Beyond Heuristics: Learning Visualization Design

Bahador Saket*, Dominik Moritz*, Halden Lin, Victor Dibia, Çağatay Demiralp, Jeffrey Heer

Abstract— In this paper, we describe a research agenda for deriving design principles directly from data. We argue that it is time to go beyond manually curated and applied visualization design guidelines. We propose learning models of visualization design from data collected using graphical perception studies and build tools powered by the learned models. To achieve this vision, we need to 1) develop scalable methods for collecting training data, 2) collect different forms of training data, 3) advance interpretability of machine learning models, and 4) develop adaptive models that evolve as more data becomes available.

Index Terms—Automated visualization design, machine learning, design guidelines, visualization recommendation, feature engineering, visualization recommendation systems

1 Introduction

The demand for data visualization has significantly grown in recent years with the increasing availability and complexity of digitized data across everyday domains. By visually encoding data properties, visualizations aim to enhance data understanding by leveraging human visual perception, which has evolved for fast pattern detection and recognition. Understanding the effectiveness of a given visualization in achieving this goal is a fundamental pursuit in data visualization research and has important implications in practice. Visualization researchers regularly conduct empirical studies to investigate how people decode and interpret visualizations (e.g., [6, 12, 23, 26, 27]). Such empirical studies are important for understanding and improving the effectiveness of visualizations. Indeed, design guidelines and heuristics that we use today in data visualization are an accumulation of what we have learned from such empirical studies over decades.

It is not, however, always possible to distill guidelines through analysis of data collected by empirical studies due to various confounding factors. Even when this is possible, derived guidelines might be inadequate for conveying the subtleties present in user data or might be marred by those providing the guidelines. Moreover, design guidelines provided by empirical studies often make their way to visualization tools slowly for two main reasons. First, our design knowledge is continually evolving as visualization researchers regularly publish results from empirical studies and provide new sets of guidelines for designing effective visualizations. Second, designers of visualization tools need to spend a significant amount of time and effort to manually incorporate these guidelines.

In recent years, there has been an increasing trend to publish data collected from empirical studies with increased awareness of the importance of replicability and reproducibility. Researchers have made large datasets of experimental data of visualizations' effectiveness publicly available (e.g., [10, 12, 23, 26]). We advocate learning models from this data and building tools that automatically apply the learned model. We believe machine learning models provide a practical opportunity to implicitly engineer the insights provided by empirical user performance data into visualization systems. The main advantage is that new guidelines can be applied in practice faster in an unbiased and reproducible manner.

*Bahador Saket and Dominik Moritz contributed equally to this work.

- *Bahador Saket is with Georgia Tech. E-mail: saket@gatech.edu.*
- *Dominik Moritz, Halden Lin, and Jeffrey Heer are with University of Washington. E-mail: domoritz, haldenl,jheer@uw.edu.*
- *Çağatay Demiralp is with Columbia University. E-mail: cd3057@columbia.edu.*
- *Victor Dibia is with IBM Research. E-mail: dibiavc@us.ibm.com.*

Manuscript received xx xxx. 201x; accepted xx xxx. 201x. Date of Publication xx xxx. 201x; date of current version xx xxx. 201x. For information on obtaining reprints of this article, please send e-mail to: reprints@ieee.org. Digital Object Identifier: xx.xxxx/TVCG.201x.xxxxxxx

In this paper, we discuss how we might automatically derive visualization design principles from experimental data in a scalable way by using machine learning. We further categorize and discuss the existing approaches for learning visualization designs to illustrate feasibility and how future systems fit into these categories. We argue that visualization design principles used today are also derived from data. However, these principles had to be abstracted by a visualization researcher, taught by a teacher, and applied manually by the designer. We propose that the next step for the research community is to curate a knowledge base of design principles that can be applied automatically. Next, we should aim to learn from data both design principles and how to trade off among them. As more data becomes available, we may one day be able to learn visualization design end-to-end. In the following, we describe each of these steps, and future research directions to achieve this vision.

2 Visualization Recommendation Systems

In this section, we discuss existing and previous work on automated visualization design and recommendation engines that incorporate guidelines derived from graphical perception studies. We categorize these systems into **knowledge-based**, **data-driven**, and **hybrid** visualization design tools. These categories are used in recommender systems [2] and represent where the knowledge that the system uses to recommend visualizations comes from. Table 1 gives an overview of existing machine learning and knowledge-based systems.

2.1 Knowledge-Based Systems

A large body of existing automated visualization design tools focuses on suggesting visualizations based on user-defined constraints (such as which data attributes to visualize) and design constraints. Mackinlay's APT [16] leverages a compositional algebra to enumerate the space of visualizations. It then uses *expressiveness* and *effectiveness* criteria based on the work by Bertin [5] and Cleveland & McGill [6] to prune and rank visualizations. *Expressiveness* refers to the ability of a visualization to convey all and only the facts in the data. *Effectiveness* refers to the ability of a visualization when the information it conveys is more readily perceived than with other visualizations. The SAGE project [18] extends APT by taking into account additional data properties such as cardinality, uniqueness, and functional dependencies. Tableau's Show Me [17] introduces a set of heuristic rules to recommend visualizations. Following this line of work, CompassQL [28] also uses similar heuristic rules to develop expressiveness and effectiveness criteria to evaluate visualization options. However, CompassQL extends the earlier work by using a set of hand-tuned scores to specify criteria such as space efficiency and legibility based on visualization and data properties.

Many of the automated design tools discussed above prune and rank candidate visualizations based on a set of manually curated design guidelines derived from previous empirical studies. Designers of these tools often spent a considerable amount of time and effort to incorporate the findings of the previous and current empirical studies.

System	Recommender Type	Modeling Approach	Learning	Model	Input Features	Design Space
`VizDeck`	Data Driven	Basic Features	Yes	Linear	5 Data properties per field	9+ Types
`Kopol`	Data Driven	Basic Features	No	Tree	Data properties, Task	5 Types
`Kim et al.`	Data Driven	Basic Features	No	Linear	Data properties, Task	12 Scatterplots
`Data2Vis`	Data Driven	Raw Features	Yes	Deep	Raw Dataset	Vega-Lite
`APT`	Knowledge Based	Hand Tuned	No	Rules	Data Types, Importance ordering	APT
`CompassQL`	Knowledge Based	Hand Tuned	No	Linear	Partial Specification	Vega-Lite
`Draco`	Hybrid	Learning Trade-Offs	Yes	Linear	Partial Specification, Task	Vega-Lite

Table 1. Comparison of different approaches to model automated visualization design by the type of recommender as (Sect. 2), the modeling approach based on the kind of features that are used (Sect. 3), whether the approach uses machine learning with genralization, the type of model, the input to the model, and the design space.

2.2 Data-Driven Systems

The visualization literature is no stranger to data-driven models elicited through human-subject experiments. For example, low-level perceptual models such as Weber-Fechner Law [9], Stevens' Power Law [25], and perceptual kernels [7] are all based on fitting parametric and non-parametric models to empirical user data, informing low-level visual encoding design. While data data-driven models are prevalent, using data-driven models for automated visualization design is a nascent area and only a handful of papers exist to date.

With advances in machine learning, more researchers in the visualization community started taking initial steps towards developing machine learning models to recommend visualizations. A machine learning model tries to best predict what visualization is most appropriate based on the given inputs (e.g., tasks, data attributes, etc.). Developers of visualization tools need to hand-craft informative, discriminating, and independent features for learning such models. To the best of our knowledge, VizDeck [11] was the first attempt at learning a recommendation model for high-level visual design. VizDeck is a web-based visualization recommender tool designed for exploratory data analysis of unorganized data. Using users' up and down votes on a gallery of visualization, VizDeck learns to recommend charts that the user is most likely going to vote up. It does so by learning a linear scoring function over statistical properties of the dataset. Visualizations are picked from a corpus of possible candidate visualizations.

Saket et al. [23] recently conducted a crowdsourced study to evaluate the effectiveness of five visualization types (Table, Line Chart, Bar Chart, Scatterplot, and Pie Chart) across 10 different visual analysis tasks and from two different datasets. Based on their findings, they developed Kopol[1], a mini JavaScript prototype visualization recommender that uses a decision tree to suggest visualizations based on the given tasks and data attribute types. Kim et al. [12] also recently developed a model for 12 scatterplot encodings, the task type, and the cardinality and entropy of some data fields. During their crowd-sourced experiment, Kim et al. had users perform tasks for a combination of features and used results to create a ranking.

Luo et al. [15] conducted a study in which 100 participants annotated 33,412 visualizations as good/bad, and provided 285,236 pairwise comparisons between visualizations. They used 42 public datasets to create the visualizations for their experiment. Luo et al. then developed DeepEye [15], a visualization recommender system that uses a binary classifier to decide if a visualization is good or bad, and a supervised learning-to-rank model to rank the visualizations based on users' preferences data collected in their experiment.

A more recent trend in machine learning is to learn models on all available features instead of hand-crafting good features—a labor intensive and often biased process. For example, Data2Vis [8] is a neural translation model that generates visualization specifications in the Vega-Lite grammar from descriptions of a dataset. The model was trained on thousands of example pairs of data and visualizations recommended by CompassQL.

2.3 Hybrid Systems

In knowledge-based systems, the system designer provides the knowledge about visualization design. In data-driven systems the system learns a recommendation model from data. Hybrid recommender systems are both knowledge-based and data-driven. The system designer has full control over the recommendation process but can augment the knowledge base with machine learning. Recently, machine learning experts argued that learning with structure must be a top priority for AI to achieve human-like abilities [3].

Draco [19] is a formal model that represents (1) visualizations as sets of logical facts and (2) design principles as a collection of hard and soft constraints over these facts. Using constraints, we can take theoretical design knowledge and express it in a concrete, extensible, and testable form. Compared to VizDeck and Kopol, Draco's visualization model covers a larger set of visualizations supported by Vega-Lite [24]. Similar to CompassQL [28], Draco's default model uses rules that are informed by empirical studies but formalized by experts. To avoid ad-hoc choices and support an evolving knowledge base, Draco can learn how to trade off among competing design rules using a simple linear model. Draco's learning algorithm uses learning to rank, a machine learning method that enables it to learn a preference model from ordered pairs of complete visualizations. Using this method, Draco can learn without the need to normalize scores between perceptual studies with different methods and conditions.

3 Features for Modeling Visualization Design

In this section, we discuss the different approaches to automated visualization design with respect to the features that they use. In particular, we discuss building models from basic features (visualization type, data properties, task), feature engineering, learning design rules, and learning without feature engineering. Table 1 provides an overview over the systems discussed here.

In the discussion below, we assume that a machine learning model for visualization design takes as input a set of features that can include some specification of the visualization, data, and task and outputs a corresponding score. This score can represent how preferred a visualization is (based on effectiveness, how easy it to read the visualization, etc.). The magnitude of the score may not be meaningful, but it provides a rank ordering of the feature vectors (i.e., possible designs). An automated visualization design system can use any model of this form by enumerating possible designs and recommending the one with the highest score.

3.1 Modeling Using Basic Features

The simplest and fastest way to model a score is to use the results of experimental studies of effectiveness. These models use data such as the type of visualization, data properties, and task type as features. For example, Kopol by Saket et al. [23] and Kim et al. [12] are examples of this approach. They use data such as tasks, visualization types, and data attribute types as features. VizDeck [11] is another example of a system that uses basic features. Instead of learning from effectiveness studies, VizDeck [11] uses users' up and down votes to change the scoring of each visualization alternative. In addition to the visualization type, VizDeck's model uses statistics of the data as input.

While it is often simple and fast to use the results of experimental studies to learn models (similar to Kopol [23] and Kim et al. [12]), the design space of these models is small since they only support a small set of visualization types and a basic set of features. In practice, visualization designers might want to consider a larger design space.

[1]https://kopoljs.github.io/

Going forward, the visualization community needs to develop models that cover a much broader design space. To effectively discriminate data in a larger design space, models need to use more expressive features. In the next section, we discuss a method for learning generalizable models with expressive features for visualization design.

3.2 Using Visualization Design Rules as Features

Another method for automating visualization design is to use the violations of design rules as features for learning models. A design rule is a predicate over properties of a visualization, the user's task, and the data. For example, "Do not use the same field for the x and the y channel of a scatterplot" describes the properties scatterplot and the same field on x and y should not occur together. Assuming a set of design rules, the feature vector representation of a visualization design is the the number of times a design rule is being violated by the design. For example, a design that does not violate any design rules can be represented as a feature vector with only zeros. We can use these feature vectors to learn a model. Since each feature corresponds to a design rule, the weight of each feature in the learned model is a measure of the relevance of its corresponding rule. Many of the design rules that we use today are not always prescriptive. Today's visualization designers need to decide what rules they prefer for the specific visualization they are working on. Machine learning models use statistical models to handle uncertainty and noise in training data and are thus well equipped to handle design rules that are not prescriptive. A model that uses design rules as features learns the trade-offs among competing rules from data and is more deterministic than human designers when applying them.

Recently, Moritz et al. proposed Draco [19], a method to derive feature vectors for a machine learning model from Vega-Lite specifications [24]. This allows for more sophisticated features to be composed, through feature-engineering, from the underlying grammar. The main idea in Draco's learning approach is to use the violations of *design rules as features* as outlined above.

The main advantage of this approach over pure data-driven learning is that the model can incorporate existing knowledge into the algorithm and thus learn a model that generalizes from little data. This approach is also a generalization of learning from basic features (Sect. 3.1), as every feature about the visualization type or the data can be written as a predicate over the structural representation of the visualization (e.g., in Vega-Lite) or the data schema. For example, to support tasks, we may just add rules for each task. However, one of the main challenges with this approach is the limited availability of design rules encoded in the system. Researchers should work on systematically enumerating design rules and encode them in a machine readable form so that they can be used as features in machine learning systems. In the next section, we discuss how machine learning may support them in this endeavour.

3.3 Learning Features from Data

Systems that use design rules for feature engineering and then learn a linear scoring function over a vector of violations are limited by these rules. Design rules relate properties of a visualization, and learning these relations is known in machine learning as structure learning. For example, inductive logic programming methods can infer logic programs from specific instances [21]. However, many algorithms assume no or very little noise. Design rules, however, are not prescriptive and have exceptions. For example, the design rule that quantitative scales should start at zero does not make sense when the data has no meaningful zero such as temperature data. However, that does not mean that the zero rule should not be used. Markov logic networks [22] are statistical models that support Bayesian reasoning and thus can handle this uncertainty well. Their structure can be learned [13]. It remains an open question whether these learning methods produce reasonable design rules from the experimental data that is available today. Moreover, more work is needed to investigate how to design an experiment to collect data that results in reliable rules. One approach may be to learn rules from data but have experts confirm them.

The key to learning structured rules is the availability of large high quality datasets of examples. However, as more data becomes available, we may be able to skip the feature engineering step and learn an end-to-end model, as we discuss in the next section.

3.4 Learning without Feature Engineering

A recent trend in machine learning is to learn models on all available features instead of hand-crafting good features. In particular, deep learning models shift the burden of feature engineering to the system by automatically learning abstract features and complex relationships that would otherwise be difficult to capture by humans. This also means models can be more flexible. Machine learning on the full data has led to impressive results in computer vision and machine translation among other areas. However, deep learning models in particular are extremely data hungry often requiring millions of training examples. The data also needs to cover a large fraction of the design space. For example, to learn a visualization design system that synthesizes visualizations not just from templates but a grammar, the model needs to first learn the grammar. For instance, Data2Vis [8] uses 215 000 training examples to learn a subset of the Vega-Lite grammar and recommendations from CompassQL. For this reason, this approach is not practical yet and more work is needed to collect enough high quality training data. A main concern with the quality of the training data is also whether the data is representative of the true distribution of visualizations in the wild. Machine learning models are probabilistic models that rank examples higher if they have seen many similar examples in their training corpus. Consequently, if the training data is biased, the model may only recommend a single visualization type. Understanding how the bias in training data affects neural models is an open area of research.

4 Next Steps in Learning Visualization Models

We view the existing body of work as the first step towards moving beyond heuristics and learning visualization design from data. Multiple avenues for future work lie in designing more interpretable machine learning models, developing scalable methods for collecting different forms of training data, and designing adaptive and evolving models.

4.1 Tooling to Help Designers Understand Models

Machine learning models can remove the manual effort of writing and tweaking rules by building rules on the fly. Moreover, as we are getting more data, it is easy to retrain the machine learning models rather quickly and frequently, thus improving the accuracy of the models. Despite the advantages of learning models, potential downsides of incorporating machine learning models include an extra layer of complexity and diminished transparency [1], both of which can make it difficult for both developers and end users to understand how the system works. Unlike cases where the designers apply visualization design guidelines manually, incorporating machine learning models might result in losing the ability to look at the designed guidelines and their underlying rationale.

Going forward, designers incorporating the empirical data to learn models should provide methods to better investigate the underlying rationale and convey an understanding of how the trained models will behave in the future. Such visualization systems should communicate a variety of technical information, such as the most representative features used in training the model, the level of correlation among different features, and others. This would help designers better understand the underlying logic behind rules extracted by the model. Moreover, we envision visualization systems combining machine learning models with state-of-the-art human-computer interface techniques capable of translating models into understandable and useful explanation dialogues for end users. Such systems should explain to the end users such things as: why does the learned model recommend a specific set of visualizations? What factors does the learned model use to prune and rank visualizations? How do user interactions with the system affect the recommendations?

4.2 Collecting Training Data

In order to learn the design of effective visualizations from user data, access to high quality data with sufficient variety and context is critical.

Perceptual studies measuring effectiveness as defined in a controlled setting can provide training data to learn design guidelines. However, current data from graphical perception studies are stored in different destinations all over the web (e.g., different repositories, personal webpages, etc.). These inconsistencies make it harder for researchers and designers to track and access available datasets. Better organizing data collected by graphical perception studies can make deriving models easier. Going forward, one solution might be to create a single data repository where the community can share data collected from these empirical studies, thus improving the accessibility of the collected data.

Graphical perception experiments often have specific research questions they set out to answer. These studies are typically conducted under different conditions, varying sample sizes, varying datasets, and for a diverse set of tasks. As such, they may provide useful but inherently partial and incomplete data to be used as training data for a generalizable models. Going forward, we need large-scale, focused data collection efforts to provide the desired consistency, size, context, and variation required to train generalizable models with large learning capacities. Active learning methods can guide the data collection.

An alternative to collecting new data is to use existing corpora of visualizations on the web, such as Tableau Public [20], Beagle [4], and the Chartmaker directory[2] or figures in papers from Viziometrics [14]. The design principles learned from this data would reflect the reality of the kinds of visualizations scientists, data analysts, and journalists create. However, the data may be confounded by the tools used in a particular community, as well as network effects.

4.3 Adapting and Evolving Models

All systems have only partial knowledge of context and user intent. For example, the analyst's goals often depend on seeing a rendered chart with real data before they realize what needs to be adjusted. As a specific example, an analyst may decide to use a log scale upon seeing that the spread of data is very large. Thus, it is crucial that recommendation systems support iterative refinement of users' intent. Moreover, individual preferences may mean that the same model is not optimal for everyone. A core concern in machine learning for visualization design should be how models can be adapted to the group or individual using it. As such, we need to develop models that take into account user feedback during visual data exploration. Ideally, the accuracy of such models should increase as users interact with the system.

Even a model that adapts to users will never provide perfect recommendation, as the models are limited. Current models are restricted by the number of features they use and the data they were trained on. Models need to evolve and expand as more data becomes available and researchers find new design rules. However, this requires designers to spend a tremendous amount of time and effort to combine existing and new data since the data are collected in different formats. One possible next step towards creating adapting and evolving models is to find a common language/format and destination to collect the results of the empirical studies. Ideally, we can develop systems that incorporate the incoming data and update the model automatically.

5 Conclusion

In the past, visualization recommendation systems have incorporated visualization design guidelines through a manual process of curation and application. In this paper, we argue it is time to move beyond this laborious process that is hard to scale. We imagine a future in which these systems are machine learning models learned from experimental data. To achieve this future, steps must be taken in engineering robust and adaptable models, developing tools for interpretability of the created models, and consolidating data. Once achieved, however, new guidelines and data collected from graphical perception studies can be applied in practice faster and in an unbiased and reproducible manner.

Acknowledgements

We thank Kevin Hu and Michiel Bakker for their feedback.

[2]http://chartmaker.visualisingdata.com/

References

[1] A. Abdul, J. Vermeulen, D. Wang, B. Y. Lim, and M. Kankanhalli. Trends and trajectories for explainable, accountable and intelligible systems: An hci research agenda. CHI '18, 2018. doi: 10.1145/3173574.3174156

[2] C. C. Aggarwal et al. *Recommender systems*. Springer, 2016.

[3] P. W. Battaglia, J. B. Hamrick, V. Bapst, A. Sanchez-Gonzalez, V. Zambaldi, M. Malinowski, A. Tacchetti, D. Raposo, A. Santoro, R. Faulkner, C. Gulcehre, F. Song, A. Ballard, J. Gilmer, G. Dahl, A. Vaswani, K. Allen, C. Nash, V. Langston, C. Dyer, N. Heess, D. Wierstra, P. Kohli, M. Botvinick, O. Vinyals, Y. Li, and R. Pascanu. Relational inductive biases, deep learning, and graph networks, 2018. arXiv:1806.01261v1.

[4] L. Battle, P. Duan, Z. Miranda, D. Mukusheva, R. Chang, and M. Stonebraker. Beagle: Automated extraction and interpretation of visualizations from the web. In *Proceedings of CHI*. ACM, 2018.

[5] J. Bertin. Semiology of graphics: diagrams, networks, maps. 1983.

[6] W. S. Cleveland and R. McGill. Graphical perception: Theory, experimentation, and application to the development of graphical methods. *Journal of the American Statistical Association*, 1984.

[7] Ç. Demiralp, M. Bernstein, and J. Heer. Perceptual kernels for visualization design. *IEEE Vis (Proc. InfoVis)*, 2014.

[8] V. Dibia and Ç. Demiralp. Data2Vis: Automatic generation of data visualizations using sequence to sequence recurrent neural networks. *CoRR*, abs/1804.03126, 2018.

[9] G. Fechner. Elements of psychophysics. 1966.

[10] J. Heer, N. Kong, and M. Agrawala. Sizing the horizon: The effects of chart size and layering on the graphical perception of time series visualizations. In *ACM Human Factors in Computing Systems (CHI)*, 2009.

[11] A. Key, B. Howe, D. Perry, and C. R. Aragon. Vizdeck: self-organizing dashboards for visual analytics. In *Proceedings of the ACM SIGMOD International Conference on Management of Data, SIGMOD 2012, Scottsdale, AZ, USA, May 20-24, 2012*, 2012. doi: 10.1145/2213836.2213931

[12] Y. Kim and J. Heer. Assessing effects of task and data distribution on the effectiveness of visual encodings. *Proc. EuroVis*, 2018.

[13] S. Kok and P. Domingos. Learning the structure of markov logic networks. In *Proceedings of the 22Nd International Conference on Machine Learning*, ICML '05. ACM, New York, NY, USA, 2005.

[14] P. Lee, J. West, and B. Howe. Viziometrix: A platform for analyzing the visual information in big scholarly data. In *BigScholar Workshop (co-located at WWW)*, 2016.

[15] Y. Luo, X. Qin, N. Tang, G. Li, and X. Wang. Deepeye: Creating good data visualizations by keyword search. In *Proceedings of the 2018 International Conference on Management of Data*, SIGMOD '18, pp. 1733–1736. ACM, New York, NY, USA, 2018.

[16] J. Mackinlay. Automating the design of graphical presentations of relational information. *ACM Transactions on Graphics*, 1986.

[17] J. D. Mackinlay, P. Hanrahan, and C. Stolte. Show Me: Automatic presentation for visual analysis. *IEEE Vis (Proc. InfoVis)*, 13, 2007.

[18] V. O. Mittal, G. Carenini, J. D. Moore, and S. Roth. Describing complex charts in natural language: A caption generation system. *Computational Linguistics*, 1998.

[19] D. Moritz, C. Wang, G. Nelson, H. Lin, A. Smith, B. Howe, and J. Heer. Formalizing visualization design knowledge as constraints: Actionable and extensible models in draco. In *IEEE Vis (Proc. InfoVis)*, 2018.

[20] K. Morton, M. Balazinska, D. Grossman, R. Kosara, and J. Mackinlay. Public data and visualizations: How are many eyes and tableau public used for collaborative analytics? *SIGMOD Rec.*, 2014.

[21] J. R. Quinlan. Learning logical definitions from relations. *Machine Learning*, 1990. doi: 10.1007/BF00117105

[22] M. Richardson and P. Domingos. Markov logic networks. *Machine Learning*, 62, 2006. doi: 10.1007/s10994-006-5833-1

[23] B. Saket, A. Endert, and Ç. Demiralp. Task-based effectiveness of basic visualizations. *IEEE Vis (Proc. InfoVis)*, 2018.

[24] A. Satyanarayan, D. Moritz, K. Wongsuphasawat, and J. Heer. Vega-Lite: A grammar of interactive graphics. *IEEE Vis (Proc. InfoVis)*, 2017.

[25] S. S. Stevens. On the psychophysical law. *Psychological review*, 1957.

[26] D. A. Szafir. Modeling color difference for visualization design. *IEEE Vis (Proc. InfoVis)*, Jan 2018. doi: 10.1109/TVCG.2017.2744359

[27] J. Talbot, J. Gerth, and P. Hanrahan. Arc length-based aspect ratio selection. *IEEE Trans. Visualization & Comp. Graphics*, 2011.

[28] K. Wongsuphasawat, D. Moritz, A. Anand, J. Mackinlay, B. Howe, and J. Heer. Voyager: Exploratory analysis via faceted browsing of visualization recommendations. *IEEE Vis (Proc. InfoVis)*, 2016.