Towards a Rule-based Visualization Recommendation System

Arnab Chakrabarti[1], Farhad Ahmad[1] and Christoph Quix[2]
[1]*Information Systems & Databases, RWTH Aachen University, Germany*
[2]*Hochschule Niederrhein, University of Applied Sciences & Fraunhofer FIT, Germany*

Keywords: Data Visualization, Recommendation Systems, Human-in-the-Loop.

Abstract: Data visualization plays an important role in the analysis of data and the identification of insights and characteristics within the dataset. However, visualizing datasets, especially high dimensional ones, is a very difficult and time-consuming process that requires a great deal of manual effort. The automation of data visualization is done in the form of Visualization Recommendation Systems by detecting factors such as data characteristics and user intended tasks in order to recommend useful visualizations. In this paper, we propose a Visualization Recommendation System, built on a knowledge-based rule engine, that takes minimal user input, extracts important data characteristics and supports a large number of visualization techniques depending on both the data characteristics and the intended tasks of the user. Through our proposed model we show the efficacy of such recommendations for users without any domain expertise. Lastly, we evaluate our system with real-world use case scenarios to prove the effectiveness and the feasibility of our approach.

1 INTRODUCTION

The enormous surge in data processing capabilities in today's world has boosted the exponential growth of data that is collected, stored and analyzed to gain valuable insights. This in turn has resulted in the evolution of the processes for Knowledge Discovery in Databases (KDD), which is a well-known procedure to extract patterns and infer knowledge from raw data. One of the proven methods of the KDD process to efficiently communicate, comprehend and interact with this complex and large amounts of information is *Data Visualization*. However, the increasing dimensionality and the growing volumes of the data pose a challenge to the current data analytics systems to visualize high dimensional data and unfold the hidden information. It not only requires a great deal of manual effort but also a considerable amount of domain knowledge related to the dataset in order to visualize it in a meaningful way such that useful insights existing within the data can be identified. The key challenges faced in visualizing a dataset for analysis include the identification of dimensions to be visualized as well as the type of visualization technique to be used.

Each of the visualization techniques has its own merits. Determining the "most meaningful" visualizations for a given dataset depends on the characteristics of the data to be visualized as well as the relationship between the dimensions. Another aspect, that the type of visualization techniques rely upon, is the type of data relationship to be visualized for a set of intended tasks. Meaningful data relationships include distribution, comparison, correlation, change over time, to name a few. Applying the most efficient visualization technique based on the characteristics of the dataset as well as the nature of the intended task, helps to improve the understandability of the data. The increase in the dimensionality of data also results in various challenges for data visualization. Visualizations based on more than three dimensions of data require efficient ways to display the provided data such that the visualization is easy to comprehend as well as the data itself does not lose its meaningful aspect. This is mainly because human cognition limits the number of data dimensions that can be visually interpreted. The potential amount of overlapping data points projected onto a two-dimensional display hinders the interpretation of meaningful patterns in the data.

To address these challenges, the process of extracting data characteristics and the generation of task-oriented visualizations is automated. One important factor for the automation of data visualization is to make it accessible for all and not just for people with technical expertise (Hu et al., 2019). This au-

Chakrabarti, A., Ahmad, F. and Quix, C.
Towards a Rule-based Visualization Recommendation System.
DOI: 10.5220/0010677100003064
In *Proceedings of the 13th International Joint Conference on Knowledge Discovery, Knowledge Engineering and Knowledge Management (IC3K 2021) - Volume 1: KDIR*, pages 57-68
ISBN: 978-989-758-533-3; ISSN: 2184-3228

tomation is done in the form of *Visualization Recommendation Systems*. As a result, there has been a significant rise in the research and development of systems in recent years (Vartak et al., 2015)(Hu et al., 2019)(Luo et al., 2018)(Krause et al., 2016). These recommendation systems process the data and provide a set of visualizations providing insights which could be helpful for the intended task of the analyst.

Most of the recommendation systems are designed for the sole purpose of selecting the most relevant features in the data and therefore, support a limited number of visualization techniques. For example, SeeDB (Vartak et al., 2015) aids the users to identify interesting visualizations using a deviation-based metric, yet it only uses bar charts to display the result and other recommendations whereas DEEPEYE (Luo et al., 2018) only uses four common visualization techniques namely bar charts, line charts, pie charts, and scatter charts. SeekAView (Krause et al., 2016), also has a fixed number of visualization techniques to select the useful types of trends from, including frequency plots, scatter plots and parallel coordinates. The drawback of using a low number of visualization techniques is that it is not possible to cover every type of task using the same type of visualization technique.

The existing visualization recommendation systems that are proposed by the current literature face several shortcomings. These shortcomings range from the number of dimensions the recommendation system can handle to the number of visualization techniques adopted. Moreover, most of these systems require inputs from expert users having domain specific knowledge. Visualization recommendation systems that use a recommendation engine based on supervised machine learning or neural networks (Hu et al., 2019)(Luo et al., 2018) also suffer from the problem of overfitting. Training the model for the recommendation engine relies mainly on the training data and a lack of available training data results in an ineffective model that may be biased towards the data similar to the training data.

To address these challenges, in this paper we propose a novel rule-based visualization recommendation system. We show how impartial and effective visualizations are recommended by using a knowledge-based rule engine which is designed and developed as a part of our work. The recommendation system uses key factors such as data characteristics, intended task and user feedback along with the knowledge-base to decide the best suitable type of visualization technique to be used. Furthermore, the recommendations generated are ranked qualitatively based on several statistical properties of the data.

Our contributions in this paper are summarized below:

1. **Classification of Data into Characteristics and Proposing a Formal Visual Taxonomy:**
   The data is categorized based on several factors such as the type of data (discrete, continuous) or its format (e.g. Numerical, Categorical). These factors are required as they influence the type of visualization technique to be used and are therefore essential for the development of the rule engine. A formal visual taxonomy is proposed as well, which provides the theoretical foundation for the construction of the knowledge base.
2. **Mapping User Tasks to Visual Structures and Creation of the Task based Visual Taxonomy:**
   The intended user task, required to generate visualization recommendations, is abstracted in the form of a task based visual taxonomy which in turn is mapped on to the type of visualization techniques in order to generate the recommendations.
3. **Creation of Rules for Knowledge-based Rule Engine:**
   After the categorization of data based on its characteristics, knowledge base rules are generated to provide recommendations. The input factors for the rules, apart from data categories, include aspects such as the intended task of the user and the number of dimensions to be visualized. Based on the input factors, the rule engine then decides the best suitable visualization techniques in the form of recommendations.
4. **Ranking of Visualization Recommendations:**
   The recommendation system generates a set of visualizations as multiple rules could be applicable. Therefore, a ranking algorithm is implemented based on task dependant statistical measures so that the visualizations are sorted in a descending order based on their scores ensuring that the most useful visualizations are displayed first to the user resulting in an efficient process.
5. **Evaluate the Designed System:**
   We test a sample scenario by using a real-world dataset in order to evaluate the usefulness of the generated and ranked recommendations and compare the results with a popular visualization tool.

2 RELATED WORK

As discussed in the previous section, there has been significant progress in the research and development of visualization recommendation systems in recent years. According to (Kaur and Owonibi, 2017), these

systems generate recommendations based on one of the four following strategies:

- **Data Characteristics Oriented:** Recommendation systems based on this strategy focus primarily on the characteristics and type of data to generate visualizations. The data attributes are used to create visual marks for the final visualization. A key feature of this approach is the formalization of visual mappings from data characteristics to visual marks. The work done in this field includes VizQL (Mackinlay et al., 2007) (used in the Show Me module of Tableau) and Vega-lite (Satyanarayan et al., 2017). Both provide formal declarative specifications to convert the data characteristics into visual mappings. These include mappings such as selecting the x and y axes dimensions, the data type, the mark type to be used and the summaries (such as the mean) to be displayed. These formal mappings are then used to create rules that can be applied to generate useful visualizations based on the dataset provided.
- **Task Oriented:** This strategy uses the concept of intended task to visualize the data. These intended tasks may include identifying data relationships such as correlation, comparison, distribution etc. and the type of visualization technique to be used by the recommendation system relies heavily on this. Most of the current studies in this area create the user task list manually (Kaur and Owonibi, 2017) and then apply the data characteristics approach based on the chosen intended task.
- **Domain Knowledge Oriented:** The research in this area focuses on improving the recommendations based on the domain knowledge. This is done by employing the task and data in the vocabulary of the problem domain in order to satisfy the user requirements in that specific domain (Kaur and Owonibi, 2017). This can be done based on knowledge-sharing or gaining domain knowledge from existing knowledge bases. RAVE (Klumpar et al., 1994) uses NASA's domain knowledge along with user-selected tasks or visualization type to generate a meaningful visualization. RAVE is capable of generating visualizations such as a 2D scatterplot, bar graph and pie chart using its knowledge base. Semantic-based recommendations in the form of ontologies are also a key research area for domain knowledge-based recommendation systems.
- **User Preferences Oriented:** This approach explicitly requires user input to decide the user preference and generate recommendations accordingly. This can, later on, be used to improve the system. The visualization recommendation systems, using this strategy, constantly analyze and record the actions performed by the user to generate visualizations so that the recommendations preferred by the user can be displayed rather than the irrelevant ones (Gotz and Wen, 2009). Recent work in this area employs machine learning approaches to steadily improve the model and prune out the irrelevant recommendations (Key et al., 2012).

The following sections in this chapter include details regarding the recent studies and work done in the field of visualization recommendation systems based on one or more of the recommendation strategies discussed above.

SEEDB. Based on these recommendation strategies, popular systems have been developed in recent times. For example, SEEDB (Vartak et al., 2017)(Vartak et al., 2015) is a recently developed visualization recommendation system. Using a subset of the data extracted from a query, SEEDB is able to generate visualization recommendations that it regarded as useful based on multiple metrics. However, SEEDB does not support multiple types of visualization techniques based on the data characteristics or intended tasks and instead generates recommendations only in the form of bar charts. The problem with this type of approach is that bar charts are only able to visualize a certain aspect of data characteristics and tasks such as comparison in magnitude. Another drawback is the dependency on the user for query generation or selection of dimensions. For this aspect, the user must have some domain knowledge regarding the dataset or expertise over the visualizations.

SeekAView. SeekAView (Krause et al., 2016) is a visual analytics system that allows users to create subspaces from a high-dimensional dataset. It also provides suggestions for the users to reconfigure their views and identify interesting insights from the generated suggestions. Manual effort is required from the user to detect dimensions that show an interesting pattern. Despite being a very useful tool for high-dimensional data visualization and identifying important data trends, SeekAView is subject to deficiencies as it recommends only a specific set of visualizations like density plots for the dimensions while PCA scatterplot, parallel coordinates and a scatterplot matrix is only used for the resulting subset. The fact that SeekAView displays density plots for every dimension, makes it complex to use for data analysis for high-dimensional datasets. Therefore, domain knowledge and a high amount of manual effort is required

while using it to analyze the density plots for important dimensions whereas some key relationships between dimensions may never be identified due to the use of a low number of visualization techniques.

DeepEye. DeepEye (Luo et al., 2018) is a visualization recommendation system developed recently and employs machine learning to generate recommendations. The motivation for using this approach is to solve three individual problems. First, to decide whether an individual visualization is useful or not, second, to compare two visualizations and select the better one and last, in the case of multiple visualizations, to find the top-k ones in a dataset. Though DeepEye proposes an effective approach towards visualization recommendation system as it uses a hybrid system based on machine learning and expert rules, it is only able to suggest recommendations in the form of pie charts, bar charts, line graphs and scatterplots which are not sufficient to cover all types of useful visualization recommendations. Using machine learning models to decide the usefulness of a visualization depends considerably on the training data that is used, therefore, datasets from diverse fields are required for the training of an unbiased model whereas expert rules for only four visualizations techniques are not enough to identify all possible types of intended tasks

VizML. VizML (Hu et al., 2019) is another Machine Learning approach for visualization recommendation systems. VizML generates visualization recommendations using Neural Network and Baseline models trained and tested on one million visualizations taken from the Plotly Community Feed (Plotly,). Data collected from Plotly is cleaned by removing the duplicate visualizations, the corpus is then used for feature and design choice extractions which are subsequently used to train the models in order to predict recommendations. Although VizML is an efficient machine learning approach to visualization recommendation, it consists of a few limitations. VizML utilizes training dataset obtained from Plotly (Plotly,) only and while the dataset is large enough, it would still be biased towards Plotly. Therefore, datasets from diverse data sources and fields should be obtained and used for training the models. Another disadvantage of using Plotly datasets is the fact that it is used by both expert and non-expert users to create visualizations. Visualizations created by non-expert users are prone to error and may not be that useful.

3 OVERVIEW OF OUR APPROACH

In this section, we present our proposed model of the recommendation system. Our system is built on top of a rule engine constructed by modeling data characteristics and intended tasks into generic rules. The data characteristics axis is used to identify key characteristics of the data such as dimensionality, data type and data format while the intended task axis takes into account the goal of the user based on the type of data relationship. As a first step, data characteristics are extracted automatically from the input dataset, using the data abstraction module while the intended tasks are mapped using the task abstraction module. These two different sets of inputs are used for the construction of rules which are modeled as generic rules and are stored in the Knowledge Base. During the execution, dynamic data and task encodings are generated which are looked up in the Knowledge Base in order to retrieve the corresponding rule sets. These rules are then ingested by the Rule Engine to trigger a specific event which consists of a set of visualizations. These visualizations are then ranked and displayed to the end-user as recommendations. The complete workflow of the developed system is given in Fig 1.

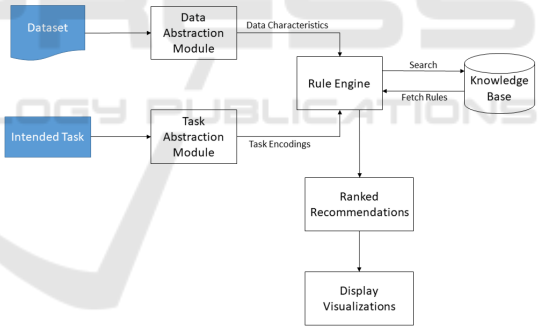


Figure 1: System architecture depicting the proposed workflow.

3.1 Workflow

As a first step for building our model, we construct a forward-chained *Knowledge Base*, which is used for storing recommendation rules. We justify the use of a knowledge base for handling the cold start problem as well as to ensure that the system is unbiased and generic, regardless of the domain it is used in. The Knowledge Base stores rules which are based on the data characteristics (extracted in the Data Abstraction module) and the intended tasks of the user which are mapped from the user input in the task abstraction module (see Figure 1).

Data characteristics include aspects such as the number of dimensions required for visualization, the type of data in the dimension to be visualized (discrete/continuous/temporal) as well the data format (numerical/categorical). Whereas, the intended task includes the type of relationship the user wants to explore within the dataset through the generated visualizations.

Data Abstraction Module:

To generate visualization rules, based on data characteristics, we first create a *chart vocabulary* consisting of 16 visualization types and 6 matrix visualizations. The purpose of this Chart Vocabulary is to map multiple types of visualization techniques to the characteristics of data with varied dimensions. In Table 1 we present the complete chart vocabulary. Next, we will explain how we use our proposed chart vocabulary to construct the rules for the Knowledge Base.

Table 1: Visualization Chart Vocabulary.

Chart	Dimensions	Characteristics
Histogram	1D	Numerical (Continuous) on x-axis Numerical/Function on y-axis
Pie chart	1D	Part-whole (Categorical)
Area graph	1D - 2D	Numerical (Continuous)/Temporal on x-axis Numerical/Function on y-axis
Line graph	1D - 2D	Numerical (Continuous)/Temporal on x-axis Numerical/Function on y-axis
Boxplot	2D	Numerical, Categorical & Temporal
Bar graph	2D	Categorical/Temporal on x-axis Numerical/Function on y-axis
Heatmap	2D - 3D	Relational data (Numerical, Categorical & Temporal)
Scatterplot	2D - 4D	Numerical (Discrete) on axis Categorical as color/glyphs
Stacked bar graph	2D or more	Grouped categories (one bar) on x-axis Numerical/Function on y-axis
Multiset bar graph	2D or more	Grouped categories (multiple bars) on x-axis Numerical/Function on y-axis
Stacked area graph	2D or more	Numerical (Continuous)/Temporal on x-axis Numerical/Function on y-axis Category represented by line
Stream graph	2D or more	Temporal Category represented by color
Radar chart	2D or more	Relational data (Both Numerical & Categorical)
Multi-line graph	2D or more	Numerical (Continuous)/Temporal on x-axis Numerical/Function on y-axis Category represented by line
Bubble plot	4D	Numerical (Discrete) on axis and as size Categorical as color
Parallel Coordinates	HD	Relational data (Usually Numerical-Categorical can be used)
ScatterPlot Matrix	HD	Numerical (Discrete) on axis Categorical as color/glyphs
Matrix of Histograms	HD	Numerical (Discrete)
Matrix of Line Graphs	HD	Numerical (Continuous)/ Temporal on x-axis Numerical/Function on y-axis
Matrix of Area Graphs	HD	Numerical (Continuous)/ Temporal on x-axis Numerical/Function on y-axis
Matrix of Multi-line Graphs	HD	Numerical (Continuous)/ Temporal on x-axis Numerical/Function on y-axis Category represented by line
Matrix of Heatmaps	HD	Relational data (Numerical, Categorical & Temporal)

Task Abstraction Module:

In Figure 1, we see that the intended tasks are encoded into rules which would later be used to establish a mapping between user tasks and recommended visualizations. The intended task is a significant decision-maker in the selection of the type of visualization techniques. In order to ensure that no expertise domain knowledge is required for the recommendation system, abstraction of the intended task from the user is performed. The process of task abstraction is modeled in a tree structure as shown in Figure 2. This process consists of multiple levels of abstraction beginning with a set of easy to understand questions for the user, which is the only input required from the user. The remaining levels are constructed automatically within the recommendation system. We map the user inputs to the abstracted tasks for the following questions:

- Do you want to find the distributed range, average or extrema of data dimensions?
- Do you want to split data into categories for filtering and analysis?
- Do you want to sort or identify the trend of temporal and continuous dimensions?
- Do you want to analyze the effect of one dimension on another?
- Do you want to retrieve specific values or identify correlated dimensions, outliers or clusters?

The response is then mapped to high-level tasks comprising of Distribution, Part-to-Whole, Change over Time, Comparison and Relationship. These high-level tasks were further mapped to several atomic tasks. The atomic tasks provide finer granularity over the generic user-intended tasks and are used to establish relationships between the user tasks and supported visualization. Next, we would describe how these relationships are captured into generic rules that are stored in the Knowledge Base of the recommendation system.

Rule Construction:

In this paper, we present a formal theory of generating recommendation rules which are stored in the knowledge base. The purpose for the formalization of rules is to help design a less error-prone and more efficient knowledge base in a dynamic manner which ensures further rules can easily be added to the rule engine. This approach is a more flexible one as compared to a hard-coded static knowledge-based rule set. For defining the building block of our rule set, we introduce the concepts of *Attributes*.

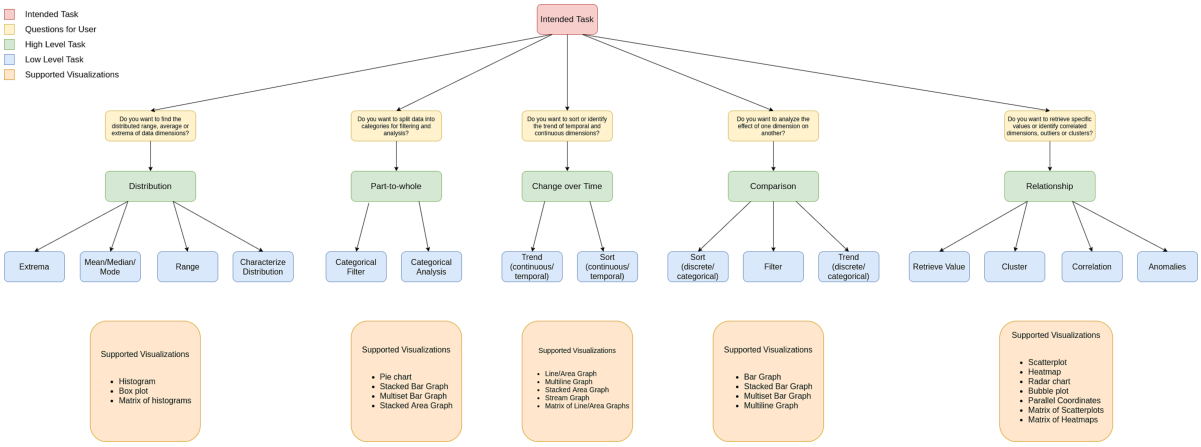


Figure 2: Task Abstraction Encodings.

We define *Attributes (A)* as the state of the system based on which a rule is fired. Every attribute is assigned a value from a specified domain set D, where $D = D_1 \cup D_2 \cup \ldots D_n$ and D_i is the finite domain for attribute A_i ($A_i \in A, i = 1 \ldots n$). Attributes are further divided into two types, atomic and composite. Formally, the distinction between atomic and composite attributes is given as $A = A_a \cup A_c, A_a \cap A_c = \emptyset$. An atomic attribute can only be assigned one domain value at a time and represented by the function: $A_i : V \rightarrow D_i$, while a composite attribute can be assigned multiple domain values and represented by the function: $A_i : V \rightarrow 2^{D_i}$, where V is a value or set of values, D_i is the domain for attribute A_i and 2^{D_i} is the subset of values taken from D_i.

The current state of a rule engine is defined as the conjuction of the current values of all the attributes in the system as is represented as:

$$s : (A_1 = V_1) \wedge (A_2 = V_2) \wedge \cdots \wedge (A_n = V_n)$$

where V_i is the current value for attribute A_i and $V_i \in D_i$ (atomic attributes) or $V_i \subseteq D_i$ (composite attributes). Unknown or unspecified state value for an attribute A_i is denoted as $A_i = null$.

Following is an example to further explain attributes within our proposed visualization recommendation system. We define the set of attributes in the visualization recommendation system as:

$$A = \{dimensionality, data\_type, data\_format, intended\_task, rec\_visualizations\}$$

The domains are formulated as:

$$D = \{D_{dimensonality} \cup D_{data\_type} \cup D_{data\_format} \cup D_{intended\_task} \cup D_{rec\_visualizations}\}$$

The attribute *dimensionality* is an example of a composite attribute and can be further divided into low dimensionality (L_D) and high dimensionality (H_D), where $L_D \subset D_{dimensionality}$, $L_D \leq 5D$, $H_D \subset D_{dimensionality}$, $H_D > 5D$ and $L_D \cap H_D = \emptyset$. The attributes *data_type* and *data_format* are atomic attributes and can have a single value at a time such as data_type = discrete and data_format = numerical. The *intended_task* attribute is also a composite attribute, e.g. for the intended task of comparison of part data to total, the intended_task attribute will be {Part-to-Whole, Hierarchical}. The last attribute consists of a set of recommended visualizations for the user and is a composite attribute as well, having one or multiple recommended visualizations as its value. Using the above formalization of the rule engine we define the generic rules for the recommendation systems that are constructed and stored in the Knowledge Base.

Below we show a sample set of our constructed rules:

- $r_1 : [(dimensionality = 1) \wedge (data\_type = \{continuous\}) \wedge (data\_format = \{numerical\}) \wedge (intended\_task = changeOverTime)]$
  $\rightarrow [(rec\_vis := line\_graph)]$
- $r_2 : [(dimensionality = 2) \wedge (data\_type = discrete) \wedge (data\_format = numerical) \wedge (intended\_task = Comparison)]$
  $\rightarrow [(rec\_visualizations := \{BarGraph\})]$
- $r_3 : [(dimensionality = 4) \wedge (data\_type = \{discrete\}) \wedge (data\_format = \{numerical, categorical\}) \wedge (intended\_task = correlation)]$
  $\rightarrow [(rec\_vis := scatterplot)]$

Based on the above rules, if data abstraction attributes *(dimensionality > 3)* $\wedge$ *(data_type = {discrete})* $\wedge$ *(data_format = {numerical})* are provided as the cur-

rent state of the system, rule r_3 is fired and the generated recommendations set is $\{scatterplot\}$.

Knowledge Base Construction:

The above formalizations for the rule generations are extended for the construction of the Knowledge Base. We propose a formal design principle, using which the Knowledge Base could be easily extended to incorporate rules on the fly.

Firstly, we define a *rule* $r = (COND, DEC, ACT)$, where $COND$ is the conjunction of a set of conditions that are to be fulfilled for the rule to be fired and can be represented as $[\mu_1 \wedge \mu_2 \wedge \cdots \wedge \mu_n]$, where $\mu_i \in COND, i = 1 \ldots n$ and $COND$ is the set of all conditions. DEC is the decision part of the rule, which assigns values to attributes when the rule is fired and can be represented as $[\lambda_1 \wedge \lambda_2 \wedge \cdots \wedge \lambda_m]$, where $\lambda_i \in DEC, i = 1 \ldots m$ and DEC is the set of all decisions. ACT is the transition in the system by performing certain actions based on the rule. Hence, we define an atomic rule r as $LHS(r) \rightarrow RHS(r), DO(ACT)$, where $LHS(r)$ is the conditional part of the rule, $RHS(r)$ is the decision part and $DO(ACT)$ is the independent set of actions performed by the rule. An ordered set of knowledge-based rules having the same ruleset schema can be grouped in the form of a table. A table is formally defined as: $t = (r_1, r_2, r_3 \ldots, r_n)$. In the case of multiple tables, such as the decision tables for task abstraction to extract the intended task for the recommendation system, the concept of intertable connection links is applied. A connection link is an ordered pair: $c = (r,t), c \in R \times T$, where R is the set of rules in the knowledge base and T is the set of tables. Each individual rule has its own connection link and based on the rule fired, the respective connection link transfers control from the table containing the rule to the next table connected by it. All the tables and connection links grouped together form a knowledge base for the recommendation system. The complete knowledge base can formally be represented as $K = (T, C)$ where T is the set of all tables in the knowledge base and C is the set of all connection links that connect rules within T. In context with the visualization recommendation system, the set F represents the set of features of the dataset imported by the user. The set D_T represents all possible supported data types for a feature f_i where $D_T = \{Discrete, Continuous, Temporal\}$ and $f_i \in F, i = 1 \ldots n$. The set D_F represents all possible data formats for a feature f_i and is defined as D_F = $\{Numerical, Categorical, Relational, Part\text{-}whole\}$ whereas the set I_T contains the list of intended tasks supported by the recommendation system and defined as $I_T = \{Distribution, Part\ to\ Whole, Change\ over\ time, Comparison, Relationship\}$. The intended task is extracted in the form of a decision tree (represented by Figure 2) with the help of user preferences to ensure meaningful and relevant visualization are recommended. The type of visualization techniques supported by the recommendation system are represented by the set $V_T = \{V_1, V_2 \ldots, V_v\}$ where v is the total number of visualization techniques the recommendation system can generate. Details regarding these visualization techniques and their data characteristics are provided in the Chart Vocabulary.

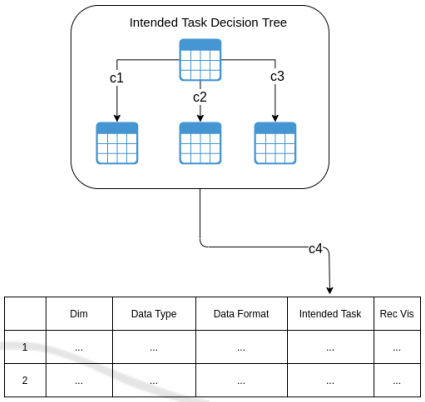


Figure 3: Conceptual Model of the Knowledge Base.

The structure of the knowledge base, based on the constructed rules, is shown in Figure 3. The knowledge base consists of a decision tree module (Figure 2) for the intended task as well as a main table consisting of knowledge-based rules. The connection links, c1, c2, c3, c4 are used to transfer control between the tables. The attributes in the main table consist of *Dim, Data Type, Data Format, Intended Task*, which are *conditional attributes*, and *Rec Vis*, which is a *decision attribute* that decides the visualizations to be recommended. The *ACT* for the decision tree component would be to populate the intended task whereas the *ACT* for the main table would be to generate the recommended visualizations. Based on the domain attribute set $A_{dim}, D_T, D_F, I_T, V_T$, the schema for the Knowledge Base can be formally defines as $S_1 = (\{A_{dim}, D_T, D_F, I_T\}\{V_T\})$.

3.2 Generate Recommendations

Algorithm 1 presents our approach for generating the visualization recommendations based on the Knowledge-Based rules. The Knowledge Base constructed based on the model, described above, is used by the rule engine to generate a list of recommended visualizations. The input parameters for the algorithm are the list of generated rules, data characteristics ex-

tracted automatically from the input dataset and the intended task mapped using task abstraction module. The algorithm then matches the provided parameters with the stored rules in a dynamic manner and adds a visualization to the recommendation list only if all parameters match the entire relevant visualization rule fields. For example, Histogram is added to the list if the dataset contains at least 1 Dimension, Data Type: Continuous, Data Format: Numerical and High-Level Task: Distribution. The list of recommendations is then sent to the Ranking module to sort them based on the ranking scores assigned to each of the visualizations and then displayed accordingly to the end-user.

Algorithm 1: Generate Recommendations.

Input: List of knowledge-based rules, data_dict = {task $\in D_{it}$, dim $\in D_{dim}$, dt $\in D_{dt}$, df $\in D_{df}$}
Output: List of recommended visualizations rec_vis $\in D_{rec\_vis}$

```
rec_vis = []
for rule in knowledge_base_rules do
    bool add_vis = true
    for item_key, item in data_dict do
        if item ≠ rule[item_key] then
            add_vis = false
        end if
    end for
    if add_vis then
        rec_vis.append(rule[rec_vis])
    end if
end for
return rec_vis
```

3.3 Ranking

Given a dataset and the set of intended task, our *Ranking Module* generates the set of top-k visualizations. The set of recommended visualizations are ranked based on the statistical properties of the data. We present our visualization ranking strategy with Algorithm 2. The proposed algorithm makes use of a list of similarity metrics to sort n items. The source entities (items) are sorted into target entities based on the most important metric and put into multiple buckets (items with the same score are put into the same bucket). The items are counted starting from the first bucket until n items are reached. In the case of multiple metrics, the remaining similarity metrics are applied iteratively in the same manner. We define the *ranking score* as :

$$S(S,T) = \sum_{i=1\ldots m} (M_i(S,T) * \prod_{j=i+1\ldots m} b_j)$$

where S and T are the source and target entities, m is the total number of metrics, $M_i(S,T)$ is the score for comparing a target entity against a source entity based on the metric M_i and b_j is the upper bound on the metric score such that $b_j > max(M_j(S,T))$. Several statistical measures are used as metrics for ranking depending on the intended task. These statistical measures include the calculation of normal distribution (Task: Distribution), the calculation of increasing/decreasing gradient (Task: Change over Time) and the calculation of correlation (Task: Comparison/Relationship). Based on the user intended task, the relevant ranking metric (normal distribution, gradient or correlation) is selected along with its respective threshold value. A score is calculated for the dimensions within the dataset and the dimensions outside the threshold score are automatically pruned out. The remaining dimensions are then sorted based on the score to ensure the highest-scoring dimensions are displayed first to the user. The calculation of such a score makes it easier to filter out meaningless visualizations and sort the remaining visualizations based on their score ensuring the user finds the most useful recommendations first.

Algorithm 2: Ranking Algorithm.

Input: D = Dimensions, R= rankingMetric
Output: *SortedDimensions*

```
sorted_dims = []
for dims in D do
    Score = calculateScore(dims, R)
    if Score in Threshold then
        sorted_dims.append(dims)
    end if
end for
Sort sorted_dims based on Score
return SortedDimensions
```

4 RESULTS AND DISCUSSION

In order to show the usefulness of our approach, we analyze the use case with the Singapore Airbnb dataset[1]. We show the recommended visualization for some of the intended tasks. The generated ranked recommendations were then compared with the recommendations generated using the "Show Me" feature of *Tableau* [2].

Experimental Setting. We have conducted our experiments in a server running Ubuntu 14.04, with two Intel Xeon X5647@2.93GHz CPUs (8 logical cores/CPU) and 16G RAM. Because of limited space, we describe the evaluation result only from one use

[1]https://www.kaggle.com/jojoker/singapore-airbnb
[2]https://www.tableau.com/

case. However, the extensive evaluation report with two more usage scenarios can be found here (https://figshare.com/s/ef97eb5e7e26374e45a1)

4.1 Singapore Airbnb Data

The Singapore Airbnb Dataset contains multiple factors that influence the price of the room available. It consists of 15 influencing dimensions, one target dimension (price) and 7922 rows. The data abstraction layer automatically discards dimensions such as the serial number (id) column and categorizes the remaining dimensions based on their characteristics. It detects two categorical dimensions, *neighbourhood_group* and *room_type* as well as three numerical continuous dimensions: *latitude, longitude* and *reviews_per_month*. There are no temporal dimensions in the dataset and the remaining dimensions are categorized as numerical discrete. Next, we show the types of recommendations generated of various tasks.

Task: Comparison:

In Figure 4, we present the snapshot of our **Recommendation Dashboard**. The Heatmap Matrix and the Matrix of Bar Graphs were the highest ranked visualizations, recommended by our system. As discussed before, the user input for the system is twofold: (i) importing the dataset and (ii) selection of the intended task. Once completed, the "Recommendation" button triggers the back-end engine to perform data pre-processing and generate mappings according to the selected tasks. These mappings were then used by the rule engine to fetch rules from the knowledge base, generate and rank recommendations and finally display them in the visualization dashboard.

Task: Distribution:

The recommendation system generates a set of visualizations and presents them in the constructed dashboard, similar to the ones generated for the comparison task as shown in Figure 4. For the lack of space, we show only the top ranked visualizations which highlights the distributions of important variables in the dataset. A histogram matrix and a boxplot matrix as shown in Figure 5 are the two top ranked visualizations. The histogram matrix contains a set of histograms for eight useful dimensions. Analysis regarding multiple factors such as price, availability over the year, minimum nights and latitude/longitude of available places can be made using this matrix. Whereas, the boxplot matrix are generated using both categorical dimensions, the *neighbourhood* dimension and the *room_type* dimension. The neighbourhood category is split into five regions of the city while there are three types of rooms (private, shared, entire apartment) within the dataset. Our system recommends the boxplots, which can easily be analyzed to show that the Central Region of the city has the highest average and maximum prices while the North-East Region is the cheapest available option for a room.

Task: Relationship (Clusters/Correlation/Outliers).

In order to identify existing relationships within the dimensions of the dataset, the recommendation system generated a Matrix of Heatmaps and two Parallel-Coordinates charts for both categories, shown in Figure 6. These visualizations can be used to detect the existence of clusters, correlation or outliers within the dataset. For example, the heatmap matrix could identify the trend in the increasing price of listings moving from the North Region (less expensive) to the West Region (more expensive) as well as the magnitude of the price difference between each room region-wise. In addition, using the generated Parallel-Coordinates charts, one can identify how multiple numerical discrete dimensions rely on each other for the provided categories.

Comparison with Tableau. As a qualitative evaluation for our proposed system, we compare the generated visualizations for the same task with the recommendations generated using the *Show Me* feature of Tableau. This resulted in a matrix of Stacked Area graphs for every region. Figure 7 shows the result with the *reviews-per-month* and *year-of-last-review* on the x-axis, average values of multiple dimensions on the y-axis and *room-type* as color. In order to generate other meaningful visualizations, manual dimensions (room type, neighbourhood group and price) were selected and the system recommended a Multibar graph for each type of room using color for the region and the average price on the y-axis (see Figure 7). Similar visualizations were recommended by our system as shown in Figure 4 along with multiple other ranked visualizations. Our proposed recommendation system performed better in terms of qualitative analysis of results compared to Tableau by not only generating the visualizations recommended by Tableau but also generating multiple other ranked visualization charts based on various statistical metrics and user intended tasks. Therefore, we conclude that the recommendation system, presented in this paper, proves its efficiency in generating automated visualizations in turn providing meaningful data insights.

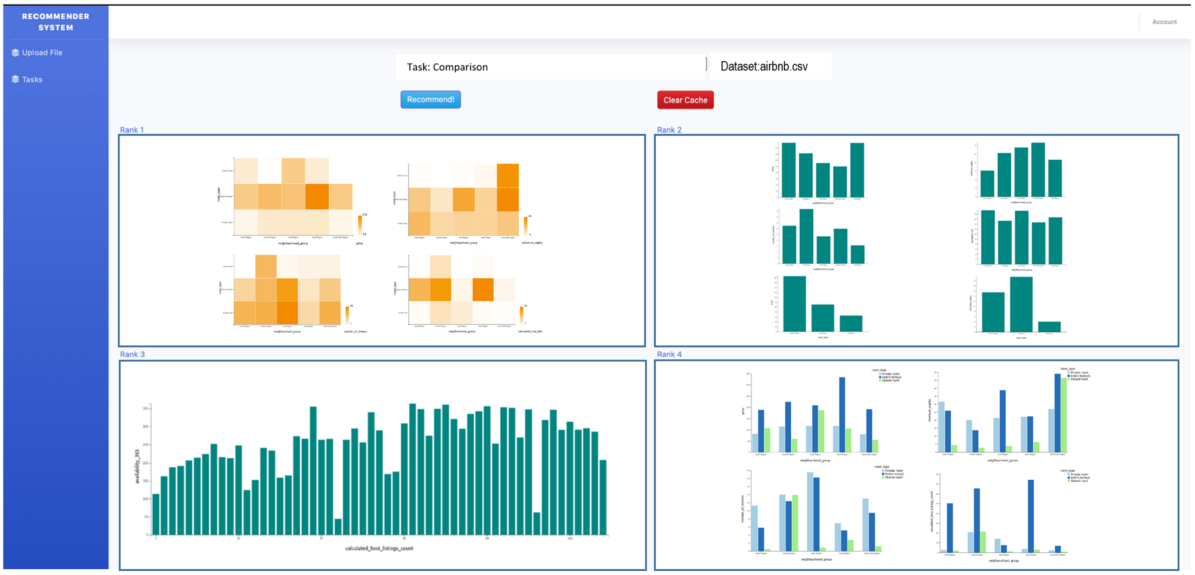


Figure 4: Visualization Recommendation Dashboard for Comparing Data Dimensions.

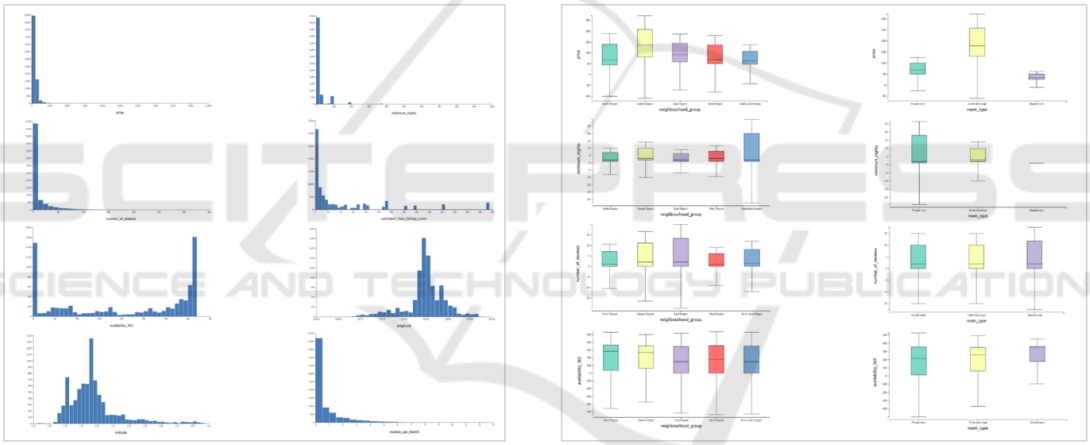


Figure 5: Recommended Visualization for the Distribution Task.

5 CONCLUSION

In this paper, we present a rule-based recommendation system that generates and ranks visualizations by extracting data characteristics and abstracting the user-intended task. An efficient Knowledge Base was presented, which mapped the data and the task abstractions as rules. Formal methods for constructing such a knowledge base, as discussed in this paper, provide a blueprint for modeling rule-based visualization recommendation systems. What makes this knowledge-base oriented rule-engine efficient is the fact that it has been implemented in a completely dynamic manner for future enhancements. As far as our knowledge, there exists no such system in the current literature which provides a recommendation model encompassing the entire visualization ecosystem. The automated workflow, starting from data ingestion and ending at the recommended visualizations being rendered in the screen for the end-user, is achieved by the capability of our system to automatically detect data dimensions and sort them into categories. To address the ”human-in-the-loop” factor for generating recommendations, we have presented a generic task abstraction model in form of a hierarchical tree structure that provides a mapping between the high and low level intended tasks and the set of visualizations supported by the relevant tasks.

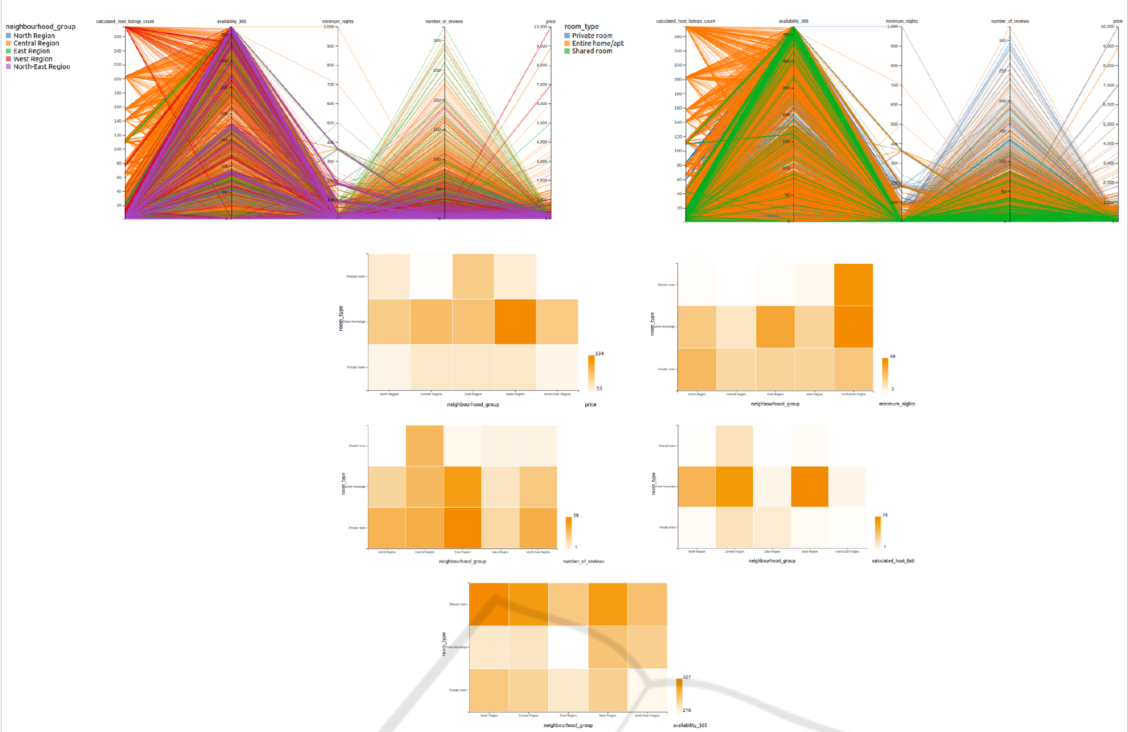
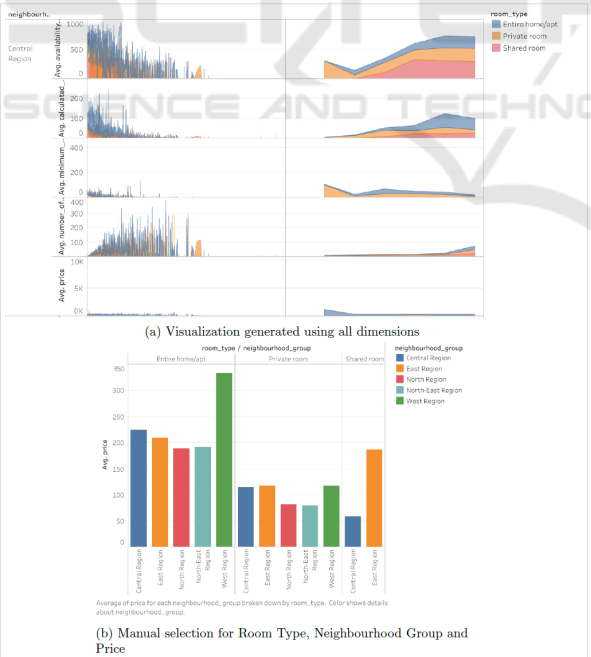


Figure 6: Recommended Visualizations Depicting Relationships.



(a) Visualization generated using all dimensions

(b) Manual selection for Room Type, Neighbourhood Group and Price

Figure 7: Visualizations Recommended by the ShowMe Function of Tableau.

Future Work. The system currently extracts key data characteristics from each dimension to divide the dimensions into respective categories. This module can further be enhanced by creating a scoring based algorithm to sort the dimensions in terms of usefulness which can then lead to filtering out irrelevant dimensions in an automated manner. Finally, due to the recommendation system being generic regardless of any specific domain field, a large set of visualization techniques and user intended tasks have been implemented in the system. However, additional enhancements can be made to the existing set of intended tasks supported by the recommendation system as well as multiple other visualization techniques not currently supported, such as a Tree Map, in order to cover a wider range of user tasks and visualization techniques.

ACKNOWLEDGEMENTS

Funded by the Deutsche Forschungsgemeinschaft (DFG, German Research Foundation) under Germany's Excellence Strategy – EXC-2023 Internet of Production – 390621612.

REFERENCES

Gotz, D. and Wen, Z. (2009). Behavior-driven visualization recommendation. In *Proceedings of the 14th International Conference on Intelligent User Interfaces*, IUI '09, pages 315–324, New York, NY, USA. ACM.

Hu, K., Bakker, M. A., Li, S., Kraska, T., and Hidalgo, C. (2019). Vizml: A machine learning approach to visualization recommendation. In *Proceedings of the 2019 CHI Conference on Human Factors in Computing Systems*, pages 1–12.

Kaur, P. and Owonibi, M. (27/02/2017 - 01/03/2017). A review on visualization recommendation strategies. In *Proceedings of the 12th International Joint Conference on Computer Vision, Imaging and Computer Graphics Theory and Applications*, pages 266–273. SCITEPRESS - Science and Technology Publications.

Key, A., Howe, B., Perry, D., and Aragon, C. (2012). Vizdeck: Self-organizing dashboards for visual analytics. In *Proceedings of the 2012 ACM SIGMOD International Conference on Management of Data*, SIGMOD '12, pages 681–684, New York, NY, USA. ACM.

Klumpar, D., Anderson, K., and Simoudis, A. (1994). Rave: Rapid visualization environment.

Krause, J., Dasgupta, A., Fekete, J.-D., and Bertini, E. (23/10/2016 - 28/10/2016). Seekaview: An intelligent dimensionality reduction strategy for navigating high-dimensional data spaces. In *2016 IEEE 6th Symposium on Large Data Analysis and Visualization (LDAV)*, pages 11–19. IEEE.

Luo, Y., Qin, X., Tang, N., and Li, G. (16/04/2018 - 19/04/2018). Deepeye: Towards automatic data visualization. In *2018 IEEE 34th International Conference on Data Engineering (ICDE)*, pages 101–112. IEEE.

Mackinlay, J., Hanrahan, P., and Stolte, C. (2007). Show me: automatic presentation for visual analysis. *IEEE transactions on visualization and computer graphics*, 13(6):1137–1144.

Plotly. Plotly community feed. https://plot.ly/feed.

Satyanarayan, A., Moritz, D., Wongsuphasawat, K., and Heer, J. (2017). Vega-lite: A grammar of interactive graphics. *IEEE transactions on visualization and computer graphics*, 23(1):341–350.

Vartak, M., Huang, S., Siddiqui, T., Madden, S., and Parameswaran, A. (2017). Towards visualization recommendation systems. *ACM SIGMOD Record*, 45(4):34–39.

Vartak, M., Rahman, S., Madden, S., Parameswaran, A., and Polyzotis, N. (2015). Seedb. *Proceedings of the VLDB Endowment*, 8(13):2182–2193.