinforcement learning: a survey. *Front Inform Technol Electron Eng*, 21(12):1726-1744. https://doi.org/10.1631/FITEE.1900533

Wang Y, Sun ZD, Zhang HD, et al., 2019. DataShot: automatic generation of fact sheets from tabular data. *IEEE Trans Visual Comput Graph*, 26(1):895-905. https://doi.org/10.1109/TVCG.2019.2934398

Warren RH, Tompa FW, 2006. Multi-column substring matching for database schema translation. Proc 32nd Int Conf on Very Large Data Bases, p.331-342.

Wen Z, Zhou MX, 2008a. Evaluating the use of data transformation for information visualization. *IEEE Trans Vis Comput Graph*, 14(6):1309-1316. https://doi.org/10.1109/TVCG.2008.129

Wen Z, Zhou MX, 2008b. An optimization-based approach to dynamic data transformation for smart visualization. Proc 13th Int Conf on Intelligent User Interfaces, p.70-79. https://doi.org/10.1145/1378773.1378784

Wilkinson L, Anand A, Grossman R, 2005. Graph-theoretic scagnostics. Proc IEEE Symp on Information Visualization, p.157-164. https://doi.org/10.1109/INFVIS.2005.1532142

Wongsuphasawat K, Moritz D, Anand A, et al., 2016. Voyager: exploratory analysis via faceted browsing of visualization recommendations. *IEEE Trans Visual Comput Graph*, 22(1):649-658. https://doi.org/10.1109/TVCG.2015.2467191

Wongsuphasawat K, Qu ZN, Moritz D, et al., 2017. Voyager 2: augmenting visual analysis with partial view specifications. Proc CHI Conf on Human Factors in Computing Systems, p.2648-2659. https://doi.org/10.1145/3025453.3025768

Wu AY, Wang Y, Zhou MY, et al., 2022. MultiVision: designing analytical dashboards with deep learning based recommendation. *IEEE Trans Visual Comput Graph*, 28(1):162-172. https://doi.org/10.1109/TVCG.2021.3114826

Xia JZ, Zhang YH, Ye H, et al., 2020. SuPoolVisor: a visual analytics system for mining pool surveillance. *Front Inform Technol Electron Eng*, 21(4):507-523. https://doi.org/10.1631/FITEE.1900532

Yan C, He YY, 2020. Auto-suggest: learning-to-recommend data preparation steps using data science notebooks. Proc ACM SIGMOD Int Conf on Management of Data, p.1539-1554. https://doi.org/10.1145/3318464.3389738

Yao QM, Wang MS, Hugo JE, et al., 2018. Taking human out of learning applications: a survey on automated machine learning. https://arxiv.org/abs/1810.13306v1

Zeng ZH, Moh P, Du F, et al., 2022. An evaluation-focused framework for visualization recommendation algorithms. *IEEE Trans Visual Comput Graph*, 28(1):346-356. https://doi.org/10.1109/TVCG.2021.3114814

Zhou MY, Tao W, Ji PX, et al., 2020. Table2Analysis: modeling and recommendation of common analysis patterns for multi-dimensional data. Proc 34th AAAI Conf on Artificial Intelligence, p.320-328. https://doi.org/10.1609/aaai.v34i01.5366

Zhou MY, Li QT, He XY, et al., 2021. Table2Charts: recommending charts by learning shared table representations. Proc 27th ACM SIGKDD Conf on Knowledge Discovery & Data Mining, p.2389-2399. https://doi.org/10.1145/3447548.3467279

Zhu EK, He YY, Chaudhuri S, 2017. Auto-Join: joining tables by leveraging transformations. *Proc VLDB Endow*, 10(10):1034-1045. https://doi.org/10.14778/3115404.3115409

Zhu SJ, Sun GD, Jiang Q, et al., 2020. A survey on automatic infographics and visualization recommendations. *Vis Inform*, 4(3):24-40. https://doi.org/10.1016/j.visinf.2020.07.002

Zöller MA, Huber MF, 2021. Benchmark and survey of automated machine learning frameworks. *J Artif Intell Res*, 70:409-472. https://doi.org/10.1613/jair.1.11854