Info Vis

Article

Information Visualization
1–18
© The Author(s) 2024
Article reuse guidelines:
sagepub.com/journals-permissions
DOI: 10.1177/14738716241265120
journals.sagepub.com/home/ivi
Sage

A framework to improve causal inferences from visualizations using counterfactual operators

Arran Zeyu Wang[1], David Borland[2] and David Gotz[3]

Abstract
Exploratory data analysis of high-dimensional datasets is a crucial task for which visual analytics can be especially useful. However, the ad hoc nature of exploratory analysis can also lead users to draw incorrect causal inferences. Previous studies have demonstrated this risk and shown that integrating counterfactual concepts within visual analytics systems can improve users' understanding of visualized data. However, effectively leveraging counterfactual concepts can be challenging, with only bespoke implementations found in prior work. Moreover, it can require expertise in both counterfactual subset analysis and visualization to implement the functionalities practically. This paper aims to help address these challenges in two ways. First, we propose an operator-based conceptual model for the use of counterfactuals that is informed by prior work in visualization research. Second, we contribute the *Co-op* library, an open and extensible reference implementation of this model that can support the integration of counterfactual-based subset computation with visualization systems. To evaluate the effectiveness and generalizability of *Co-op*, the library was used to construct two different visual analytics systems each supporting a distinct user workflow. In addition, expert interviews were conducted with professional visual analytics researchers and engineers to gain more insights regarding how *Co-op* could be leveraged. Finally, informed in part by these evaluation results, we distil a set of key design implications for effectively leveraging counterfactuals in future visualization systems.

Keywords
Counterfactual, visualization, causal inference, operator, data subset

Introduction

Visual analytics systems have had a significant impact across numerous application domains, ranging from healthcare[1] to event sequence analysis[2] to decision-making.[3] Building on the rich history of visualization and exploratory data analysis,[4,5] a primary goal of many visual analytics systems is the exploration of data to facilitate pattern discovery and generate new insights. Such systems aim to enable users to quickly generate new views of data and to explore the relationships between variables in ad hoc ways. This can result in many different visualizations of various data subsets, each quickly created as part of the exploratory analysis process.[6–8] Users view these visual depictions in an attempt to identify and interpret meaningful signals within the data. This last step can be challenging, however, as it can be difficult to distinguish between visual patterns that are consequential for a

[1]Department of Computer Science, The University of North Carolina at Chapel Hill, Chapel Hill, NC, USA
[2]RENCI, The University of North Carolina at Chapel Hill, Chapel Hill, NC, USA
[3]School of Information and Library Science, The University of North Carolina at Chapel Hil, Chapel Hill, NC, USA

Corresponding author:
David Gotz, School of Information and Library Science, The University of North Carolina at Chapel Hill, Manning Hall, 216 Lenoir Drive, Chapel Hill, NC 27599, USA.
Email: gotz@unc.edu

given task from those that might result only from noise or spurious correlations.

This challenge becomes even more difficult with the increasing complexity of large-scale datasets and visual designs which can make it even more difficult for users to properly comprehend and interpret the views created within visualizations and visual analytics systems.[9–11] Substantial efforts from visualization researchers have thus been made to find new approaches that can aid people in better understanding data and avoiding false findings.[12,13] These efforts include research that has studied how people draw potentially unsupported causal conclusions about visualized data relationships, along with proposed methods to mitigate this effect.[2,14] Despite these efforts, however, the creation of visualization tools that can support the reliable identification of causal relationships remains a significant and critical challenge.[15]

This challenge is closely related to the topic of causal inference as studied within the statistics community. One core theory in this area of statistics is *counterfactual reasoning*,[16] an approach to thinking about causation from the perspective of considering hypothetical scenarios (i.e. counterfactual scenarios) in which alternative conditions that differ from an original observation were to have occurred. For example, an analysis of data related to students' academic performance without adequate sleep might consider the counterfactual scenario in which similar students had in fact slept the fully recommended amount of time. This approach provides the conceptual basis for analysts to reason about causal relationships between factors by isolating and examining "what-if" conditions for specific variables.

Counterfactual approaches have been more broadly adopted beyond the statistics community in recent years as data-driven applications have become more ubiquitous. One such domain is the burgeoning machine learning community, which has demonstrated several useful applications of counterfactual concepts including model fairness[17] and evaluations.[18]

Within the visualization community, recent studies have assessed the impact of visualization on the quality of causal inferences[14,19] and shown that integrating counterfactuals with visualizations and visual analytics systems can improve users' understanding of causal relationships and overall data interpretation.[6,20–22] These approaches have shown great promise in studies of users' analytical behavior. However, they have depended upon bespoke implementations and proof-of-concept prototypes that make wider adoption of these techniques more difficult.[23] Specifically, adopting counterfactual techniques is reliant on developers to design the data model, workflows, and algorithmic modules required to integrate these concepts within an exploratory visualization environment.[23]

Motivated by lessons learned from prior studies of counterfactual visualizations such as *CoFact*,[6] this paper derives a set of general design motivations and introduces a formal *model of counterfactual operators*. This model is designed to provide a deeper conceptual foundation for the use of counterfactual reasoning via subset computations within the context of visualization and visual analytics. At the core of this approach is a data subset-based mathematical computation model and a classification of counterfactual operators that reflect a range of key concepts important to the use of counterfactuals within visualization systems.[6,22] These operators transform or derive values from the underlying sets in various ways which, when used in combination, can support a range of counterfactual-based visualization workflows. This generalized approach to a counterfactual-based subset computation model can be leveraged by a range of visualization designs even beyond those from the prior work that informed its design.

As an instantiation of the proposed model, this paper also presents the *Co-op* library. *Co-op* implements the core counterfactual operators and corresponding set-based data structures proposed in our model. This library enables accelerated development of new counterfactual-based visualizations, and provides an extensible framework via which developers can apply and extend the core set of operators introduced in this paper. With *Co-op*, developers can easily design and implement visualizations integrating counterfactual subsets.

The effectiveness and versatility of *Co-op* is demonstrated through the development of two visualization systems, along with expert interviews. The first system is a re-implementation of the *CoFact* system[6] created using *Co-op* operators in place of the system's original computational methods. This workflow of *CoFact*, however, only uses a portion of the capabilities offered by the *Co-op* library. A second novel exploratory visual analysis system, *CoExplorer*, was created by utilizing a wider range of *Co-op*'s functionality to facilitate a new exploratory workflow incorporating counterfactual visualization that has not been reported in prior work. Further, we conducted an interview with six visual analytics researchers and professional engineers. They provided valuable feedback on the design goals, *Co-op* model, as well as potential benefits and drawbacks of practical aspects of the library.

Together, these two systems—developed with the same *Co-op* library at their core—and the expert interview help demonstrate the versatility of the counterfactual operator model. They also showcase the utility of the *Co-op* library itself as a tool for creating a diverse range of counterfactual visualizations that provide a variety of interactive functions and support various data types.

Finally, this paper offers general guidance on the effective usage of *Co-op*. This guidance distils key design implications that can aid researchers and practitioners in the visualization community in developing improved counterfactual-based visual analytics systems.

Related work

The counterfactual operators model introduced in this paper builds on prior research developed in several areas of related work. This includes foundational work that has developed the theory of counterfactual reasoning to support causal inference, causal inference visualizations, and how mathematical models can contribute to visualization.

Counterfactuals in causal inference

Causal inference techniques aim to support the understanding of relationships of cause and effect between factors in a system. Pearl[16] established a three-level causal inference hierarchy that describes progressively more powerful – and more difficult to attain – levels of understanding: association, intervention, and counterfactual.

Within this hierarchy each level builds upon the previous, with counterfactuals (providing the highest level of evidence for causation) as the apex. Counterfactuals enable the exploration of "what if" scenarios given changes to variables, and reasoning about "why" changes in the expected outcomes occur in response. In this way, counterfactuals offer a window to imagine hypothetical situations that did not necessarily occur in reality.

Take, for instance, a students poor performance on assessments after a sleepless night. A counterfactual analysis might probe scenarios where the student had adequate sleep, helping examine if lack of sleep is indeed a causal factor in poor assessment performance. There are many methods for estimating and computing counterfactuals in causal modeling such as instrumental variables,[24] machine learning approaches,[25] and matching methods.[26–28] For a more comprehensive overview of counterfactual-based causal inference, we refer the reader to Glymour et al.[29]

Counterfactual analyses, through the identification and/or simulation of unobserved scenarios, enhance our exploration of how certain factors may influence outcomes, enabling the examination of possible causal links, including those that were not directly observed within the data.

Visualization for causal inference

Understanding causal relations from complex data has long been an important goal for the visual analysis community.[30] Counterfactuals are emerging as a promising approach toward achieving this analytical need.[23]

Past efforts have often centered around the use of graphical causal models, for example, domain knowledge-enhanced visual exploration,[31,32] algorithm interpretation,[33] and DAG-based causal representation.[34] Visualizations using graphical models have been designed for specific application scenarios, such as supporting decision-making workflows,[35] urban time series exploration,[36] and event sequence analysis.[2]

Most relevant, Kaul et al.[6] introduced *CoFact*, the first use of counterfactuals in the context of general-purpose exploratory visual analysis scenarios with high-dimensional datasets. It includes a filter-constraint-based counterfactual computation method to dynamically create ad hoc comparator groups for improved user inference of causal relationships. Wang et al.[22] explored the use of counterfactuals in static visualizations and found that it helped users better infer relationships in datasets. These studies demonstrate the utility of visualizing counterfactuals, but have limited flexibility to extend beyond their proposed application scenarios and make no attempt to define a more generalizable framework that could enable the broader adoption of this type of approach.[23]

Inspired by these early steps, this paper proposes a formal operator-based model that provides a generalized approach to counterfactual computations that can support a unified approach to a wide range of visual analysis system designs.

Mathematical computation models in visualization

Mathematical computation models play a crucial role in enhancing the understanding and effectiveness of visualizations.[37] Widely-employed mathematical models in visualizations including discrete mathematics, such as network centrality measures[38,39] and graph-theoretic measures,[40,41] and statistical methods, such as dimensionality reduction[42,43] and Bayesian modeling.[44,45] Further, mathematical models have been applied to describe general visualization frameworks. For example, concepts and measurements from information theory such as entropy can be employed to qualify visual information[46] algebraic mathematical structures can help characterize data and encodings in visualization design,[47] and visualization tasks can be generalized into computational operations and pipelines.[48] By leveraging mathematical concepts, these models provide generalizable approaches based on theoretical foundations that can offer easier computation of data characteristics and enable more effective representations in visualizations. In a similar manner, our operator-based conceptual model and reference implementation are designed to enable more efficient and effective counterfactual-based visualization.

Counterfactual subset

The basic process of counterfactual-based analysis is built upon a foundation of sets, set manipulation, and set comparison. This section introduces core set-related concepts and essential terminology that form the basis for the design of the *Co-op* model described later in this paper.

Data subset selection

In high-dimensional datasets, processing or visualizing the entire dataset can be challenging, and generating data subsets becomes crucial to provide users with at-a-glance information.[49] Similarly, data subset selection methods are widely employed in various machine learning tasks, including enhancing model robustness[50] and reducing the size of training data.[51] Meanwhile, in exploratory data analysis, analysts often manually create subsets by filtering data based on specific drill-down choices.[52] For instance, assuming a user chooses a constraint such as *brand = Ford* in a car model dataset, a subset can be created and visualized by filtering all data points that correspond to Ford cars.

As the usage scenarios above highlight, the subset selection process is typically used to identify a smaller, more focused, subset of data from a larger data collection. The selection criteria for this process are often referred to as *inclusion criteria*, and we refer to the resulting subset (the data points that fit the inclusion criteria) as the *included subset*, S_{INCL}.

Critically, the selection process implicitly creates a second complementary subset that contains the remaining data points: the portion of the dataset that did not meet the inclusion criteria. We refer to this as the *excluded subset*, S_{EXCL}. For instance, using the *brand = Ford* example from earlier in this section, the excluded subset would include all non-Ford vehicles.

Mathematically, S_{INCL} can be defined as follows:

$$S_{INCL} = \{x \in S_{ALL} | F(x)\}, \quad (1)$$

where S_{ALL} is the complete dataset, x represents the individual data points, and $F(x)$ denotes the filtering of data to enforce the inclusion criteria. Similarly, S_{EXCL} can be defined as follows:

$$S_{EXCL} = \{x \in S_{ALL} | x \notin S_{INCL}\}. \quad (2)$$

Throughout a user's exploratory analysis session, the values for S_{INCL} and S_{EXCL} evolve through changes to the filter function $F(x)$ that reflect the user's dynamic analytic focus.

Counterfactual subset filtering

In many visual analytics systems, data from S_{INCL} is visualized in isolation to enable users to focus their analysis on a specific subset of their choice. This is reflected in the zoom and filter step of the oft-cited visualization mantra "Overview first, zoom and filter, then details-on-demand."[53] In other systems, S_{INCL} can be visualized together with data from S_{EXCL} to facilitate comparison between the two subsets.

In exploratory analysis settings, the subset S_{INCL} will change interactively in response to a user's changes in analysis focus. Traditionally, these changes would be driven in response to a user's explicit requests. More recently, a number of semi-automated approaches have also been proposed to help users more effectively explore complex datasets and find more meaningful insights. These include, for example, contextualizing selection bias[54–57] and aggregating data subsets.[58–60]

For both manual exploration and semi-automated approaches, the choices for S_{INCL} and the comparisons with S_{EXCL} that inform a user's insights are often based on correlations between variables that are either visually presented or algorithmically computed. Relying on correlations, however, can result in misleading representations and incorrect causal inferences on the part of the user.[16] Hence the well-known axiom that correlation does not imply causation.

Counterfactuals can provide a mechanism for more effectively contextualizing the correlations discovered in traditional exploratory analysis. As demonstrated in earlier work, comparing the data from S_{INCL} with a subset of data points from S_{EXCL} that are similar to the point in S_{INCL} across all dimensions other than the constraints can lead to more accurate user judgments of causal relationships.[6] These similar data points from S_{EXCL} serve as a counterfactual subset S_{CF} for S_{INCL}, providing more information about potential causal relationships in the data. One challenge for this approach is identifying an appropriate subset of data points to place within S_{CF}.

Mathematically, S_{CF} is defined as follows:

$$S_{CF} = \{x \in S_{EXCL} | Similar(x, S_{INCL})\}, \quad (3)$$

where *Similar* (x, S_{INCL}) denotes the computational process of determining if x is one of the *most similar* data points in S_{EXCL} to S_{INCL}.

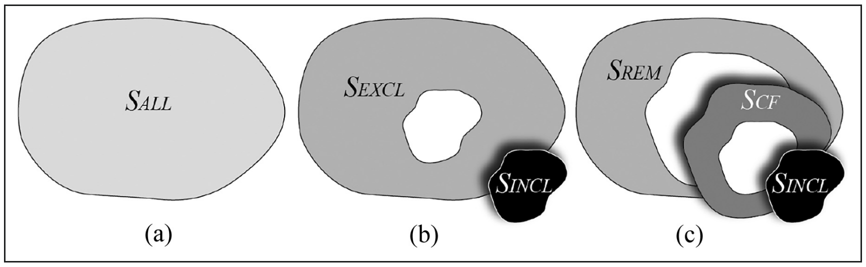

Figure 1. The subsets S_{INCL}, S_{EXCL}, S_{CF}, and S_{REM} are derived from S_{ALL} as illustrated in this figure. (a) S_{ALL} contains all data. Applying a filter creates (b) the included subset S_{INCL} and a corresponding excluded subset S_{EXCL}. Selecting counterfactuals from the excluded set divides S_{EXCL} into (c) the counterfactual subset S_{CF} (closest to S_{INCL}) and the remaining excluded data points S_{REM}.

We note that this definition partitions S_{EXCL} into two subsets: (a) S_{CF}, and (b) the remainder of the excluded subset that is not part of the counterfactual subset. We refer to this remainder as S_{REM} which we define mathematically as follows:

$$S_{REM} = S_{EXCL} - S_{CF} \tag{4}$$

The relationships between S_{ALL}, S_{INCL}, S_{EXCL}, S_{CF}, and S_{REM} are illustrated in Figure 1.

This definition of S_{CF} sidesteps a critical question: What is meant by similarity and how do we define the *Similar* function? A wide variety of similarity metrics have been proposed in the literature. For example, *Cofact*[6] employed a simple Euclidean distance measure as the similarity metric for subset computation. However, in many real-world applications such a simplistic approach is often insufficient. The demands of different analytical goals vary widely,[60,61] which may necessitate a variety of similarity computation methods. Moreover, similarity is context-dependent. Consider a health example focused on the similarity between two patients. The two patients may be more similar with respect to cardiac questions but less similar with respect to dermatological questions. These considerations mean that calculations of similarity are necessarily application and context specific.

In this paper, instead of making a fool-hardy attempt to explicitly solve this problem by introducing "the right metric" for all cases, we define similarity flexibly based on an extensible set of measures that can describe the relations between data points and subsets. These can then be used individually or in combination to express more sophisticated concepts of similarity as appropriate for specific scenarios.

Design motivation

To highlight the various ways in which visualization developers can leverage counterfactual reasoning, we introduce a series of usage scenarios and discuss the key design considerations that those scenarios raise. We then outline a set of design goals for our work based on those considerations.

Usage scenarios

To help illustrate the wide variety of ways in which counterfactuals can be useful within the context of visualization, we describe four different usage scenarios. For each scenario, a number of key design considerations drawn from previous work are highlighted in **bold**.

Algorithm Explanation with Visual Analytic Effective graphical inference is an important consideration for the design of explanatory visualizations as it is crucial that such systems help users draw valid conclusions from their data exploration and analytical focus.[10,62] In model explanation, which is the most widely applied counterfactual usage scenario for model developers, counterfactuals have been incorporated with visual analytics systems to provide improved graphical inference.[20,21,63,64] These analytic systems leverage counterfactuals to provide intuitive explanations for model decisions by creating synthetic or modified data points that demonstrate how small alterations in input features affect a model's predictions, enabling users to better understand the inner workings of machine learning models, including which features had the most influence on a specific prediction.[63] Therefore, such applications require counterfactual computation to exhibiting the following: (i) **Interpretability and Transparency –** mechanisms for clear and intuitive explanations of model decisions and behaviors using counterfactuals, (ii) **Integration with Visualization Tools –** APIs or frameworks that enable seamless integration of counterfactual computations with existing visual analytic systems, and (iii) **Performance and Scalability**– efficient computation of counterfactuals, ensuring responsiveness even with large and complex models.

Visual Data Splitting and Cleaning For developers, data splitting and cleaning are crucial in evaluating and improving the performance of machine learning and statistics models.[65] Moreover, previous research has found that visualization can provide benefits during this process.[66] This task requires examining the entirety of, random portions of, or specific data samples from a dataset, and selecting points of interest (i.e. the S_{INCL} subset) through explicit criteria.[67] Under this scenario, counterfactual computation should exhibit: (i) **Flexible and Accurate Subset Identification –** accurate and efficient operators to filter and compute counterfactual subsets based on different user interests, such as missing values, outliers, or feature quality, (ii) **Subset Comparisons –** enable comparison of distributions and statistics between the S_{INCL} and S_{CF} subsets based on various criteria such as data integrity, anomalies, and feature quality, and (iii) **Subset Modification**– enable users to create, modify, and combine different data subsets for more fine-grained splitting and cleaning.

Guided Exploratory Data Analysis Existing studies suggest that counterfactual information can help people reason about data within visual analytics systems.[6,20,21] However, relatively little work has focused on the use of counterfactuals as the basis for offering guidance to users performing exploratory data analysis. For example, counterfactual information could serve as an artificial agent[3] to inform users' exploration strategy and help support decision-making. By providing a unified framework for generating counterfactual scenarios, a counterfactual computation model could reduce the development requirements in such systems. It would also help developers easily integrate counterfactual reasoning into their existing visualization systems without requiring extensive domain knowledge. Specifically, the system should be able to support (i) **Exploratory Workflows –** enable the creation of a complete pipeline to direct analysts in exploratory data analysis, from data selection to counterfactual visualization, for generating and managing counterfactual scenarios, (ii) **User-Friendly Explorations –** utilize counterfactual information incorporating various different measures to help explain current exploration results to users in refining their exploration strategies and support more informed decision-making, and (iii) **Domain-Neutral Applications –** enable analysts to apply counterfactuals across various visualization systems and domains without extensive domain-specific knowledge.

Visualization Recommendation Counterfactuals have been used within NLP algorithms[18] and recommender systems,[68] to improve explainability and model performance. However, existing visualization recommendation algorithms, have given limited attention to the counterfactual approach. For example, prior research has employed existing design theories,[69] behavioral models,[70] or content-based insights[71] as the basis for recommendations. Counterfactuals could be useful to help find causal relations between different data variables,[6,16] and therefore visualizations revealing these relations. A counterfactual computation model could lower the barrier for using counterfactuals within such visualization recommendation systems, and should support: (i) **Diverse Recommendation Space –** provide visualizations to expand the potential recommendation space, (ii) **Seamless Integration –** enable existing visualization recommendation toolkits to incorporate counterfactual computation with little effort, and (iii) **User-Driven Recommendations –** enable analysts to influence the generation of visualization recommendations based on specific measurements or preferences.

Design goals

Based in part on the considerations introduced in the above usage scenarios, we derived a set of practical design goals to help shape the design of the operator-based counterfactual model presented in The Co-op Model. These include efficiency, transparency, comprehensibility, flexibility, and compatibility.

At the same time, the motivation to follow an operator-based approach in the first place is informed by previous studies[47,48,72] which demonstrate the benefits of mathematical and algebraic frameworks for efficiently representing complex and diverse concepts in a formal way that can be translated to practical implementations for developers. The operator-based model that we propose provides a unified way of representing a diverse set of operators and measures that can support core design considerations such as **Subset Modification** and **Exploratory Workflows**.

Efficiency: The proposed model should be scalable and efficient to handle large and complex datasets.[73] It supports the consideration of **Performance and Scalability** and **Flexible and Accurate Subset Identification**.

Transparency: The proposed model should be transparent and explainable to show the processes of underlying mechanisms,[74] that is, how counterfactual subsets are generated and why they are relevant or interesting. It is crucial to enable **Interpretability and Transparency** and **User-Friendly Explorations**.

Comprehensibility: The proposed model should be easy to understand and use by developers with different levels of expertise and background knowledge.[75] Good comprehensibility can benefit building

Domain-Neutral Applications and reduce the knowledge requirement to guide **User-Driven Recommendations**.

Flexibility: The proposed model should be generalizable and flexible enough to handle different types of data and visualization tasks.[76] High flexibility could be able to support **Subset Comparisons** and **Diverse Recommendation Space**.

Compatibility: The proposed model should be compatible and interoperable with existing visualization tools and frameworks.[77] This is necessary for **Integration with Visualization Tools** for visual analytics systems and **Seamless Integration** for visualization recommendations.

The Co-op model

This section introduces *Co-op*, an operator-based model for counterfactuals that is designed for exploratory visual analytics workflows. Building upon the set-based concepts introduced in the previous section, *Co-op* formalizes the computational model for counterfactuals using two broad categories of algorithmic building blocks: *measures* and *operators*. Within the *Co-op* model, operators represent logical units that take one or more sets as input and return one or more sets as output. Measures, in contrast, represent functions which map input data points and/or sets to a scalar value. For some operators, measures can be provided as additional inputs to control how the operator performs. The model described in this section provides a formal framework for the general-purpose counterfactual software library described in Co-op as A Library. Table 1 summarizes all symbols and notations.

Table 1. A summary of the notation used throughout this paper.

Symbol	Definition
S_{ALL}	The overall dataset
S_{INCL}	The included subset
S_{EXCL}	The excluded subset
S_{CF}	The counterfactual subset
S_{REM}	The remaining subset
S, S_i	An arbitrary set of data points
x, p, q	Individual data points
$pd(\cdots)$	A point-distance P2P measure
$pd_w(\cdots)$	A weighted point-distance P2P measure
$pd_{avg}(\cdots)$	An average-distance P2S measure
$pd_{wavg}(\cdots)$	A weighted average-distance P2S measure
$sd(\cdots)$	A subset-distance S2S measure
$sd_w(\cdots)$	A weighted subset-distance S2S measure
$sd_{ent}(\cdots)$	An entropy-based S2S measure
$sd_{went}(\cdots)$	A weighted entropy-based S2S measure
$variance(\cdots)$	An intra-subset variance measure
$variance_w(\cdots)$	A weighted intra-subset variance measure
$entropy(\cdots)$	An intra-subset entropy measure
$entropy_w(\cdots)$	A weighted intra-subset entropy measure
M_d	A distance function, for example, *euclidean* or *hamming*
weights	A weight vector for specifying variable importance
$Filter(\cdots)$	Filter operator
$GroupBy(\cdots)$	GroupBy operator
$Counterfactual(\cdots)$	Counterfactual operator
$Resize(\cdots)$	Resize operator
C, Dim	Filter constraints, a data dimension
Measures	Vector of measures for similarity for use within a counterfactual operator

Measures

As described in sec-subsets, the identification of the counterfactual subset S_{CF} requires the identification of the data points in S_{EXCL} that are most similar to S_{INCL}. This process can involve a variety of different calculations which derive scalar values from different forms of data. These quantities can be used for different purposes such as similarity assessment (e.g. "how similar is a data point to a given set"), thresholding (e.g. "which data points are considered *most* similar"), quality assessment (e.g. "how similar is one subset to another"), and more. Specifically, in this model, the similarity is computed by calculating or accumulating the distance between different combinations of points and sets, for example, equations (5) and (7).

The *Co-op* model supports these types of calculations with *measures*. At its most generic, a measure is a function of some input that resolves to a scalar value. Different types of measures can be used to quantify various properties of the input data, and measures can be broadly classified based on the types of input to which they can be applied. *Co-op* specifies four categories of measures including point-to-point, point-to-subset, subset-to-subset, and intra-subset. This section defines these measurement classes and provides examples of common measures in each category.

Point-to-Point Measures Point-to-point (P2P) measures describe relations between two individual data points. In this way, P2P measures represent the finest granularity of measurement in the *Co-op* model. Two specific variants of P2P measures are included in the model: *point distance* and *weighted point distance*.

Point distance: Similarity assessments are often framed as a distance calculation between pairs of points. Reflecting this basic framing of data similarity, the first P2P measure in *Co-op* is a *point distance* measure, noted as *pd*. This measure evaluates the distance between two data points p and q using a given distance function M_d as follows:

$$pd(p, q, M_d) = M_d(p, q), \\ M_d \in \{custom, euclidean, hamming, \ldots\} \quad (5)$$

This basic pd measure is made flexible and extensible by abstracting the distance function M_d from the overall measure. The generic pd can be used to note the distance calculation as part of larger expressions that use the model's notation independent of the underlying details of the specific distance function M_d, which instead would be provided to pd as an additional parameter. M_d could be any pointwise distance formulation.

For example, the reference software implementation of the *Co-op* model is described in as A Library. In that *Co-op* software library, the default for M_d is the commonly used Euclidean distance.[78] Beyond that default, the *Co-op* library provides an extensive list of other distance metrics reflecting those supported by the Scipy Python library.[79] This includes other well-known distances such as the Hamming distance[80] and the Mahalanobis distance.[81] For applications that require it, developers can extend beyond these pre-defined options by implementing a custom M_d function.

In the *Co-op* library, pd is the default P2P similarity measure in *Co-op*, and Euclidean distance is the default option for M_d.

Weighted point distance: In many real-world usage scenarios, the impact and significance of different dimensions and variables can vary widely.[61] Moreover, this variance can occur by application, analysis task/topic, or time.

For example, in a survey dataset about a large population of people, an analysis of financial status would likely place more importance on variables related to income when compared to variables about weight or height. In such a scenario, measurements of similarity would want to give more weight to financial variables when determining which people can serve as counterfactual examples. An analysis of health outcomes from the same data, however, would likely result in a very different notion of what it means to be similar.

Reflecting this notion of relative importance assigned to individual variables, the Weighted Point Distance pd_w measure provides an alternative to the default pd. This P2P variant incorporates the concept of a per-variable weight which can be used to shape the underlying distance function's calculations. More specifically, we define the weighted distance measure $pd_w(p, q, weights, M_d)$ which the same arguments as pd with an additional weight vector *weights*. As implemented in the *Co-op* library, this measure enables developers to customize the importance of different data variables based on weight calculated using their own bespoke application-specific logic.

Point-to-Subset Measures Point-to-Subset (P2S) measures describe relations between a single data point x and a set S. This type of measure can, for example, provide an overall assessment of how similar x is to the members of S. We note that this formulation is equally applicable for both $x \in S$ and $x \notin S$. Therefore, a P2S measure can provide data to help guide decisions such as "should x be added to set S" as well as "should x be removed from set S".

The overall assessment can utilize a variety of aggregation functions such as maximum, minimum, or average. Reflecting the most common usage scenario in the counterfactual workflow, measures defined below introduce two variants based on an average distance calculation.

Average distance: We extend the definition of point distance (see equation (5)) to fit the point-to-subset relation. The average distance pd_{avg} is computed as the mean of all pairs of point distances from the target point p to each point in the target subset S:

$$pd_{avg}(x, S, M_d) = \frac{1}{|S|} \sum_{i \in S} pd(x, i, M_d). \quad (6)$$

Weighted average distance: Similarly, the weighted average distance pd_{wavg} is an extension of the point-to-point pd_w measure based on the mean of all point-to-point distances from the target point x to each point in the target subset S. As in the P2P case, this weighted P2S measure enables differentiation in the importance given to individual variables in the distance calculation.

Subset-to-Subset Measure Subset-to-Subset (S2S) measures describe relations between two sets of data points. For example, this type of measure can capture the overall similarity between two subsets of data (e.g. S_{INCL} and S_{CF} as described in Counterfactual subsets). Reflecting common use cases in counterfactual-based workflows, three different S2S measures are defined below.

Subset distance: The most basic S2S measure, subset distance, is an extension of the original point distance measure defined in equation (5). For two data subset S_1 and S_2, the subset distance sd can be expressed as:

$$sd(S_1, S_2, M_d) = \frac{1}{|S_1||S_2|} \sum_{i \in S_1} \sum_{j \in S_2} pd(i, j, M_d). \quad (7)$$

Connecting this formulation with prior work, this measure is mathematically equivalent to the set similarity measure adopted in the *CoFact* system[6] except for the normalization factor which makes the distance comparable across subsets of different sizes.

Entropy-based similarity: An alternative S2S similarity measure builds on the information theory concept of entropy, commonly used in many machine learning and statistical applications. Following the canonical entropy formulation proposed by Shannon,[82] we first define the entropy of a given subset S as follows:

$$entropy(S) = -\sum_{i=1}^{N} P(x_i) \cdot log(P(x_i)), \tag{8}$$

where $P(x)$ is the probability of x given the distribution of subset S. Adopting this definition of entropy, we employ the KullbackLeibler divergence[83] as the default entropy-based S2S similarity metric $sd_{entropy}$, which computes the differences in the entropy of two subsets' probability distributions:

$$sd_{ent}(S_1, S_2) = \sum_{i=1}^{N} P_1(x_i) \cdot (log(P_1(x_i)) - log(P_2(x_i))), \tag{9}$$

where $P_i(x)$ is the probability distribution of subset S_i.

Weighted subset-to-subset similarity: As in the P2P and P2S measures, weighted versions of these measures enable the assignment of importance to different variables based on the application context. This leads to a weighted subset distance measure sd_w, and a weighted entropy-based similarity measure sd_{went}.

Intra-Subset Measure Intra-Subset (IS) measures are the final category required for the counterfactual-based approach described in Counterfactual Subset. IS measures describe relations between points within a single subset.

Variance: The variance IS measure is constructed using the traditional variance statistic and captures the expected deviation from the mean for a typical data point x in subset S. Variance is a widely used measurement and has shown utility as an assessment for counterfactual subsets.[84] Generally speaking, counterfactual subsets with lower variance are desired because they reflect a more homogeneous set of data points. The variance measure is defined as follows:

$$variance(S) = \frac{1}{|S|}\sum_{i \in S}\left(i - \frac{1}{|S|}\sum_{j \in S} j\right)^2, \tag{10}$$

$$Relevance = \frac{1}{2}\left(M1 + \sqrt{M1 * M2}\right), \tag{11}$$

$$Subset = \frac{|CF|}{(|CF| + |IN|)}, \tag{12}$$

Entropy: Similar to the S2S measures, we further introduced the entropy IS measure. Compared to variance, entropy has an obvious advantage that its value is independent of the scale of data variables and is only associated with the probability of data points (as shown in equation (8)). Thus it could be easier to capture the impact of variables with smaller value ranges for computing counterfactual subsets. Just as entropy was used as the basis for an S2S measure in Subset-to-Subset Measure, a similar approach can be taken for assessing the disorder within a single set. In fact, the entropy equation introduced in equation (8) (first introduced as a building block for sd_{ent}) is defined as a function of a single subset S. This same equation can be directly used as an IS measure.

Weighted variance and entropy: As with the other measure types, weighted versions of the IS measures can be defined to account for differing importance for each variable given a particular application context. This is reflected in weighted formulations of the variance and entropy measures: $variance_w(S, weights)$ and $entropy_w(S, weights)$, respectively.

Operators

Operators are logical units that take one or more sets as input and return one or more sets as output. Many operators include additional inputs beyond sets, such as measures, which help determine how the operator behaves. Prior mathematical models of visualizations exhibit operators with different capabilities, including operators that apply constraints to produce data subsets,[85] specific operators tailored to achieve the model's core features,[86] and more generic dyadic set operators.[48] In a similar way, we use three general categories of operators that comprise the *Co-op* model: constraint operators, counterfactual operators, and functional operators.

Constraint Operators Constraint operators are used to manipulate the data points included within a given subset based on the inclusion or organizational criteria. These operators would typically be used in response to user interaction to manipulate S_{INCL} such that it reflects the user's analytic focus. Two common constraint operators are defined in this section: *Filter* and *GroupBy*.

Filter: The most basic constraint operator, *Filter*, enables users to apply one or more constraints as inclusion criteria. The *Filter* operator produces as output a revised S_{INCL} based on the new criteria, as well as a corresponding S_{EXCL}. The *Filter* operator can be formally expressed as follows:

$$Filter(C) \Rightarrow \begin{cases} x \in S_{INCL}, & \exists x \in S_{ALL}, x \models C \\ x \in S_{EXCL}, & otherwise \end{cases} \tag{13}$$

$$C = [\{Dim_1, cstr_1\}, \{Dim_2, cstr_2\}, \ldots], \tag{14}$$

where x is a data point, and C is the filter constraints. The constraints consist of dimension names Dim_i and constraint functions $cstr_i$.

GroupBy: Similar to Filter, the *GroupBy* operator manipulates a set of data points based on criteria defined over one or more dimensions. However, rather than filtering to determine included and excluded subsets, the *GroupBy* operator turns a set into one or more subsets (i.e. groups) as determined by the criteria. Each of these can then be processed by further operators to identify per-group counterfactual subsets, for example. The *GroupBy* operator can be expressed as:

$$GroupBy(S, Dim[]) \Rightarrow \begin{cases} x \in S_1, & \exists x \in S_{ALL}, \\ x.Dim \in Dim[0] & \\ x \in S_2, & \exists x \in S_{ALL}, \\ x.Dim \in Dim[1] & \\ \ldots & \end{cases} \tag{15}$$

where S is the initial set to be grouped, Dim is the user-chosen grouping dimension, $Dim[n]$ is the *n-th* grouping value range in Dim, $x.Dim$ is the value in Dim of data point x, and S_i is *i-th* resulting group.

Counterfactual Operators Counterfactual operators are used to identify the points in an excluded subset that best serve as counterfactuals for a given included subset. More formally, counterfactual operators derive a S_{CF} (and corresponding S_{REM}) from excluded subset S_{EXCL} (or a specific subset of S_{EXCL}) to best match a given S_{INCL} according to a given measure. There are two common operator types within this category: *Counterfactual* and *Resize*.

Counterfactual: The *Counterfactual* operator is the most critical in the *Co-op* model as it is responsible for the core feature of identifying the S_{CF} subset for a given S_{INCL} by selecting data points from the excluded subset S_{EXCL}. The output of the *Counterfactual* operator is a new counterfactual group S_{CF} and the corresponding remainder S_{REM}. As introduced in Subset Filtering, Counterfactual *Counterfactual* operates using a combination of similarity measures which can be configured to control the behavior of this operation. Formally, we note the *Counterfactual* operator as follows:

$$\begin{aligned} &Counterfactual(S_{INCL}, S_{EXCL}, Measures) \\ &\Rightarrow S_{CF} + S_{REM}, \end{aligned} \tag{16}$$

$$Measures = [\{w_1, M_{d1}\}, \{w_2, M_{d2}\}, \ldots], \tag{17}$$

where *Measures* is the input similarity measure vector, consisting of one or multiple similarity measures M_{di} and (where appropriate) corresponding weights w.

Resize: The *Resize* operator enables users to modify an existing counterfactual subset by increasing or decreasing the number of data points. For example, when a small S_{CF} is produced by a *Counterfactual* operator, the *Resize* operator can be used to adjust the threshold such that more data points are included. The change in size can be driven by the original measures used to create the S_{CF} or by a new set of measures used specifically for the resize process. The *Resize* operator is defined as follows:

$$\begin{aligned} &Resize(S_{CF}, S_{INCL}, S_{REM}, Size, Measures) \\ &\Rightarrow S'_{CF} + S'_{REM}, \end{aligned} \tag{18}$$

where *Size* is the anticipated number of data points in the new S_{CF} subset; *Measures* is optional; and S'_{CF} and S'_{REM} are the updated CF and REM subset.

Functional Operators The final group of operators are functional operators. These are designed to support dyadic set operations[87] and provide basic set manipulation capabilities. Previous studies have demonstrated the effectiveness of dyadic operations in visual analytic system design,[48] and the concept of sets is a foundational element in the counterfactual workflow. The *Co-op* model therefore adopts the three most common dyadic set operations: union, intersection, and difference. Other set operators, such as complement or symmetric difference, can be constructed as a combination of these as needed.

Union: The *Union* of two subsets S_1 and S_2 is the set of all those elements which are in S_1, S_2, or both:

$$Union(S_1, S_2, Dims) = S_1 \cup S_2, \tag{19}$$

where *Dims* is the vector of target dimension names to be employed to compute the union set.

Intersection: The *Intersection* of two subsets S_1 and S_2 is the set of all elements which are in both S_1 and S_2:

$$Intersection(S_1, S_2, Dims) = S_1 \cap S_2. \tag{20}$$

Difference: The *Difference* of two subsets S_1 and S_2 is the set of all those elements which are in S_1 but not in S_2:

$$Difference(S_1, S_2, Dims) = S_1 - (S_1 \cap S_2). \tag{21}$$

Co-op as a library

The *Co-op* model of measures and operators provides a formal foundation for counterfactual-based workflows within visual analytics systems. Building on this model, we have developed a general-purpose software library for counterfactual-based exploratory visual analysis.

Implementation

Given the widespread use of Python for statistical computing and data science, the initial core version of the *Co-op* library has been implemented as a Python library and is available as open-source software. The library implements all measures and operators following algorithms described in The Co-op Model-operators and provides an extensible API that enables the easy integration of custom components such as special measures for a given application domain. The library builds on the widely-used NumPy[88] and SciPy[79] packages for measures, along with Pandas[89] operators. The default input data structure is the Pandas DataFrame. The *Co-op* library is organized around two packages: measure and operator. These packages are further organized as modules that reflect the groupings outlined in The Co-op Model (P2P, P2S, S2S, and IS measures; constraint, counterfactual, and functional operators). The source code is available at https://github.com/VACLab/Co-op.

Parameter settings

Defaults are provided for all measures and operators as suggested by a previous study[6] and other empirical guidance of measure computation.[4] See Choosing proper parameter settings for more detailed discussions and implications of parameter settings. However, developers can modify the default behaviors by implementing their own components. In particular, measures and distance functions are abstracted from the design as outlined in the model. This enables custom new parameters of measure or distance implementations which can be "plugged in" to the library to customize the behavior for a given application. When adding plugins, developers can first implement new measures in the measures package, which should output a condensed distance matrix, like pdist from SciPy,[79] and then simply input the desired new measure or measures into the *Measures* parameter of the *Counterfactual* operator from the operators package as introduced in Counterfactual Operators.

Demonstration

The *Co-op* model and corresponding library have several potential uses, as outlined in the previous section. However, the motivation for developing this approach and the primary usage scenario for the library is in support of counterfactual visualizations within the exploratory visual analysis process as a mechanism for improving users' inferences from visualized data. To demonstrate how the *Co-op* model provides a flexible set of general-purpose capabilities, this section describes the workflows of two visual analytics systems built using the *Co-op* library, along with expert interviews regarding the library.

Re-implementing CoFact

As a first demonstration of utility, we used the *Co-op* library to re-implement an existing counterfactual-guided visual analytics system that had been created as a bespoke application. For this effort, we re-implemented *CoFact*[6] and its counterfactual-based workflow for improving the accuracy of users' causal inferences from charts.

Workflow The original *CoFact* enables analyses of numerical and categorical feature outcomes. Users of the system perform variable filtering (e.g. Houses where $YearConstructed < 2015$) via user interactions to identify subsets of interest. The system then computes counterfactual subsets based on an Euclidean distance measure.

Design Space To faithfully support the same functionality in our re-implementation, we also support categorical and continuous data types in our re-design. Using *Co-op*, constraint operators support the filtering process while the counterfactual operator is used to compute the counterfactual subset. For the *Counterfactual* operator, we selected the P2S distance as the measure type of our implementation, using the default Euclidean distance just as described in the original *CoFact* paper. This approach resulted in essentially a feature-by-feature replication of the original system design.

CoExplorer: using additional Co-op features

The re-implemented *CoFact* system, like the original design, uses only a small portion of *Co-op*'s expressive power. To help validate a broader range of features of the library, we developed a new exploratory visual analysis platform called *CoExplorer* which leverages source code from the Voyager system.[8,90]

Workflow While enabling all functionalities in *CoFact*, the workflow of *CoExplorer* enables more exploratory processes. Users are required to perform variable filtering to get the S_{INCL} (top view in Figure 2) at first, meanwhile, *CoExplorer* will show the corresponding visualizations of S_{CF} (bottom left view) and S_{EXCL} (bottom right view). Other than the default measures from *CoFact*, users can further specify supported distance and measures from *Co-op* to compute the S_{CF}. For more exploration, users can click S_{CF} to replace the S_{INCL} to start a new round of analysis and can iteratively add new filter constraints to explore more insights within S_{CF}.

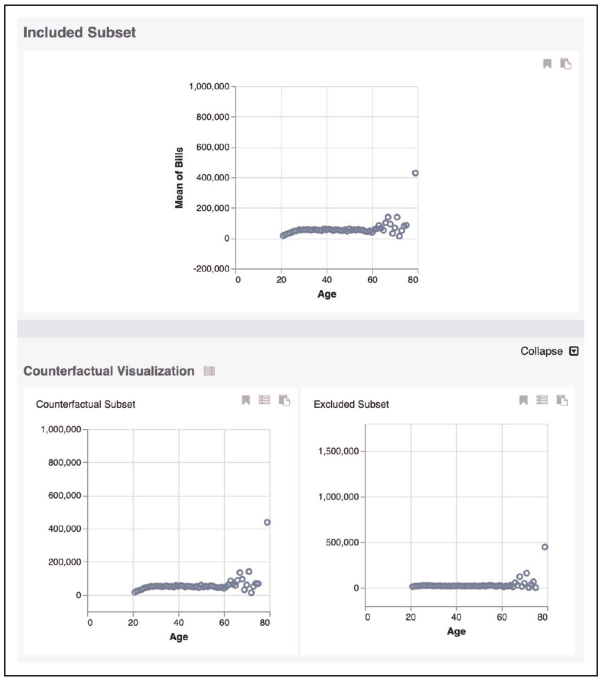

Figure 2. A screenshot from *CoExplorer*. The top panel shows S_{INCL}, while the bottom left and right show S_{CF} and S_{EXCL}, respectively. Users can select S_{CF} as the new included subset to start a new round of analysis.

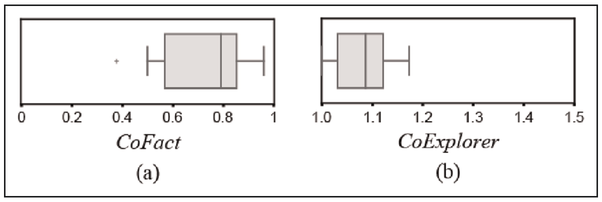

Figure 3. (a) When comparing computation times between *Co-op* and the previously published *CoFact* system, *Co-op* was faster as evidenced by relative ratios below one. (b) In the new *CoExplorer* prototype, comparing computation times with and without *Co-op*'s counterfactual computation enabled shows a relatively small performance overhead for that capability.

Design Space Existing exploratory visual analysis systems enable users to conduct a variety of data analyses and interactions with heterogeneous data types.[91] The *CoExplorer* design therefore supports categorical, continuous, and time-series data. *CoExplorer* employs both distance and entropy-based similarity measures to compute data subsets using the counterfactual operator. Figure 2 shows a counterfactual view within the *CoExplorer* system. This is combined with a filtering panel (not shown) that adopts the design found in the original Voyager.[8]

Performance analysis

We present the results of a preliminary performance analysis of the two prototype systems.

CoFact Analysis We evaluated the performance of computing counterfactuals between the original *CoFact*'s algorithm (from the CoFact source code) and the new *Co-op*-supported approach. We tested the system's performance on 12 high-dimensional datasets applied in the previous study6, such as *House Price* and *College Majors* datasets. To normalize performance times across datasets of different sizes and complexity in our performance analysis, we computed the relative ratio of time required by the two systems (original CoFact, and our re-implementation). The relative ratio is defined as $\frac{Time(Reimplemented\ CoFact)}{Time(Original\ CoFact)}$ for each dataset. In this formulation, a smaller ratio represents better performance (lower time) when using the *Co-op* library.

Figure 3(a) illustrates the relative ratios for computation time when creating counterfactual subsets within the original *CoFact* and the *Co-op*-supported re-implementation of *CoFact*. The overall results show that the average relative ratio is about 0.79, which means *Co-op* achieves 20% better performance on average in computing counterfactuals compared to the original *CoFact*'s computation method.

CoExplorer Analysis To better understand the time required to compute counterfactual subsets using *Co-op*, we compared the performance of *CoExplorer* in two modes: (a) with counterfactual subset calculations and (b) without. Using the same datasets used in the performance evaluation for *CoFact* (see section), we employed a similar relative ratio measure to evaluate the computational resources required in the two modes.

Figure 3(b) shows the resulting relative ratios across the 12 datasets with and without counterfactual computations. The data shows that including the counterfactual computations using *Co-op* introduces about 9% additional compute time on average, and no $>$20% for the tested datasets.

Limitations of Performance Analysis While the computational overhead in these examples shows a relatively modest overhead in compute time, however, since our demonstrations mainly focused on showing the functionality of *Co-op*, this preliminary-level analysis is inadequate to provide further insights into the source of the performance advantage or deficit. Therefore, it is not clear how these delays in responsiveness would impact users' experience in analysis tasks and broader exploratory usage scenarios and whether the performance gain is due to which specific measures or operators in the model. In addition, the compute time is dependent on dataset size as well as available computational resources. A deeper performance analysis that more systematically studies these parameters is an important topic for future work.

Additionally, existing user interfaces provided advanced abilities such as layered or faceted views,[7] future work should focus on employing *Co-op* to those advanced functionalities.

Expert interviews

To demonstrate the usability of *Co-op* to visualization developers, we conducted 30-min qualitative interviews with six experts. These experts comprised three visualization researchers (denoted as R1–R3) and three data visualization engineers (denoted as E1–E3). R1 and R2 had prior experience in creating counterfactual visual analytics systems.

We first provided a brief introduction of counterfactual visualizations using examples from prior studies.[6,22,92] Then we asked them to think about what their requirements would be to incorporate a counterfactual library into their own visualization systems. Further, we introduced and provided the details and workflows of our model and prototypes, and asked for their opinions on the benefits and limitations of this model, and to what extent it would meet their demands.

Here we revisit the set of design goals (presented in Design Goals) after considering the two prototype systems and experts' responses.

Efficiency: All experts agreed that the library in Python would be able to provide higher computational capability through efficient Python statistical libraries in computational tasks. R1 and R2 specifically talked about the drawback in JavaScript that they needed to implement distance and subset computations manually when using counterfactuals in their systems. E1 appreciated that we primarily provided a Python library which is the most common for analysts. Our preliminary performance analyses (see Performance Analysis) also suggest that the computational efficiency of *Co-op* outperformed an existing algorithm[6] and added relatively little overhead to a non-counterfactual-based system. However, further studies of performance and issues of scale are still needed and the computational capacity at scale must be further studied to improve efficiency.

Transparency: R1 and R3 agreed that the mathematical formulations of the model and the open-source software for the library make the operation of the software quite transparent to developers and system designers. R2 said the JavaScript appendix would be able to help people without Python knowledge see how the model computes counterfactuals. In addition, E3 pointed out concerns about entropy which they felt was a concept that might not be general knowledge for many data visualization software engineers. This implies that the entropy measure may require additional effort to learn before it can be used effectively.

Comprehensibility: The operator-based model that underpins the *Co-op* library provides a relatively logical organization of the library's capabilities and behaviors. All three engineers agreed that the provided examples make it easy to understand counterfactuals, and the definitions of measures and operators should be easy for people with an engineering background to understand. However, this paper did not focus on evaluating human cognition of counterfactual visualizations or developer understanding of the library's API. Additional studies are required to help answer these questions.

Flexibility: The proposed prototypes demonstrate that *Co-op* can work with multiple kinds of data (e.g. categorical, continuous, time-series), visualization types, and exploratory workflows. However, E2 mentioned that presenting counterfactuals in spatial data may provide great value for causal inference as well, providing an example that viewing counterfactuals may help easily find the poison pump in John Snow's broad street pump map of the 1854 cholera outbreak,[93] however *Co-op* currently does not directly support spatial data.

Compatibility: The *CoExplorer* prototype shows that *Co-op* can be integrated with pre-existing code from the Voyager system.[8] In addition, we used *Co-op* to implement the *CoFact* system.[6] These examples show that the library is interoperable in a practical way to provide capabilities for a variety of existing systems. R1 appreciated our reimplementation of *CoFact* which maintained the functionalities while significantly reducing the programing workload of creating counterfactual-supported visual systems. E3 mentioned that counterfactual computations in *Co-op* can be regarded as an advanced step of data filtering, thus the computation flow is obviously generalizable and compatible with most visual analytics systems.

Lowering Implementation Barriers The experts offered insights into how *Co-op* can help overcome or mitigate implementation barriers when creating new counterfactual-based visualization systems. First, providing robust implementations of core capabilities within an open-source library naturally makes technologies easier to use by eliminating the need for others to re-implement algorithms or design new computational models. For this reason, leveraging reliable libraries where they exist is a recognized best practice.[94] All experts agreed that *Co-op* can ease the implementation burden for developers who want to add counterfactual computations into visualizations, since they would not need to learn the details of complex counterfactual concepts or design a new calculation pipeline. Further, data visualization engineers E1-E3 concurred that the functions that comprise the API for *Co-op* are simple and easy to understand, even for junior-level developers. Specifically, R1, based on her own experience working without the benefits of *Co-op*, pointed out that she spent a long time determining parameters and measures when computing counterfactuals. In contrast, *Co-op* offers a range of choices of established measures that can typically provide good results and adhere to best practices, often using default parameters. However, R1 also mentioned that there is no magic solution for all situations. That is to say, to achieve the best comparison of counterfactual subsets, developers still need to understand and compare the parameters across different measures. This suggests that additional work to better characterize how subset distributions are impacted by different parameter settings could be a valuable addition in the future development of *Co-op*. Another concern was raised from R2 that *Co-op* did not provide a visualization configuration panel, which could make it more difficult for developers to easily compare the results of different parameters.

Discussion

The operator-based model and corresponding *Co-op* library provide application developers with a rich set of capabilities for developing counterfactual-based visual analytics systems. As shown by Demonstration, the library can be leveraged in different ways to support a variety of user experiences. This section presents some reflections regarding the design considerations outlined earlier in this paper, discusses implications for system developers, highlights key limitations, and identifies opportunities for future work.

General discussion

Co-op and visualization As discussed in the expert interviews, *Co-op* abstracts the complexities of implementing counterfactual computations, which in turn enables visualization developers to focus on design and user experience rather than low-level technical details. In this way *Co-op* can accelerate visualization development and ease the integration of counterfactual-based capabilities in a manner similar to other libraries designed to support the development of visualization software. Its open-source nature encourages collaboration and engagement with real-world use cases. This can help in soliciting feedback from the community to advance its development, including the contribution of new operators, sharing best practices, and collectively advancing its capabilities.[94]

Choosing proper parameter settings To maintain the structures and provide proper framing of S_{CF} subsets, it can be useful to set up some restrictions on computing counterfactuals. Existing evidence showing that simpler counterfactual explanations that are comparable with input data might lead to easier understanding for users[95] Thus we recommend developers select the default size of S_{CF} to be equal to S_{INCL}. However, for cases where S_{INCL}‘s size approaches the size of S_{EXCL}, this becomes problematic. Therefore, the size of S_{CF} should generally not grow to be larger than half the size of S_{EXCL}. This methodology is also seen in prior work.[6]

Prior studies also suggest that counterfactual sets should have low variance to minimize bias.[96] Other related prior work focused on financial and commodity applications also suggests benefits from lower entropy.[97] We therefore suggest using measures of variance and entropy to validate the computed S_{CF} subsets following the above guidelines. Our goal is to minimize the similarity metrics while maintaining the major counterfactual structures by providing low variance and entropy. Empirically, we recommend developers initially use the lower bound of the interquartile range (IQR)[4] of variance and entropy among the data subsets as the default threshold settings for computing counterfactual subsets, as this has been demonstrated to be effective in existing quality metrics.[40,41]

Adding interactions with Co-op-supported systems Interaction plays a fundamental role in visualization and has the potential, when correctly designed, to improve the quality of analytics systems.[98] Researchers have defined a number of interaction types in information visualization.[98] Though the prototypes in this paper exercise only a small number of examples of possible interactions, *CoExplorer* can still provide more exploratory ability than *CoFact* by simply clicking on S_{CF} to replace the S_{INCL}. The rich variety of interaction types and design methodologies developed by the visualization community can be used in combination with *Co-op* to enable a wide variety of potential applications.

Fitting with the user's goals, data types, tasks, and preferences In real-world usage scenarios, varying tasks and user demands require different types of outputs and insights[99,100] and result in different levels of comprehension.[11] A task-aware approach to data communication and exploratory visualization design is required to maximize the utility of a given system.[15,35,99] Therefore, the prototype applications in this paper are simply exemplars of possible counterfactual workflows. *Co-op* should be used in the context of designers' specific application objectives to best support user performance.

Limitations

Though the *Co-op* model and library can enable a wide range of counterfactual-based visualization usage scenarios, there remain some limitations that pose challenges. For instance, while the *Co-op* library provides default settings and we recommend certain best practices for creating S_{CF} subsets (see Choosing proper parameter settings), these are based on our empirical observations and other related work. However, there remains little evidence-driven guidance on how to best select the ideal counterfactual subset during exploratory analysis. This remains an important open problem for future study. Moreover, *Co-op* currently cannot process spatial datasets, a common data type that many applications require.

The current model also does not make any attempt to unify the various measures into a comprehensive framework which could be the basis for automation. Usage of *Co-op* as currently described therefore still requires manual configuration and design to yield the best results. Moreover, the prototypes presented as examples of *Co-op*-enabled applications both adopted existing visualization designs without any attempt to improve or optimize how the counterfactual information is presented to users. Therefore this paper does not make suggestions for how to best visualize counterfactuals in an intuitive and effective way.

Finally, while we preliminarily measured the performance of the prototypes, no analysis was done to study how the library works at scale. The performance of the system as datasets grow very large, along with the impacts of potentially slow computations on user experience, remains unstudied.

Future work

In future work, we aim to address a number of the limitations outlined above. First, we aim to extend *Co-op* to support spatial data and the potential for other additional data types. We also hope to conduct broader user studies to help build an evidence base for best practices when designing counterfactual-based visual analysis systems. This includes a better understanding of users' ability to interpret counterfactual information and preferences for how that information is displayed.

Improving computational performance and scalability, as well as providing a richer set of measures and operators for specific contexts is also an important future direction. Related to this goal, we aim to expand and promote the usage of counterfactuals in a wider variety of visualization algorithms and designs, such as using counterfactuals to guide visualization recommendations and data subset selection in dashboard design. By expanding the variety of applications and domains in which these approaches are used, the community will more quickly overcome limitations and learn about best practices.

Conclusion

This paper presented an operator-based model to enable counterfactual-based subset computation in visual analysis. The model included a variety of measures and operators that combine to support a counterfactual workflow which has been shown to improve users' inferences from visualized data. This model was instantiated within the *Co-op* library, an open-source Python library for bringing general counterfactual-based subset computation algorithms to exploratory visualization workflows. *Co-op* can be used by developers to easily incorporate counterfactual workflows into their visual analytics systems, and the library's design enables developers to extend and customize its behavior to meet application needs. The general utility of the model was demonstrated through its use in two prototype systems with different workflows, including both a re-implementation of an existing system and a new exploratory one, and an interview for experts in visualization research and engineering. Informed by these experiences, the paper concluded with a discussion of implications on system design, limitations, and areas for future work.

Funding

The author(s) disclosed receipt of the following financial support for the research, authorship, and/or publication of this article: This work is supported in part by Award #2211845 from the National Science Foundation.

ORCID iDs

Arran Zeyu Wang https://orcid.org/0000-0002-7491-7570
David Borland https://orcid.org/0000-0002-0162-4080
David Gotz https://orcid.org/0000-0002-6424-7374

Supplemental material

Supplemental material for this article is available online.

References

1. Gotz D and Borland D. Data-driven healthcare: challenges and opportunities for interactive visualization. *IEEE Comput Graph Appl* 2016; 36(3): 90–96.
2. Jin Z, Guo S, Chen N, et al. Visual causality analysis of event sequence data. *IEEE Trans Vis Comput Graph* 2021; 27(2): 1343–1352.
3. Monadjemi S, Guo M, Gotz D, et al. Human–computer collaboration for visual analytics: an agent-based framework. *Comput Graph Forum* 2023; 42: 199–210.
4. Tukey J. *Exploratory data analysis*, vol. 2. Addison-Wesley, Reading, 1977.
5. Szafir DA, Borgo R, Chen M, et al. *Visualization psychology*. New York: Springer Nature, 2023.
6. Kaul S, Borland D, Cao N, et al. Improving visualization interpretation using counterfactuals. *IEEE Trans Vis Comput Graph* 2022; 28(1): 998–1008.
7. Guo M, Zhou Z, Gotz D, et al. Grafs: graphical faceted search system to support conceptual understanding in exploratory search. *ACM Trans Interact Intell Syst* 2023; 13(2): 1–36.
8. Wongsuphasawat K, Moritz D, Anand A, et al. Voyager: exploratory analysis via faceted browsing of visualization recommendations. *IEEE Trans Vis Comput Graph* 2016; 22(1): 649–658.
9. Walny J, Frisson C, West M, et al. Data changes everything: challenges and opportunities in data visualization design handoff. *IEEE Trans Vis Comput Graph* 2020; 26(1): 12–22.
10. Zhou Z, Wen X, Wang Y, et al. Modeling and leveraging analytic focus during exploratory visual analysis. In *2021 ACM CHI Conference*, Yokohama, pp.1–15.
11. Quadri G, Wang A, Wang Z, et al. Do you see what i see? a qualitative study eliciting high-level visualization comprehension. In *2024 ACM CHI Conference*, Honolulu, pp.1–26.
12. Gotz D, Wang W, Chen AT, et al. Visualization model validation via inline replication. *Inf Vis* 2019; 18(4): 405–425.
13. Oghbaie M, Pennock M and Rouse W. Understanding the efficacy of interactive visualization for decision making for complex systems. In *2016 Annual IEEE Systems Conference*, Orlando, pp.1–6.
14. Xiong C, Shapiro J, Hullman J, et al. Illusion of causality in visualized data. *IEEE Trans Vis Comput Graph* 2020; 26(1): 853–862.
15. Yen CE, Parameswaran A and Fu W. An exploratory user study of visual causality analysis. *Comput Graph Forum* 2019; 38(3): 173–184.
16. Pearl J. *Causality*. Cambridge: Cambridge University Press, 2009.
17. Kusner M, Loftus J, Russell C, et al. Counterfactual fairness. In *2017 NeurIPS Conference*, Long Beach, p.30.
18. Wu T, Ribeiro M, Heer J, et al. Polyjuice: generating counterfactuals for explaining, evaluating, and improving models. In *2021 ACL Conference*, Bangkok, pp.6707–6723.
19. Kale A, Wu Y and Hullman J. Causal support: modeling causal inferences with visualizations. *IEEE Trans Vis Comput Graph* 2022; 28(1): 1150–1160.
20. Cheng F, Ming Y and Qu H. Dece: decision explorer with counterfactual explanations for machine learning models. *IEEE Trans Vis Comput Graph* 2021; 27(2): 1438–1447.
21. Gomez O, Holter S, Yuan J, et al. Vice: visual counterfactual explanations for machine learning models. In *2020 ACM IUI Conference*, Cagliari, pp.531–535.
22. Wang AZ, Borland D and Gotz D. An empirical study of counterfactual visualization to support visual causal inference. *Inf Vis* 2024; 23(2): 197–214.
23. Borland D, Wang AZ, Gotz D, et al. Using counterfactuals to improve causal inferences from visualizations. *IEEE Comput Graph Appl* 2024; 44(1): 95–104.
24. Angrist JD, Imbens GW and Rubin DB. Identification of causal effects using instrumental variables. *J Am Stat Assoc* 1996; 91(434): 444–455.
25. Johansson F, Shalit U and Sontag D. Learning representations for counterfactual inference. In *2016 ICML Conference*, New York, pp.3020–3029.
26. Stuart EA. Matching methods for causal inference: a review and a look forward. *Stat Sci* 2010; 25(1): 1.
27. Caliendo M and Kopeinig S. Some practical guidance for the implementation of propensity score matching. *J Econ Surv* 2008; 22(1): 31–72.
28. Iacus SM, King G and Porro G. Causal inference without balance checking: coarsened exact matching. *Polit Anal* 2012; 20(1): 1–24.
29. Glymour M, Pearl J and Jewell NP. *Causal inference in statistics: a primer*. New York: John Wiley & Sons, 2016.
30. Zuk T and Carpendale S. Visualization of uncertainty and reasoning. In: *2007 Smart Graphics Symposium*, Kyoto, pp.164–177.
31. Wang J and Mueller K. The visual causality analyst: an interactive interface for causal reasoning. *IEEE Trans Vis Comput Graph* 2016; 22(1): 230–239.
32. Wang J and Mueller K. Visual causality analysis made practical. In *2017 IEEE VAST Conference*, Phoenix, pp.151–161.
33. Hoque MN and Mueller K. Outcome-explorer: a causality guided interactive visual interface for interpretable algorithmic decision making. *IEEE Trans Vis Comput Graph* 2022; 28(12): 4728–4740.
34. Guo G, Karavani E, Endert A, et al. Causalvis: visualizations for causal inference. In *2023 SIGCHI Conference on Human Factors in Computing Systems*, Hamburg, pp.1–20.
35. Xie X, Du F and Wu Y. A visual analytics approach for exploratory causal analysis: exploration, validation, and applications. *IEEE Trans Vis Comput Graph* 2021; 27(2): 1448–1458.
36. Deng Z, Weng D, Xie X, et al. Compass: towards better causal analysis of urban time series. *IEEE Trans Vis Comput Graph* 2022; 28(1): 1051–1061.
37. Hotz I, Bujack R, Garth C, et al. Mathematical foundations in visualization. In: Chen M, Hauser H, Rheingans P and Scheuermann G (eds.) *Foundations of data visualization*. Cham: Springer, 2020, pp.87–119.
38. Rodrigues FA. Network centrality: an introduction. In: Macau E (ed.) *A mathematical modeling approach from*

nonlinear dynamics to complex systems. Cham: Springer, 2019, pp.177–196.
39. Friedkin NE. Theoretical foundations for centrality measures. *Am J Sociol* 1991; 96(6): 1478–1504.
40. Wilkinson L, Anand A and Grossman R. Graph-theoretic scagnostics. In *2005 IEEE Symposium on Information Visualization*, Minneapolis, pp.21–21.
41. Wang Y, Wang Z, Liu T, et al. Improving the robustness of scagnostics. *IEEE Trans Vis Comput Graph* 2020; 26(1): 759–769.
42. Abdi H and Williams LJ. Principal component analysis. *Wiley Interdiscip Rev Comput Stat* 2010; 2(4): 433–459.
43. Van der Maaten L and Hinton G. Visualizing data using t-SNE. *J Mach Learn Res* 2008; 9(11): 2579–2605.
44. Kim Y, Walls L, Krafft P, et al. A Bayesian cognition approach to improve data visualization. In *2019 ACM CHI Conference*, Glasgow, pp.1–14.
45. Gabry J, Simpson D, Vehtari A, et al. Visualization in Bayesian workflow. *J R Stat Soc Ser A Stat Soc* 2019; 182(2): 389–402.
46. Chen M and Jänicke H. An information-theoretic framework for visualization. *IEEE Trans Vis Comput Graph* 2010; 16(6): 1206–1215.
47. Kindlmann G and Scheidegger C. An algebraic process for visualization design. *IEEE Trans Vis Comput Graph* 2014; 20(12): 2181–2190.
48. Wu A, Tong W, Li H, et al. Computableviz: mathematical operators as a formalism for visualisation processing and analysis. In *2022 ACM CHI Conference*, New Orleans, pp.1–15.
49. John G, Kohavi R and Pfleger K. Irrelevant features and the subset selection problem. In *1994 ICML Conference*, New Brunswick, pp.121–129.
50. Killamsetty K, Sivasubramanian D, Ramakrishnan G, et al. Glister: generalization based data subset selection for efficient and robust learning. In *2021 AAAI Conference*, Vancouver, pp.8110–8118.
51. Kaushal V, Iyer R, Kothawade S, et al. Learning from less data: a unified data subset selection and active learning framework for computer vision. In *2019 IEEE WACV Conference*, Waikoloa Village, pp.1289–1299.
52. Gray J, Chaudhuri S, Bosworth A, et al. Data cube: a relational aggregation operator generalizing group-by, cross-tab, and sub-totals. *Data Min Knowl Discov* 1997; 1: 29–53.
53. Shneiderman B. The eyes have it: a task by data type taxonomy for information visualizations. In: *1996 IEEE Symposium on Visual Languages*, Boulder, pp.336–343.
54. Gotz D, Sun S and Cao N. Adaptive contextualization: combating bias during high-dimensional visualization and data selection. In *2016 ACM IUI Conference*, Sonoma, pp.85–95.
55. Borland D, Wang W, Zhang J, et al. Selection bias tracking and detailed subset comparison for high-dimensional data. *IEEE Trans Vis Comput Graph* 2020; 26(1): 429–439.
56. Borland D, Wang W and Gotz D. Contextual visualization. *IEEE Comput Graph Appl* 2018; 38(6): 17–23.
57. Borland D, Zhang J, Kaul S, et al. Selection-bias-corrected visualization via dynamic reweighting. *IEEE Trans Vis Comput Graph* 2021; 27(2): 1481–1491.
58. Gotz D, Zhang J, Wang W, et al. Visual analysis of high-dimensional event sequence data via dynamic hierarchical aggregation. *IEEE Trans Vis Comput Graph* 2020; 26(1): 440–450.
59. Lee D, Dev H, Hu H, et al. Avoiding drill-down fallacies with vispilot: assisted exploration of data subsets. In *2019 ACM IUI Conference*, Marina del Ray, pp.186–196.
60. Zhang Z, Gotz D and Perer A. Iterative cohort analysis and exploration. *Inf Vis* 2015; 14(4): 289–307.
61. Tang B, Han S, Yiu M, et al. Extracting top-k insights from multi-dimensional data. In *2017 ACM SIGMOD Conference*, Chicago, pp.1509–1524.
62. Hullman J and Gelman A. Designing for interactive exploratory data analysis requires theories of graphical inference. *Harv Data Sci Rev* 2021; 3(3): 3.
63. Wexler J, Pushkarna M, Bolukbasi T, et al. The what-if tool: interactive probing of machine learning models. *IEEE Trans Vis Comput Graph* 2020; 26(1): 56–65.
64. Ciorna V, Melançon G, Petry F, et al. Interact: a visual what-if analysis tool for virtual product design. *Inf Vis* 2024; 23(2): 123–141.
65. Faraway JJ. Does data splitting improve prediction? *Stat Comput* 2016; 26: 49–60.
66. Krishnan S, Franklin M, Goldberg K, et al. Activeclean: an interactive data cleaning framework for modern machine learning. In *2016 SIGMOD Conference*, San Francisco, pp.2117–2120.
67. Chu X, Ilyas I, Krishnan S, et al. Data cleaning: overview and emerging challenges. In *2016 SIGMOD Conference*, San Francisco, pp.2201–2206.
68. Dong Z, Zhu H, Cheng P, et al. Counterfactual learning for recommender system. In *2020 ACM RecSys Conference*, Rio de Janeiro, pp.568–569.
69. Moritz D, Wang C, Nelson GL, et al. Formalizing visualization design knowledge as constraints: actionable and extensible models in Draco. *IEEE Trans Vis Comput Graph* 2018; 25(1): 438–448.
70. Gotz D and Wen Z. Behavior-driven visualization recommendation. In *2009 ACM IUI Conference*, Sanibel Island, pp.315–324.
71. Zhou Z, Wang W, Guo M, et al. A design space for surfacing content recommendations in visual analytic platforms. *IEEE Trans Vis Comput Graph* 2023; 29(1): 84–94.
72. Chen M, Grinstein G, Johnson CR, et al. Pathways for theoretical advances in visualization. *IEEE Comput Graph Appl* 2017; 37(4): 103–112.
73. Han D, Pan J, Zhao X, et al. NetV.js: a web-based library for high-efficiency visualization of large-scale graphs and networks. *Vis Inf* 2021; 5(1): 61–66.
74. Ren D, Lee B and Höllerer T. Stardust: accessible and transparent GPU support for information visualization rendering. *Comput Graph Forum* 2017; 36(3): 179–188.
75. Scanniello G, Gravino C, Risi M, et al. Documenting design-pattern instances: a family of experiments on source-code comprehensibility. *ACM Trans Softw Eng Methodol* 2015; 24(3): 1–35.

76. Gonnella G and Kurtz S. Gfapy: a flexible and extensible software library for handling sequence graphs in python. *Bioinformatics* 2017; 33(19): 3094–3095.
77. Nishino R and Loomis S. Cupy: a numpy-compatible library for NVIDIA GPU calculations. *In 2017 LearningSys Workshop at NeurIPS Conference*, Long Beach, pp.1–7.
78. Danielsson PE. Euclidean distance mapping. *Comput Graph Image Process* 1980; 14(3): 227–248.
79. Virtanen P, Gommers R, Oliphant TE, et al.; SciPy 1.0 Contributors. Author correction: SciPy 1.0: fundamental algorithms for scientific computing in Python. *Nat Methods* 2020; 17(3): 352–272.
80. Hamming RW. Error detecting and error correcting codes. *Bell Syst Tech J* 1950; 29(2): 147–160.
81. De Maesschalck R, Jouan-Rimbaud D and Massart DL. The Mahalanobis distance. *Chemometr Intell Lab Syst* 2000; 50(1): 1–18.
82. Shannon CE. A mathematical theory of communication. *Bell Syst Tech J* 1948; 27(3): 379–423.
83. Csiszar I. I-divergence geometry of probability distributions and minimization problems. *Ann Probab* 1975; 3: 146–158.
84. Wu H and Wang M. Variance regularized counterfactual risk minimization via variational divergence minimization. In *2018 ICML Conference*, Stockholm, pp.5353–5362.
85. Chuah M and Roth S. On the semantics of interactive visualizations. In *1996 IEEE Symposium on Information Visualization*, San Francisco, pp.29–36.
86. Chi E and Riedl J. An operator interaction framework for visualization systems. In *1998 IEEE Symposium on Information Visualization*, Research Triangle. pp.63–70.
87. Hall M. *The theory of groups*. New York: Courier Dover Publications, 2018.
88. Harris CR, Millman KJ, van der Walt SJ, et al. Array programming with numpy. *Nature* 2020; 585(7825): 357–362.
89. McKinney W, et al. Pandas: a foundational python library for data analysis and statistics. *PyHPC Workshop* 2011; 14(9): 1–9.
90. Wongsuphasawat K, Qu Z, Moritz D, et al. Voyager 2: augmenting visual analysis with partial view specifications. In *2017 ACM CHI Conference*, Denver, pp.2648–2659.
91. Liu Z and Heer J. The effects of interactive latency on exploratory visual analysis. *IEEE Trans Vis Comput Graph* 2014; 20(12): 2122–2131.
92. Wang A, Borland D and Gotz D. Countering simpsons paradox with counterfactuals. In *2023 IEEE VIS Poster Proceedings*, Melbourne, pp.1–2.
93. Brody H, Rip MR, Vinten-Johansen P, et al. Mapmaking and myth-making in broad street: the London cholera epidemic, 1854. *Lancet* 2000; 356(9223): 64–68.
94. Wilson G, Bryan J, Cranston K, et al. Good enough practices in scientific computing. *PLoS Comput Biol* 2017; 13(6): e100551.
95. Wachter S, Mittelstadt B and Russell C. Counterfactual explanations without opening the black box: automated decisions and the GDPR. *Harv J Law Technol* 2017; 31: 841.
96. Gibson R, Lanctot M, Burch N, et al. Generalized sampling and variance in counterfactual regret minimization. In *2012 AAAI Conference*, Toronto, pp.1355–1361.
97. Benedetto F, Giunta G and Mastroeni L. On the predictability of energy commodity markets by an entropy-based computational method. *Energy Econ* 2016; 54: 302–312.
98. Yi JS, Kang YA, Stasko J, et al. Toward a deeper understanding of the role of interaction in information visualization. *IEEE Trans Vis Comput Graph* 2007; 13(6): 1224–1231.
99. Gotz D and Zhou MX. Characterizing users' visual analytic activity for insight provenance. *Inf Vis* 2009; 8(1): 42–55.
100. Gotz D, Zhou M and Wen Z. A study of information gathering and result processing in intelligence analysis. In *2006 Intelligence Analysis Workshop in ACM IUI Conference*, Sydney, p.6.