ECS272 Final Report: Implementing Chart Constellations

Pouneh Bahrami Chitrabhanu Gupta Matt Lyons Jackson Vanover

March 17, 2020

1 Introduction

With Chart Constellations, Xu et al presented a meta-visualization system intended to foster collaboration by supporting a single analyst in the review and analysis of data stories created by a team of analysts. Rather than iterating through individual charts from each data story, the Chart Constellations system allows a user to project charts into a two-dimensional space, cluster based on their similarity, filter based on attributes, and otherwise aggregate insights based on the previous work of others.

However, our team harbors skepticism regarding the theoretical underpinning of the means by which the clustering and dimensionality reduction techniques were applied in the original work. Consequently, our implementation of this visualization system attempts to provide a similar user interface while restructuring the underlying data-processing techniques.

2 Our Approach

Collecting The Dataset For the original implementation of the Chart Constellations visualization system, the authors used a dataset they constructed from 11 authors' analyses on Kaggle, of the Global Terrorism Dataset. For these 11 authors, they manually tagged all of their charts (47 in total) with keywords, dimensions used, and its Vega-Lite code. Then, they could convert these values into a feature space, in order to cluster the graphs.

The starting point of our implementation was the task of sourcing the data. We scanned several kernels on Kaggle for the Student Alcohol Consumption dataset and collected a set of 54 different charts across 7 different authors. Our objective was to compile a dataset that would possess substantial diversity in the type of visualizations, while simultaneously having subsets of datapoints with certain similarities (often subtle) in terms of the topic of the visualization, the type of the chart, the motivation behind it or even basic appearance. This was necessary in order to validate the generation of satisfactory clusters of similar visualizations. Once we were satisfied that we had a suitable dataset, in addition to recording the visualization itself as a PIL object, the author name and the title of the visualization, we tagged each image with descriptive keywords and noted against each visualization the features of the data that were used in it. The keyword and feature data were the basis for developing the keywords embedding and the represented features embedding. These two embeddings, along with the local binary encoding embedding, were normalized and concatenated into a single dynamic vector that is consumed by our clustering and PCA modules. This formed the structure of the underlying dataframe in our system. The high

volume of keyword tags that we generated made the dataset extremely information-rich, which was beneficial for generating well defined keyword embeddings.

Defining The Feature Space Once we had our keywords, dimensions used, and the image of the graph, we needed to convert these into usable feature vectors, so that we could do clustering and dimensionality reduction in a way. To encode the keywords, we used word2vec, and to encode the dimensions used, we used a 1-hot encoding. These 2 techniques are identical to what was employed in the paper.

To convert the image of the graph into a meaningful feature vector, we used a technique called "local binary patterns". This method looks at each pixel in the graph, and assigns it a code based on its neighbors. For each neighbor, if it is darker than the current pixel, it is a 0, and if it is equal to or brighter than the current pixel, then it is 1. The value of each neighbor is concatenated together to create a code for each pixel. Then, a histogram is generated, using the pixel encodings, and each code is sorted into the bins of the histogram, which becomes the new feature space for the image. Effectively, what this does is capture the relative quantities of edges and corners (or lack thereof) in each image.

Clustering and Dimensionality Reduction Once we had 3 feature vectors for each graph, representing keywords, dimensions used, and visual similarity, we needed to cluster based on these attributes, and to dimensionally reduce them to a 2d space, in order to visualize them. First, we normalized all values to be between 0 and 1. Then, each of these vectors are multiplied by a user controlled weight, via sliders in the visualization, ranging from 0 to 1. Then we concatenate them all together to make one large feature vector for each image.

To cluster, we used K means on the concatenated feature vectors, which is the same technique employed by the paper. However, in the paper, to map the charts to the 2d space, they measured the aggregate pair-wise distance of each chart's features to all others. Then, they used principal component analysis (PCA) on this distance measurement to find a 2d mapping. In our implementation, we decided to instead use PCA on the entire concatenated feature space for all charts, to find the top two principal axes for graphing. Using PCA on the entire feature space seemed like a better choice to us, because it is based on the distance between points in the feature hyperspace, as is K means, and so our clustering and dimensionality reduction are based on the same features, so in effect the clustering will more accurately align with the 2d data.

Implemented Visualization and Exploration Tools The main view of the Chart Constellations visualization system is what the original authors call the "Collab View". We implemented this using a scatterplot in which each chart was plotted as a coordinate point based on its top two principle components; this closely mirrors the choices made in the original paper. Where our "Collab View" differs is in some of the visual encodings. Because our Dash framework did not allow us to create bubble sets within the rendered scatterplot, we elected to encode cluster membership via color; then, because color was originally used to encode authorship, we chose to allow users to explore authorship of charts by applying optional filters to the scatterplot.

We used treemap to show which attributes in the initial dataset, Student Alcohol Consumption, has been used in different charts to visualize data. Each block of the treemap corresponds to an attribute in the dataset. The size of each square shows how much this attribute is used in different charts. For example, G3, has the largest square and is used in many charts. We also used color

to emphasize the amount of contribution of attributes. The amount of their frequency in each chart can be seen by hovering on the square or looking at the numbers beside the color bar. The latter one gives an approximate number of frequency. To find the number of frequencies we used "represented_features_embedding" feature in our dataset which corresponds to the embeddings of attributes in the initial dataset. Selecting an attribute in the treemap highlights any data point in the scatter plot with an asterisk symbol to show which charts are using the selected attribute as one of their dimensions.

One issue in our implementation regarding treemap was about clicking on an attribute and highlighting any intersecting dimension with the selected attribute. We could find the list of intersecting dimensions with the selected attribute, but the serious problem of treemap in dash, opposed to JavaScript, is that by clicking on an attribute, it expanded and dominates whole the treemap and doesn't let us see other attributes which cause not to see which attributes have been highlighted in response to the selected one. Therefore, we ignore this part of the visualization.

3 Evaluation and Analysis of Results

PCA based on full feature set By using PCA on the full feature set, rather than the aggregate pair-wise distances, visually it appears that the results of our K means clustering better aligns with expectations, based on relative location in the scatterplot. This fits with our hypothesis, that by having PCA and K means clustering operate on the same features, their results would be more visually comparable.

Analyzing effectiveness of local binary patterns Binning the results of the local binary patterns via a histogram turned out to be a decent metric for measuring visual similarity between images, although it has some flaws. For one, by far the most common case is when a pixel is surrounded by neighbors that are all the same color, and this is given the code 255 (8 bits of 1 for each neighboring pixel). When the results were normalized to between 0 and 1, all of the "interesting" codes were minimized, and the 255 bin was given a value of nearly 1. To fix this, the case of 255 was removed. However, even removing this outlier, the histograms of all the graphs ended up being similar enough that their euclidean distances in the dimensional space was dwarfed by the keyword word2vec encoding, and the one-hot encoding of the dimensions used. In order for the PCA and K means clustering to be influenced significantly by the local binary pattern histograms, the weights of the other two vectors must be made much smaller. In the future, this could be fixed by increasing the weight of the visual similarity vector, or by finding an encoding method that produces more diverse values.

The "Collab View" Our departure from the visual encoding choices made in the original work yielded some benefits. Firstly, using bubble sets to encode cluster membership in a more densely populated scatterplot has the potential to become a visual mess. Color is a more extensible encoding in this respect and lends itself more readily to visual perception in this usage case versus encoding authorship; this is because we expect clusters to be embedded nearer one another in the two-dimensional space whereas authors may have multiple charts that are clustered in vastly different groups. Secondly, the original paper mentions that using colors to encode authorship reduces the number of authors that can be accurately represented in the collab view so, by doing away with this encoding, we remove this restriction. This also more realistically fits a use case in which a user

would only really be interested in analyzing the longitudinal results of one or a few authors at a time rather than all of them at once.

4 Conclusion

Overall, we have succeeded in our goal, of implementing the core functionality of the chart constellations in dash, and extending its performance by doing the PCA on the entire vectorized feature space, and by using local binary patterns to encode visual information of each chart. Additionally, we have improved the scalability of the "Collab View" via our choice of visual encodings that remove limiting factors (i.e. limited ability to distinguish between hues limiting number of potential authors, visual confusion of bubble sets when increasing number of clusters or density of the scatterplot).