ECS 272 Final Project Report

Weidong Guo
University of California Davis
wdguo@ucdavis.edu

Baotuan Nguyen
University of California Davis
btnguye@ucdavis.edu

1. Project Overview

In this project, we implement the tool from "Visual Abstraction and Exploration of Multi-class Scatterplots." [2] This work looks to overcome occlusion for very large datasets where the limited space of scatterplots becomes a challenge. This is problem is exacerbated when there are different classes in the data, resulting in overdraw. In such a situations, the drawing order determines which class shows up on top, leaving earlier classes hidden beneath the later drawn data points. The paper explores sampling techniques to abstract large datasets while still giving users good intuition into the original class distribution. Furthermore, the researchers have implemented a dashboard that provides various tools to better explore the sampled data. This includes functionality such as the ability to introduce or remove certain classes from the scatterplot, the ability to brush and apply different types of color blending, and the ability to switch back and forth between sampled and unsampled versions of the data. Our goal is to sample our chosen dataset(s) in the same manner and implement a similar dashboard in D3.

2. Our Experience

We found the sampling technique to be the most crucial, and also the most difficult portion of implementing this work. This work utilized a multi-class blue noise sampling technique from Wei et. al [3]. Following this reference, we found the technique to be an extension of an earlier work by Bridson et. al [1], which is a fast method of calculating poisson disk sampling (single class). The multi-class variant presented by Wei, at least at a high level, iteratively applies the poisson disk sampling per class to maintain an equal fill rate between classes and ensure blue noise characteristics (random, but uniformly distributed data) for both the entire dataset, as well as within each class. However, critical details in the algorithm, such as calculating inter-class minimum radius values and per class sample targets were not explained well. We could not simply use the values provided in the paper since those were specific to their dataset. Furthermore, there were not many resources available. Wei did provide source code, however it was targeted for Visual Studio 2003, which is no longer compatible with more modern versions of the IDE. Thus, given these constraints, we implemented the simpler Bridson technique, which we were able to find many resources and examples upon. This technique still showed decent results in our dataset. This perhaps due to non-ideal characteristics of our chosen datasets, which we will further explain in a later section.

For the visualization implementation, we mainly used d3.js and Bootstrap. We used d3.js to create and manipulate SVG elements to draw scatter plot. We also implemented zooming and brushing. Zooming is essentially another approach to solve occlusion because as we zoom in, the size of the data points remain the same, while the gaps between them enlarge. Brushing is a way to select data points and view them with a chosen blending mode. Finally, we used BootStrap to enable tootip to show details about each data point. It also let us build beautiful navigation bar and dropdown menu.

We appreciate the handy functions that d3.js provide such as allowing us to register callbacks for zooming and brushing event, so then we could update our view accordingly. We like how we can bind each element in DOM with data, so we can visualize each piece of data easily. Bootstrap provides really neat design of menus and dropdowns, which saved us so much time. Due to the good documentation of d3.js and Boostrap, we can focus more on the design of our system rather than spending lots of time understanding how to use the function calls.

On the other hand, getting zooming and brushing to work together was not straightforward. If we allow zooming, brushing are not not really selecting the points enclosed by the brushed region. It was not obvious to us that we needed to add the zoomed offsets in order to locate points that were enclosed by the brushed region. After carefully examining the code through the Chrome Developer Debugger, we were able to spot that it was necessary to consider the offset into the calculation.

3. Dataset

Our requirements for a dataset for this project were bivariate (at least two available numerical features to plot on a 2D scatterplot) and multi-class data that would have significant occlusion and inter-class overdraw when plotted. Given the relatively lower scope of work required in implementing the visualization components of this tool, we selected two datasets upon which to experiment. The first is a dataset on trending YouTube Video Statistics[1]. Information for each video includes likes, dislikes, comment counts, view counts, video category, upload date, url, and thumbnail image link. The second dataset was health inspection data for San Francisco restaurants[2]. The dataset contains individual records for each inspection violation. Thus a restaurant may have several entries, one for each type of violation encountered during an inspection. We aggregate these individual violations to get a summary for each restaurant, for which we use the overall inspection score as a class (i.e. excellent health score range, good, average, poor, etc.).

Compared to the datasets explored in the paper, we realized that our data was not as uniformly occluded. Though we had many areas in both of our datasets where there was high occlusion, there was also a lot of sparsity. This was especially true for the YouTube data, which had many outliers that left large amounts of white space on the scatterplot. This meant that our selected data was not the best examples to demonstrate the advantages of the multi-class blue noise sampling as utilized in the paper. This sampling technique extracts random, but uniformly distributed samples from the source distribution, which should be uniformly distributed and highly occluded. In hindsight, selecting a dataset that is more spatially confined, such as the basketball dataset experimented on in the paper would have been ideal.

4. System Functionality & Interactions

Our dashboard starts out with on the San Francisco Health Inspection dataset. Please refer to Figure 1 for an example screenshot of our interface. To the right of the screen are smaller scatterplots for each class, which for the SF dataset, are groupings for excellent, good, average, and poor inspection scores. Click on one of these scatterplots to select that class on the main scatterplot view. This allows for easy comparison of a few classes at a time. If you wish, you can click on all classes to view the entire dataset at once. Clearly there will be a lot of occlusion and overdraw if you do! Thus in the top menu, we have an option to select different levels of sampling. See Figure 2 for a before and after comparison of sampling on the SF dataset. We provide to options for various r values, where a larger r will result in more abstraction (resulting in less points and better clarity) while a smaller r will retain more of the original data, albeit with more occlusion. You can zoom on the main scatterplot by swiping up and down on your touchpad with two fingers. Drag your mouse along the outside of each axis to shift the view in a certain direction. You can brush across a set of points to apply a blending mode (see Figure 3). The different color blending modes can be selected via the buttons at the top of the tool. Finally, clicking

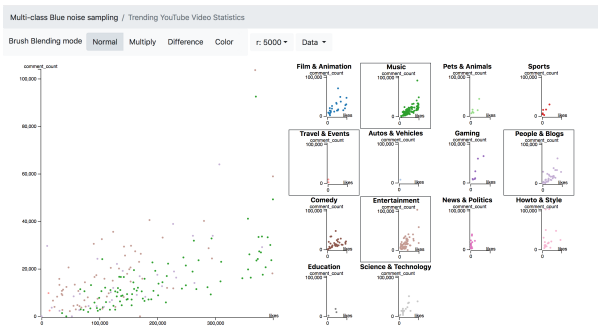

Figure 1. Overview of our interface with the YouTube dataset. Leftside is main scatterplot view. Rightside contains per class scatterplots, which when clicked on can be toggled on or off in the main view.

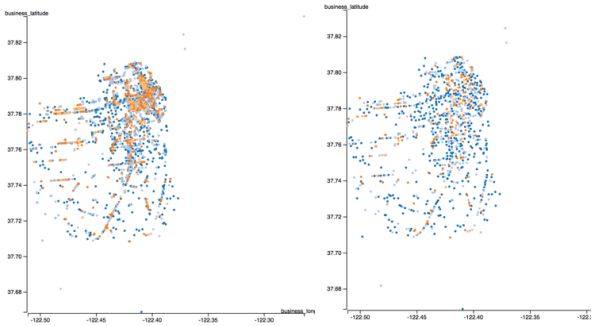

Figure 2. SF Health Inspection Dataset Results, Left: Unsampled scatterplot, Right: Results after sampling, less occlusion/overdraw

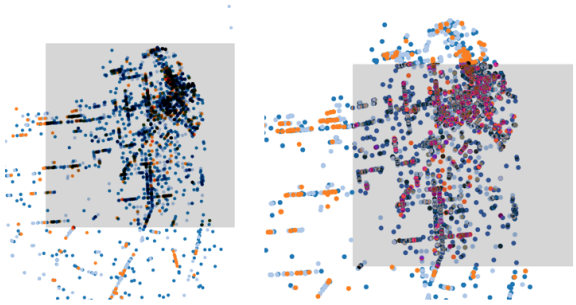

Figure 3. Brushing functionality allows users to select a group of points and apply a specific color blending technique. (Multiply techinque on the left and color difference on the right)

[1] https://www.kaggle.com/datasnaek/youtube-new

[2] https://www.kaggle.com/jonathanbouchet/san-francisco-restaurants-inspection

on a specific data point on the scatterplot will bring a tool tip where you can view specific data on each data point. The last drop down menu allows users to switch to other datasets. The YouTube dataset will have a similar interface, however, containing many more classes. Additionally, click on each datapoint will bring up a tooltip from which you can directly play the YouTube video (See Figure 5). Again, play with different r-values to see how the abstraction works. As you'll notice, picking a higher r-value will cause the sparsest classes two disappear. This will be explained further in the following section.

5. Results & Discussion

We found that the poisson disk sampling method does provide a good abstraction of the data. To evaluate this, we compared each class in our tool with and without sampling. The general ratio and relative distribution of classes is held constant between sampled and unsampled versions of the dataset, while significantly reducing occlusion and overdraw. One issue we experienced on the YouTube dataset however, is that in selecting a larger r (the parameter that controls the minimum acceptable distance between sampled points) results in a few classes being dropped out. After some inspection, we realized this is due to the proximity of the sparse classes. For example in the categories "shows" and "non-profits", each class only has a few datapoints, and these points are in very close proximity with each other. So if we select a large enough r to prevent occlusion at the default zoom level (i.e. r = 10,000, measured in units of the euclidean space between the likes and comment count axes), than selecting a point from one class, will necessary result in points from the other class being rejected during sampling. This can be seen in our tool, when switching to the r = 10,000 sampling. Thus, the sparsity and non-uniform proximity in this particular dataset was an issue. We felt this was a fair tradeoff and thus allow users to be able to select a higher level of abstraction, albeit losing information on these less populated classes. See figure 4 for the differences between different r values.

In evaluating the visualization tool, we found the features implemented to be very useful in our exploration of the data. We found the class selection to be especially useful, especially for datasets such as the YouTube one which had many classes resulting in significant clutter when viewing all classes together. Being able to view a certain selection of classes at once gave much more clarity. We were able to gain some interesting insights from the data. For one, music videos often have many likes, but fewer comments. One would expect that users would listen to music, perhaps while working, and not be inclined to comment. Some music videos may have comments disabled altogether. Other video categories such as gaming and politics, show much more comments and less likes, as the audiences for these

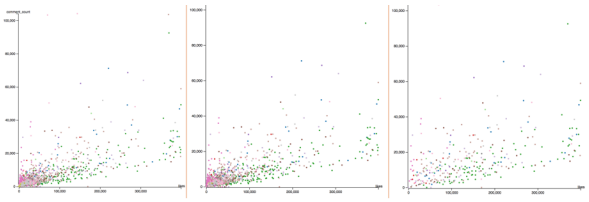

Figure 4. YouTube Dataset, Left: Unsampled, Middle: Sampling at r = 500, Right: Sampling at r = 5,000. In order to keep all 16 classes represented, r=500 was the largest value we could use. This granularity of sampling does not improve occlusion over the unsampled data.

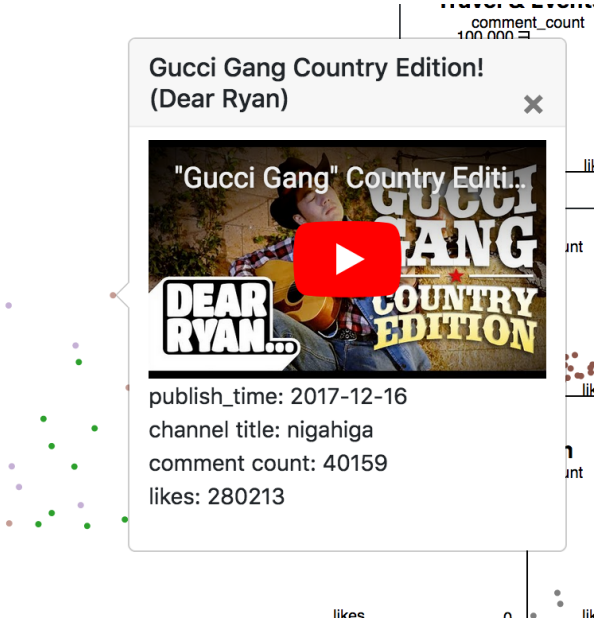

Figure 5. Users can click on datapoint to view video specfic information as well as play the video within the tooltip.

types of videos are often very involved and opinionated respectively.

There were several components of the paper that we did not implement, such as persisting views and annotations. We felt this required more web dev work, and was not entirely pertinent to the course. Another aspect was color optimization, which we frankly did not have enough time to get to.

6. Extra Credit

We implemented a tooltip on each YouTube video data point on the main scatterplot that allows users to quickly see other video statistics such as title, publisher, publish date, view count, and even play the video directly embedded within the tooltip.

References

[1] R. Bridson. Fast poisson disk sampling in arbitrary dimensions. In *ACM SIGGRAPH 2007 Sketches*, SIGGRAPH '07, New York, NY, USA, 2007. ACM.

[2] H. Chen, W. Chen, H. Mei, Z. Liu, K. Zhou, W. Chen, W. Gu, and K. L. Ma. Visual abstraction and exploration of multi-class scatterplots. *IEEE Transactions on Visualization and Computer Graphics*, 20(12):1683–1692, Dec 2014.

[3] L.-Y. Wei. Multi-class blue noise sampling. *ACM Trans. Graph.*, 29(4):79:1–79:8, July 2010.