Visualizing Linguistic Diversity in Vancouver

Roger Yu-Hsiang Lo, Namratha Rao, and Anika Sayara
roger.lo@ubc.ca, rao.namratha@gmail.com, sayanika@cs.ubc.ca

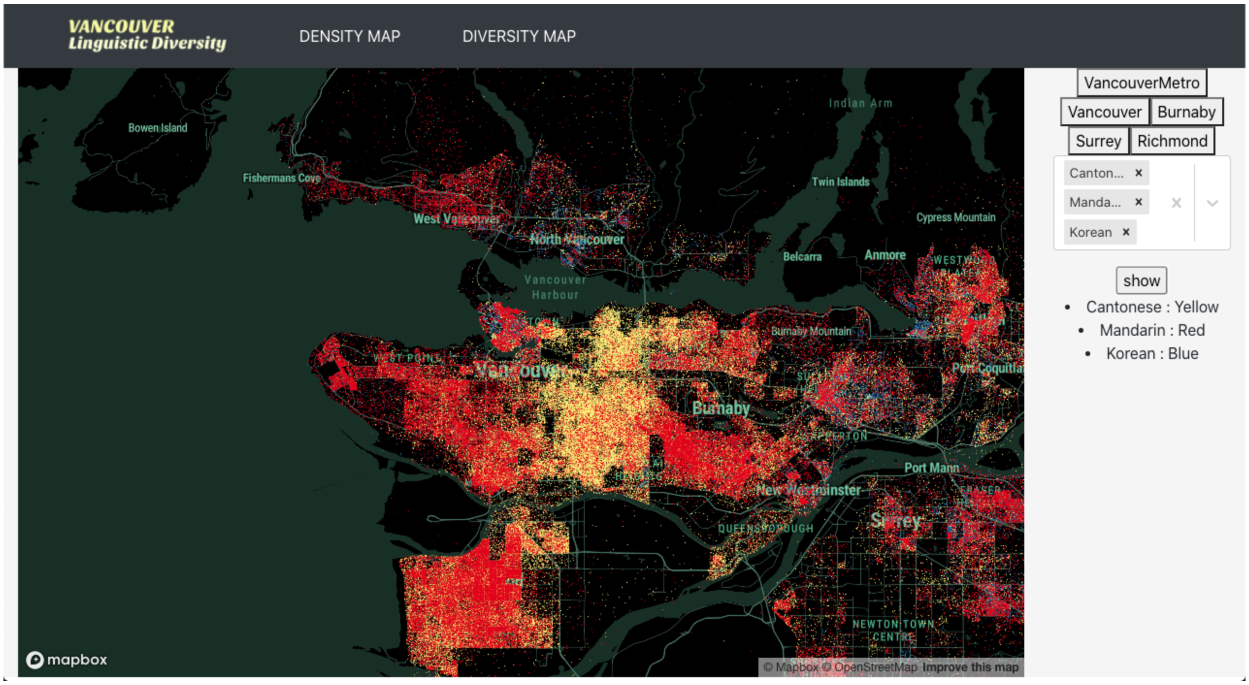


Fig. 1. View for interacting with the dot density map. Individual speakers are represented by dots in this visual idiom, with different languages being distinguished by colour hues. The user can zoom in to a specific neighbourhood by clicking on one of the buttons on top of the control panel to the right, and they can choose the languages to be displayed on the map by using the selection widget in the middle of the control panel.

Abstract— Canada prides itself on its multicultural society, and one hallmark of multiculturalism lies in the diversity of languages used by its members. The current project aims to provide a means for the user to take a glance at the linguistic diversity of Metro Vancouver. We utilize a choropleth map for an overview of linguistic diversity and a dot density map for visualizing the density differences and clusters of languages spoken across Metro Vancouver. The detailed linguistic information associated with a neighbourhood is summarized in a collapsible tree.

1 Introduction

With two official languages, diverse First Nation languages, and a long and rich history of immigration from a wide range of countries and cultures, Canada has now been home to a large number of languages. In fact, according to a ranking by UNESCO in 2009[1] [13], Canada ranked 79th in linguistic diversity, only after Belgium, Israel, Norway, Latvia, and Italy among the Organisation for Economic Co-operation and Development (OECD) countries.

This diversity in languages is especially prominent in a handful of Canada's largest cities, with Metro Vancouver being one of the hubs that attract a sizable number of international migrants. With immigrants comes new languages—it has been a common experience for Vancouverites to constantly brush shoulders with fellow residents who can speak a different language that they do not understand. In fact, it is also impossible to find a person in Metro Vancouver who does not have any experience with languages other than English or French.

However, the distribution of languages in Metro Vancouver is not random, as immigrants of the same ethnicity or original country tend to cluster around specific neighbourhoods. Many questions then can be asked about this *nonuniform* language distribution. For example, if one walks in different cities in Metro Vancouver, in which city is one most likely to meet someone who speaks a different language? Or a Bengali speaker might wonder to what extent is his/her language spoken in different neighbourhoods in Metro Vancouver. These questions can be easily answered with census data, but only if the person has substantial skills in extracting and wrangling raw data. The aim of this project is to create a visualization that lowers this bar and allows the user to find answers to these questions, and potentially further explore the linguistic landscape of Metro Vancouver. We hope that the visualization tool developed in this project will lower this bar and be of interest to linguists who want to understand the language distributions in Metro Vancouver and laypeople who perhaps wonder what language the sounds that people around them are constantly making belong to.

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

[1] This is the latest report we can find online.

2 Related Work

To the best of our knowledge, there has been little work directly addressing the visualization of language diversity. We extend our literature review to include visualization research that targets similar problems around geospatial data and related visualization solutions.

2.1 Cartographic Projections

Most of the previous works we identified are mainly from the field of cartography. For example, McNew, Derungs, and Moran [8] overcome the prominent biases in linguistic data and cartographic projection that tend to lead to visual illusions through the Eckert IV projections and Voronoi/Thiessen tessellations to model linguistic areas, instead of the more traditional languages-as-points approach. Luebbering, Kolivras, and Prisley [7] use the language diversity index [5] to study the extent of the presence of different languages in an area, based on the language categories of the U.S. Census.

2.2 Temporal Visualization Techniques

Geospatial datasets are often unstructured. However, the locations, directions, distances, size, and altitudes in the dataset can give them an inherent positional structure and shape. Shanbhag, Rheingans, and deJardins [10] deal with the complexity of partitioning geospatial data by using a combination of three temporal visualization techniques—wedges, rings, and time slices to visualize the changing patterns of population data while maintaining the spatial boundaries of the map.

2.3 Self-Organizing Map

In contrast, Skupin and Hagelman [12] propose trajectory-based techniques that represent change in the attribute as movement of objects across a 2D Self-Organizing Map (SOM) surface to analyze population census data. The spatial location of the data is distorted by computational changes to make developed patterns visible. As the quantity and diversity of geospatial data increase, analysis becomes difficult due to their size and hidden patterns.

2.4 Artificial Neural Networks

Artificial neural networks are an evolving solution for data analysis and pattern recognition. Koua [6] uses an SOM algorithm to uncover the structure and patterns from the geospatial dataset consisting of a collection of socio-economic indicators related to municipalities in a region of the Netherlands. He explores visualization techniques like mesh, multiple linked views of component planes, and 2D/3D surface plots of distance matrices that use spatial metaphors such as distances, regions, and scale.

2.5 Dot Density Maps

Dot density maps visualize the geographic distribution of either univariate or multivariate count data. Walker [15] discusses pointillist interactive dot density maps, which map the colour of dots to a set of categorical values, allowing the visualization of both the areas of categorical homogeneity and areas of diversity as the colours for different categories blend. He outlines techniques like dasymetric dot mapping, zoom-dependent data and styling, interactive filters, and linked charts to summarize data on the map to improve the user experience. This drives our design consideration for the interface envisioned.

Our project was inspired by Dmitry Shkolnik's work in trying to identify most linguistically diverse regions in Canada, using the language diversity index [11]. Even though he did provide maps that visualize the results, these maps are all static and only depict aggregated results. Our proposed visualizations expend upon these basic maps and add interactivity that allows the user to retrieve more detailed information.

3 Data and Task Abstraction

The data and task abstraction described below follows the framework detailed in Munzner [9].

3.1 Data Description and Preprocessing

Our visualization relies on two datasets—2016 Canada Census data and language family tree data, which we describe in detail below respectively.

3.1.1 Language Usage Data

The language usage data comes from the 2016 Canada Census and is derived from the answers Vancouverites gave to the question of what language one uses most often at home. The dataset is retrieved from the R package `cansensus` [14]. The extracted dataset is stored in the GeoJSON format and includes a total of 3450 data entries and 230 attributes. Each data entry represent a single dissemination area (DA), which is a small, relatively stable geographic unit that houses from 400 to 700 people. It is the smallest standard geographic area for which all census data are disseminated[2]. In the original dataset, each DA item consists of eight categorical and 222 quantitative attributes. Among the categorical attributes are identifiers, such as Census Tract Identifier and Census Division Identifier, that are irrelevant to the current project and are therefore ignored. Only three attributes—Type, Region Name, and Geometry—are pertinent, the descriptions and sample values of which are given in Table 1. Four out of the 222 quantitative attributes encode information about the number of households, dwellings, residents, and area in square kilometer in each DA respectively. The other quantitative attributes encode the numbers of speakers of different languages, with each attribute corresponding to one language in principle. More detailed information concerning the quantitative attributes is given in Table 2.

The raw language usage data is used to derive the Linguistic Diversity Index (LDI) [5], which is calculated as follows:

$$LDI = 1 - \sum_i p_i^2,$$

where p_i is the proportion of population in a DA that uses language i. LDI measures the probability that any two speakers in a population will speak the same language, with a value of 1 implying that no two individuals share a language and a value of 0 indicating that everyone uses the same language.

For the implementation of the dot density map, we generate a JSON file containing 2,443,540 entries in the form `[latitude, longitude, language_code]`. Each entry in the file represents a dot on the map, where the longitude and latitude values give the coordinate of the dot and the `language_code` is the the code for the language it stands for. For each language in a DA, we generate n number of random points within the DA, where n is the number of users of the respective language within that DA. This is done for the following two reasons:

- Pre-calculating the coordinates of the dots and storing them in a file allows us to simply parse the file and place the dots on the map without having to compute the coordinates in run-time. This makes the implementation considerably faster.
- We use the `ScatterplotLayer` of `deck.gl` for plotting dots on the map. The prop data of a layer in `deck.gl` [16] specifies the layer's data source for visualization. The data prop of the `ScatterplotLayer` in particular receives an array of paired latitude and longitude points and renders them as circles with a certain radius. Hence the JSON file generated has the format `[latitude, longitude, language_code]`.

3.1.2 Language Family Data

The language family data is in the JSON format and is constructed manually from the mother tongue and home language classification used for the 2016 Canada Census, a snippet of which is shown in Figure 2. The classification is first broken down into non-official and official languages, with the former being further broken down into aboriginal languages and non-aboriginal languages while the

[2] https://www12.statcan.gc.ca/census-recensement/2016/ref/dict/geo021-eng.cfm

Table 1. Categorical attributes in the dataset.

Name	Description	Cardinality	Sample value
Type	Constant value field indicating dissemination area (DA)	1	`'DA'`
Region Name	The name of the neighbourhood the DA is in	25	`'Richmond'`
Geometry	Polygon geometry of the DA	3450	`POLYGON ((-123.28147116478 49.36803241352), ...)`

Table 2. Quantitative attributes in the dataset. Note that there are 222 attributes with the name v_CA16_XXXX. We only list one such attribute for illustration.

Name	Description	Min	Max	Median
Households	The number of families in the dissemination area (DA)	0	4923	213
Dwellings	The number of dwelling units in the DA	0	5631	229
Population	The total number of people in the DA	0	8778	586
Shape area	Area of the DA in sq. km.	0.002	846.8	0.1
v_CA16_1367	The number of speakers of language 1367 (French)	0	85	0

```
{
    "name": "Classification",
    "children": [
        {
            "name": "Official",
            "children": [
                {"name": "English"},
                {"name": "French"}
            ]
        },
        {
            "name": "Non-official",
            "children": [
                {
                    "name": "Aboriginal",
                    "children": [
                        {
                            "name": "Algonquian",
                            "children": [
                                {"name": "Blackfoot"},
                                ...
                            ]
                        },
                        ...
                    ]
                },
                ...
            ]
        }
    ]
}
```

Fig. 2. A snippet of language family data.

latter branching into English and French. Non-aboriginal languages are subdivided into 19 language groups, and aboriginal languages into 13 groups. These language groups then fork into a total of 214 individual languages. Therefore, in terms of data structure, leaves or terminal nodes represent individual languages, while non-terminal nodes represent language groups. Note also that we use *language group*, *language branch*, and *language family* interchangeably in this paper. The detailed breakdown can be found at `https://www12.statcan.gc.ca/census-recensement/2016/ref/dict/figures/f3_3-eng.cfm`.

3.2 Task Abstraction

The main users we have in mind are the general public. We envision the users of our visualization come with questions like 'What are the neighbourhoods in Metro Vancouver that are most diverse linguistically?', 'How many languages are spoken in Surrey? And what are the numbers/proportions of speakers of these languages?', or 'How are Korean speakers distributed in Metro Vancouver?'. More abstractly, our visualization aims to support queries in two directions: (i) given a geographic location—a neighbourhood in our case—find out the distribution of languages spoken in this location, and (ii) given one or more languages, find out the geographic distributions associated with these languages.

To abstract our task into a higher level, the goal of our visualization is to provide functionality for the user to *discover features* from a geometry dataset and hopefully to *derive some enjoyment* along the process. As we do not constrain the type of search actions the user can perform, the user should be able to carry out tasks that involve *lookup*, *locate*, *browse*, and *explore*. At the lowest level, the user can retrieve information by *identifying specific targets*, such as finding the number of speakers of a language in a particular neighbourhood, or comparing across targets, for example, by comparing the numbers of speaker of a particular language among several neighbourhoods.

4 Solution

This section is organized as follows: we first describe the overall interface before diving into the three major components of the interface—a diversity map, a density map, and a language family tree. For each component, we provide justifications for the chosen visual idiom over potential alternatives. We briefly describe the encoding of each visual idiom and give a high-level account of supported interactions. The detailed implementation of each visual idiom will be explained in Section 5.

4.1 The Overall Interface

The interface consists of two views—the density map and the diversity map. The user can switch between the views by clicking on the respective tab on the top navigation bar. The density map contains a dot density map along with a control panel, as shown in Figure 1 and the diversity map contains a choropleth map along with a movable and collapsible panel that encloses a language family tree, shown in Figure 3. As addressed in the task abstraction section, these two views together support interactivity going in two directions: (i) the user can select a neighbourhood on the diversity map, and the linguistic information of the selected neighbourhood will be updated in the language family tree, and (ii) the user can pick some languages with the control panel in the density map view, and the geographic distribution of these languages will be depicted on the density map. The detailed description of interactivity will be given in Section 5.

4.2 The Diversity Map

The purpose of this map is to visualize the magnitude of LDI associated with DAs. There are three well-established thematic map types for quantitative data on the ration scale—choropleth maps, cartograms, and proportional symbol maps. We use a choropleth map instead of the

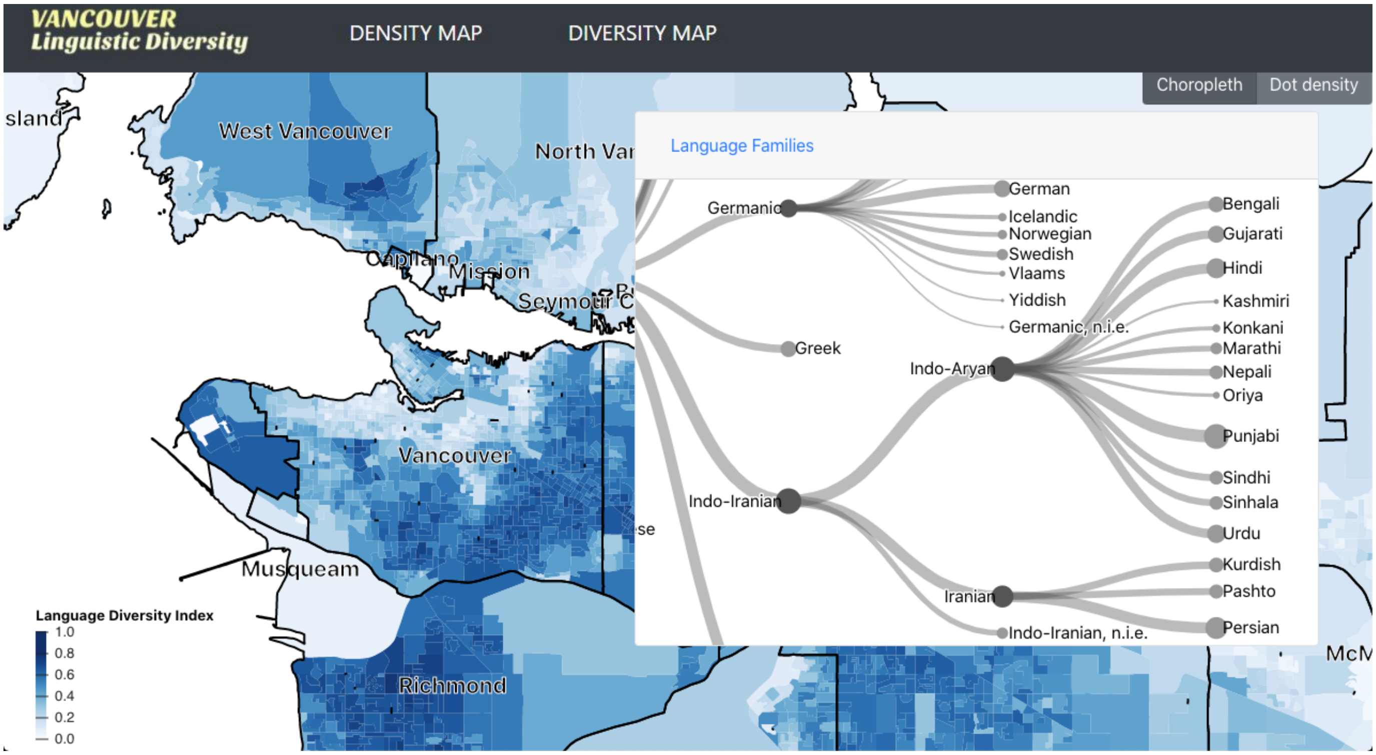


Fig. 3. View for interacting with the diversity map. The value of the Language Diversity Index associated with each dissemination area is encoded with a sequential colour scale. The map offers basic zooming and panning functions. In addition, the user can select one of the neighbourhoods marked on the map, and the linguistic information of the selected neighbourhood will be shown in the language family tree. Only the languages present in the selected neighbourhood will be plotted as nodes in the tree.

other two alternatives because the limitations posed by cartograms and proportional symbol maps are more severe than those of choropleth maps in our case. Specifically, cartograms deform the underlying regions, which can make the map unrecognizable when the data value deviates greatly from the original area of a region, or if some data points are missing for some regions. Our derived LDI data falls into both problems, and hence the option of a cartogram is ruled out. In addition, given the large number of DAs (i.e., 3450 DAs) concentrating in a samll area in our dataset, a proportional symbol map will appear extremely cluttered with many overlapping symbols. Choropleth maps therefore emerge as the best option, though there are still limiations, as will be discussed in Section 8.

The resulting choropleth map is shown in Figure 3. In our map, DAs are delimited as area marks, with their shapes reflecting the underlying geometry. The area marks are shaded with a sequential colourmap, given that LDI only spans a positive range from 0 to 1.

The user interactivity offered by the choropleth map includes panning, zooming, and selection. Panning and zooming with a map is a straightforward operation for many users. The mouse-click selection function supports selection at the *neighbourhood* level: when the user selects one of the preset neighbourhoods—as singled out with texts—in Metro Vancouver, the selected neighbourhood will be centred in the screen, and the language information of the selected neighbourhood will be updated in the language family tree view. Clicking elsewhere on the map will bring the user back to the default map and tree overviews.

4.3 The Density Map

The goal of this map is foreground the geographic distribution of various languages in Metro Vancouver, for which we opt for a dot density map. Dot density maps are a technique for visualizing the geographic distribution of either univariate or multivariate count data, where each dot can represent either a single (i.e., one-to-one dot-to-data ratio) or multiple data points (i.e., one-to-many dot-to-data ratio). Typically, dot density maps are realized as *pointillist* maps, in which the colour of dots is mapped to a set of categorical values. Pointillist maps allow for not only the visualization of the distribution of a geographic phenomenon but also the consideration of how this distribution varies among subcategories. Pointillist maps are therefore suited to showing the internal heterogeneity of areal units and to illustrating smoother demographic transitions between neighbourhoods.

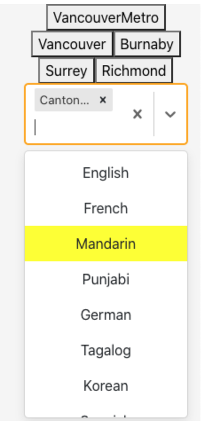


Fig. 4. Control panel for the density map view. The buttons on the top indicate which neighbourhood to zoom in to. The selection widget in the middle dictates which language(s) to be included in the map.

In our implementation (see Figure 1) of the dot density map, we use the one-to-one dot-to-data ratio approach where each dot represents a person and the colour of the dot signifies the language spoken by the person.The legend that appears in the control panel provides user with the information on what colour corresponds to what language.

The user interactivity offered by the density map view includes zooming and selection. The user can use the provided control panel as illustrated in Figure 4 to interact with the density map. Specifically, the user can select a neighbourhood from the control panel to zoom in on the selected neighbourhood as shown in Figure 5 or can select one or more languages to be displayed on the selected region of the map. We put a threshold of five languages to be shown simultaneously to avoid visual occlusion caused by dots overlapping on one another.

4.4 The Language Family Tree

Given that the goal of this view is to summarize the linguistic profile of a selected neighbourhood and that there are more than 200 languages presented in Metro Vancouver, displaying all languages with a flat structure is more likely to overwhelm the user than to inform them. One way to aggregate the raw data is to realize that many languages are genetically related to each other and to group languages into branches/families based on this relatedness.

Capitalizing on the fact that language families assume a hierarchical tree structure, a natural way to visualize this structure is through a tree diagram, with leaves representing individual languages and non-terminal nodes different language families or branches within a language family. However, due to the depth of the language family trees (i.e., the maximal depth of 7) and to avoid clutter the view, we opt for a collapsible tree, with only a couple of branches expanded by default. The user can manually expand other branches if they so wish. We implement the collapsible tree in the *tidy tree* layout, as opposed to the cluster dendrogram or the radial layout, because tidy trees are visually more compact.

The tree is designed in a way such that the radius and the width of nodes and links encode the number of speakers of the language/language branch represented by the node or link. Note that we associate a link with the child/target node, as opposed to the parent/source node, when size-encoding its width. Also, instead of mapping the radius/width linearly to the number of speakers, we map the size to the *log-transformed* number of speakers in order to make nodes/links more comparable among other nodes/links. However, we pay the price that log transformation effectively compresses the actual difference and can potentially convey a distorted view. We further disseminate this point Section 8. The nodes are also colour-coded according to their type, such that nodes containing children are of higher saturation. An example of the tree is presented in Figure 6.

The tree provides two types of interactivity: (i) collapse-expand (see Figure 7): when the user clicks on a node that has child nodes, the child nodes collapse into the parent node or expand from the parent node according to the current configuration, and (ii) hover (see Figure 8): when the user hovers over a node, a tooltip pops up, showing the number of speakers associated with the language or language group represented by the node. This interaction is in part to counteract the magnitude compression introduced by log transformation.

5 Implementation

The solution is implemented as a web application, built with HTML, CSS, Javascript, and associated libraries. We use Javascript library `D3` [4] for the diversity map and the language family tree visualization, and `deck.gl` [16] and `react-map-gl` (`https://visgl.github.io/react-map-gl/`) for implementing the density map. Python and R are used for data preprocessing.

5.1 Overall Interface

The overall interface, including the navigation bar and various panels, is built with CSS framework `Bootstrap` and Javascript library `React`.

5.2 The Diversity Map

The diversity map is built using `D3`; areas are projected with the Mercator projection implemented in the `d3-geo` library, and the magnitude of LDI associated with each DA is encoded with a sequential colour scale. The resulting map is displayed in Figure 3.

The code for adding interactions is largely based on an Observable notebook authored by Bostock [3].

5.3 The Density Map

The density map is built using `deck.gl`, together with `react-map-gl` for rendering of the base map. In particular, we use the `ScatterplotLayer` of `deck.gl` to render the dots on the map.

Even though the code examples of `ScatterplotLayer` is available[3], we make use of it for learning purpose only and have to write our own way of implementing the `ScatterplotLayer` to achieve better performance during user interaction and to adapt it according to the need of our application. Our implementation uses `React State Hooks` for state management, allowing dynamic allocation of colours to languages and control of the opacity of the dots that are rendered based on user selections. The `getFillColor` prop of `ScatterplotLayer` takes colour value in `[R, G, B, A]` format where the value of `A` ranges between 0 and 255, 0 being transparent and 255 being opaque. When the application first loads, we render all dots with an alpha value of 0, and later each time the user makes a new language selection, we assign a colour from a custom colour palette and change the alpha value to 255, making the dots of the selected languages opaque, while keeping the rest of the dots transparent. This approach of implementing the density map avoids creating and rendering a new layer or reloading the page each time a new selection is made.

5.4 The Language Family Tree

The language family tree summarized language relatedness and numbers of speakers is constructed with `D3`'s `d3-hierarchy` library. A partial view of the tree is shown in Figure 6.

The distance between adjacent sister nodes is determined by the size of the nodes in question (x, y), which is in turn determined by the number of speakers (ns) associated with the node using the following function:

$$\mathrm{Dist}(x,y) = \begin{cases} \ln(ns(x)+ns(y))/7.5 & \text{if } ns(x) >= 50 | ns(y) >= 50 \\ 1 & \text{else} \end{cases}$$

The code for collapsing and expanding nodes is modified from an Observable notebook written by Bostock [2].

6 Results

In this section, we provide two scenarios of use that we envision for the user of our visualization.

6.1 Scenario 1

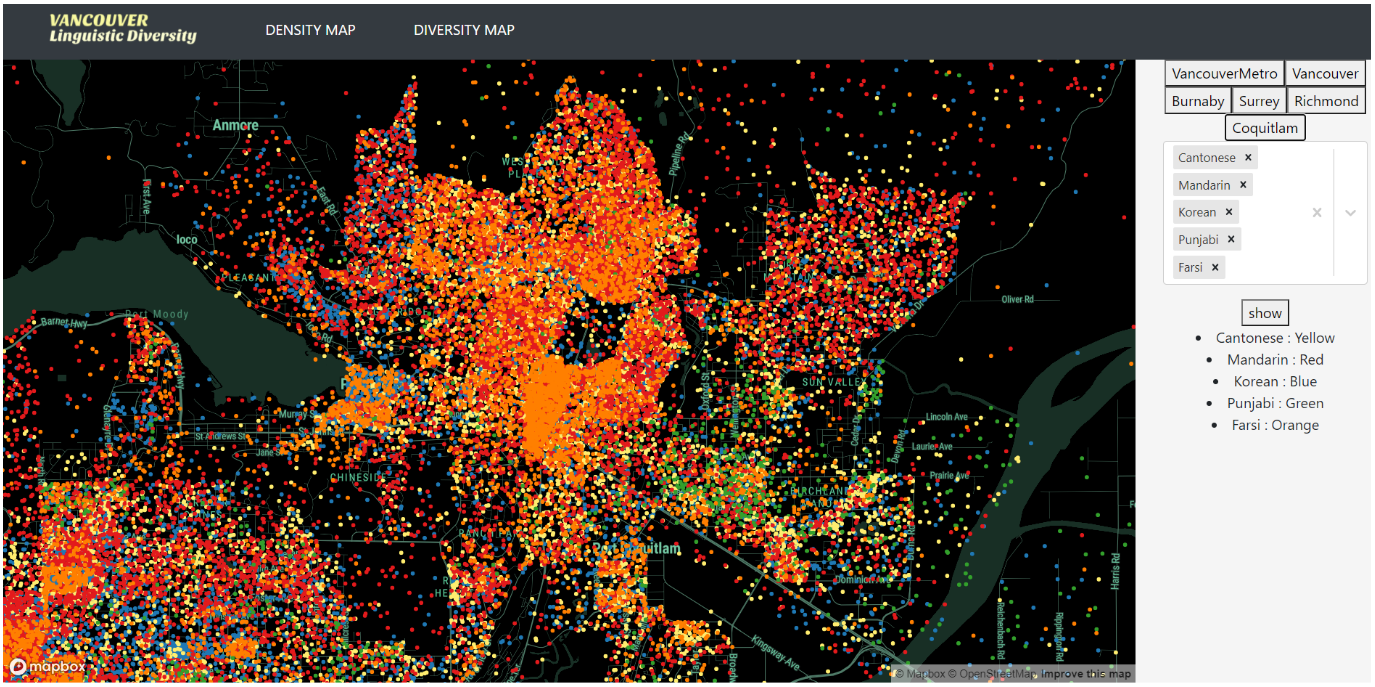
A user wants to see how speakers of the Korean language are distributed in Metro Vancouver, so, in the density map view, she selects it using the selection widget in Figure 4 and then press 'show'. Now she can see how speakers of the Korean language are distributed in Metro Vancouver in the dot density map. This also allows her to locate clusters and identify regions which have more Korean language speakers than others.

Now she is wondering how the distribution of Korean speakers compares to that of German and Spanish language speakers, so she adds German and Spanish to the selection widget. Now she sees the distribution of all three selected languages in the dot density map and can also get a sense the numbers of speakers of the languages by comparing the numbers of dots corresponding to these three languages. If she is interested in the distribution in a particular neighbourhood, she can zoom in to that neighbourhood by clicking on the button on the top of the widget that corresponds to that neighbourhood.

6.2 Scenario 2

A user is interested in seeing what languages are the biggest contributors to the total number of speakers of each language family in Richmond. Hence he selects the neighbourhood labelled as 'Richmond' on the diversity map by clicking on it. Now he can see the relative number of

[3] `https://deck.gl/docs/api-reference/layers/scatterplot-layer` and `https://deck.gl/examples/scatterplot-layer/`



- Cantonese : Yellow
- Mandarin : Red
- Korean : Blue
- Punjabi : Green
- Farsi : Orange

Fig. 5. Density map view when zoomed in on Coquitlam.

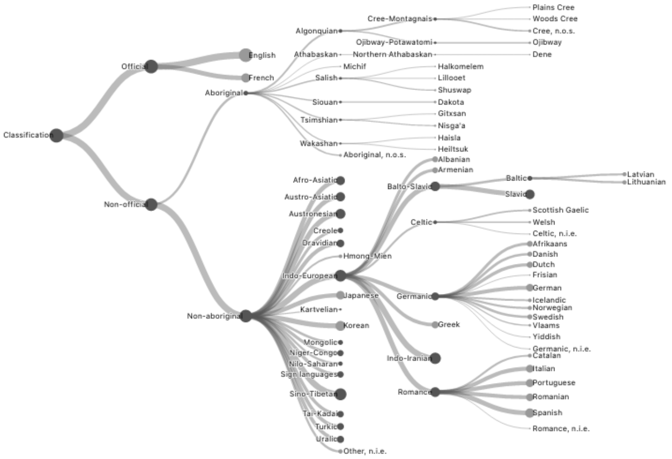


Fig. 6. Collapsible tree of language families. The size of nodes and links reflects the log-transformed number of speakers. The saturation of nodes indicates whether a node is a leaf.

speakers in various language families in terms of the size of the nodes and links in the language family tree. If he wants to know the precise numbers of speakers, he can just hover over any of the nodes. He can also expand or collapse any non-terminal nodes as he sees fit.

7 Milestone

We plan to spend about 217 hours together towards the project. Table 3 provides a rough estimate of the project's tasks, the ownership of the tasks and the actual numbers of hours spent on each task.

8 Discussion and Future Work

This section is structured as follows: we first provide a critique of the visual idioms in this project, focusing on the limitations of these idioms, and then we list some action items to be carried out had we have more time.

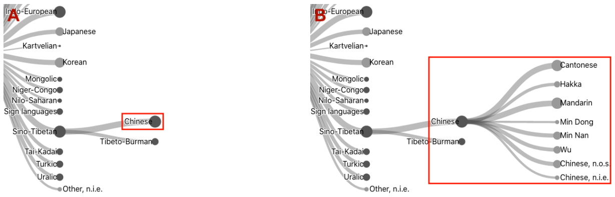


Fig. 7. Illustration of the collapse-expand interaction of non-terminal nodes. (A) The 'Chinese' node is collapsed. (B) The 'Chinese' node is expanded.

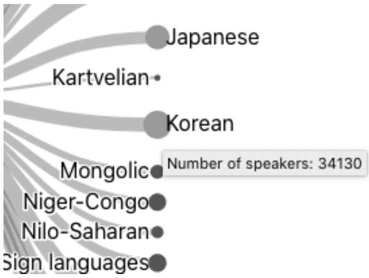


Fig. 8. Illustration of the hover interaction of the 'Korean' node.

8.1 Limitations

We address the limitations from two aspects: (i) those inheriting in the visual idioms themselves, and (ii) those induced by memory and computational cost due to the nature of our dataset.

8.1.1 The Diversity Map

One major limitation of choropleth maps is that they tend to overemphasize large regions. One way to mitigate this problem is to use a smoothed contour map, although smoothing might lead to loss of detail.

8.1.2 The Density Map

Visually, one major problem with dot density maps occurs when there are plenty of dots concentrating in one region, such that they overlap to a large extent. When this serious occlusion happens, underlying clusters can be obscured. One approach to tackle this problem is to dynamically adjust the number of tokens each dot represents, based on the zoom level of the map, so that underlying clusters can still be visible without sacrificing dot density. However, this strategy will not work well if there are categories that only have extremely few tokens—this is precisely what happens in our case, as a lot of DAs only have a handful of speakers of aboriginal languages.

Another limitation concerns the number of categories that can be shown on the map at the same time, as the categories are distinguished on the basis of colour hues. We quickly run out of colours when we want to compare more than a dozen categories, which renders showing all 214 languages simultaneously out of the question. The way we deal with this 'colourful' problem is to constrain the number of categories the user can compare, which arguably is not very elegant.

Computationally, dot density maps also present challenges when there are a huge number of dots to plot. In our dataset, if we represent each speaker or signer as a dot, the number easily exceeds two millions. SVG elements are out of the window under this circumstance, and using the canvas element plus zooming or panning can quickly crash the browser. This is why we resort to `deck.gl`, which is designed to handle large-scale datasets.

8.1.3 The Language Family Tree

The origin of problems in our language family tree traces back to a large number of languages and the skewed distribution of speakers of these languages.

First, while hierarchical aggregation solves the problem of displaying all categories, it also means that it is hard to compare leaves that are situated in branches far away from each other. This type of comparison will increase cognitive load of the side of the user, as they need to memorize information while jumping between different parts of the tree. One way to solve this problem is to allow the user to dynamically change the relative locations of nodes, such that nodes to be compared can be in close proximity.

Second, even with many languages claiming grounds in Vancouver, English is the dominant tongue, with its speakers outnumbering all the other languages in almost all neighbourhoods. Also, in many neighbourhoods, two or three languages outweigh all the other languages combined. As pointed out in Section 4, a linear scale mapping the number of speakers to line width or node size will make a few nodes or links dominate the tree. Log-transforming values solves this dominance problem, at the cost that this significant imbalance in speaker distributions is diminished visually. In our earlier attempts, we also try to use a zoomable treemap to encode the distribution of speakers across different languages. However, the same problem persists with the treemap—many languages occupied areas so small as to render them invisible in practice. Treemaps are also not as visually pleasant as tree diagrams, so we end up opting for a collapsible tree.

8.2 Prospect

With sufficient supply of time, this visualization can benefit from several add-ons, aside from those fixes mentioned in the above sections:

- In our original proposal, the language family tree serves a dual function: to display languages and to input selected languages. A checkbox can be attached to each node, so that the user can check-select languages or language groups for comparison in the dot density map. In this case, the colour channel of the collapsible tree can also be used.
- The neighbourhood selection function on the control panel of the dot density map can be replaced by direct selection from the dot density map.
- Allowing for both selections of neighbourhoods and DAs.
- Unifying the styles of the choropleth and the dot density maps.

9 Conclusion

Visualization is full of compromises, but these compromises are necessary to get the core messages across. In our visualization, we use a choropleth and a dot density map to show the geographic distributions of linguistic diversity and individual languages respectively. A collapsible tree is employed to provide information on the number of language users. Several interactions are supported to make our picture worth more than a thousand words.

Acknowledgments

We would like to thank Tamara Munzner for her feedback on different stages of this project. Additional thanks also go to Claude Demers-Belanger and Sanyogita Manu for their inputs from peer review.

References

[1] M. Bostock. Poisson-disc [https://bl.ocks.org/mbostock/19168c663618b7f07158], December 2017.

[2] M. Bostock. Collapsible tree [https://observablehq.com/@d3/collapsible-tree], October 2018.

[3] M. Bostock. Zoom to bounding box [https://observablehq.com/@d3/zoom-to-bounding-box], August 2019.

[4] M. Bostock, V. Ogievetsky, and J. Heer. D^3 data-driven documents. *IEEE Transactions on Visualization and Computer Graphics*, 17(12):2301–2309, 2011.

[5] J. H. Greenberg. The measurement of linguistic diversity. *Language*, 32(1):109–115, 1956.

[6] E. L. Koua. Using self-organizing maps for information visualization and knowledge discovery in complex geospatial datasets. In *Proc. the 21st International Cartographic Conference (ICC)*, pp. 1694–1702, 2003.

[7] C. R. Luebbering, K. N. Kolivras, and S. P. Prisley. Visualizing linguistic diversity through cartography and GIS. *The Professional Geographer*, 65(4):580–593, 2013. doi: 10.1080/00330124.2013.825517

[8] G. McNew, C. Derungs, and S. Moran. Towards faithfully visualizing global linguistic diversity. In *Proc. LREC 2018*, pp. 805–809, 2018.

[9] T. Munzner. *Visualization analysis & Design*. CRC Press, 2014.

[10] P. Shanbhag, P. Rheingans, and M. desJardins. Temporal visualization of planning polygons for efficient partitioning of geo-spatial data. In *IEEE Symposium on Information Visualization*, pp. 211–218, 2005.

[11] D. Shkolnik. Language diversity in Canada [https://www.dshkol.com/2017/language-diversity-in-canada/], October 2017.

[12] A. Skupin and R. Hagelman. Attribute space visualization of demographic change. In *Proc. the 11th ACM international symposium on Advances in geographic information systems (GIS)*, pp. 56–62, 2003.

[13] UNESCO. *Investing in cultural diversity and intercultural dialogue: UNESCO world report*. UNESCO, 2009.

[14] J. von Bergmann, D. Shkolnik, and A. Jacobs. *cancensus: R package to access, retrieve, and work with Canadian Census data and geography*, 2020. R package version 0.3.2.

[15] K. E. Walker. Scaling the interactive dot map. *Cartographica*, 53(3):171–184, 2018.

[16] Y. Wang. Deck.gl: Large-scale web-based visual analytics made easy, 2019.

Table 3. Project timeline and task breakdown.

Task	Est. hours	Act. hours	Deadline	Description	Assignee
Proposal writeup	5	5	23 Oct.	- Brainstorm ideas	All
Update writeup	10	8	17 Nov.	- Incorporate feedback from the proposal writeup	All
Final writeup	25	21	14 Dec.	- Finalize paper	All
Peer project review	5	2	19 Nov.	- Paper reading and commenting; - presentation	All
Final presentation	5	4	10 Dec.	- Slide preparation; - presentation	All
Pre-proposal meeting	1	1	13 Oct.	- Meeting note preparation	All
Post-update meeting	1	1	24 Nov.	- Meeting note preparation	All
Literature review	20	20	1 Nov.	- Browse/read relevant papers	All
Tool familiarization	20	27	8 Nov.	- Parallel learning (`D3`, `React`, `Bootstrap`) during implementation	All
Dataset preprocessing	10	45	1 Nov.	- Extract census data	Roger
				- Calculate LDI	Roger
				- Python implementation for random distribution of point coordinates for each DA, relative to each language for dot density map	Namratha
				- Construct family tree dataset	Roger
				- Construct family tree dataset with language counts for treemap	Namratha
				- Generate TopoJSON file from GeoJSON	Anika
				- Format CSV files containing language information and data description to map language codes to number of speakers in each DA identified by GeoID	Anika.
Implementation					
- Interface	10	7	18 Nov.	- Set up diversity map view	Roger
				- Set up density map view	Anika
- Choropleth map	10	15	25 Nov.	- Core map with SVG elements	Roger
				- Enable zooming and panning	Roger
				- Peripherals and legend	Roger
				- Connect map with collapsible tree	Roger
- Dot density map	45	95	25 Nov.	- Dot density map using SVG element and `D3` [Very poor performance and hence not used in final implementation]	Anika
				- Dot density map using canvas element, `D3`, and Poisson-disc sampling [1] to equally space the generated dots [Better performance than previous implementation, but was still taking considerable time to load and hence was not used in final implementation]	Anika
				- Search for alternative tools for achieving better performance	Anika
				- Dot density map using `react-map-gl` as base map and `deck.gl` as data layer	Anika
				- Design a custom map style for the base map using `Mapbox Studio`	Anika
				- Enable language selection interaction	Anika
				- Select a neighbourhood to zoom in	Anika
				- Further reduce loading time by controlling opacity and using React Hooks to manage states	Anika
				- Show and update map legends based on current selection of languages	Anika
- Collapsible tree	30	20	25 Nov.	- Basic layout with `d3-hierarchy`	Roger
				- Adjust node size and link width	Roger
				- Enable collapse and expand interaction	Roger
				- Update tree in response to neighbourhood selection	Roger
- Treemap	20	25	6 Dec.	[Not adopted after careful consideration]	
				- Map layout to visualize proportion of language users in a language group	Namratha
				- Explore zoomable treemaps	Namratha
				- Enable zoom-in and zoom-out on language selection	Namratha
Total	217	296			