Data Visualization Milestone2

Feng Yiyang 352042, Zhou Naisong 353331, Tang Xuehan 353567
School of Computer and Communication Sciences, EPFL, Switzerland

I. Project Goal

Our data visualization project aims to present an engaging and intuitive view of the programming languages landscape, highlighting their geographical distributions, relevant terminology, and interconnections. Driven by a desire to enrich understanding and appreciation for the field's progression, we aim to engage programming language developers, software engineers, computer science students, educators, and technology enthusiasts in exploring the world of programming languages and their diverse impact. To effectively convey these insights, we have devised four primary visualization components:

- Programming Languages Cloud: A dynamic programming languages cloud that showcases a random batch of programming languages.
- Geographical distribution: A map highlighting the global distribution of programming languages.
- Text representation: A graphical representation of the frequency and prominence of terms associated with different programming languages.
- Network visualization: A diagram illustrating the connections and associations between various programming languages.

II. Sketches of the Visualization

Our webpage could be found here.

A. Dynamic Programming Languages Cloud

Figure 1 presents a random batch of programming languages. By hovering over a programming language, users can view details such as its debut year and rank. Clicking on a programming language directs users to the corresponding language's information in the table.

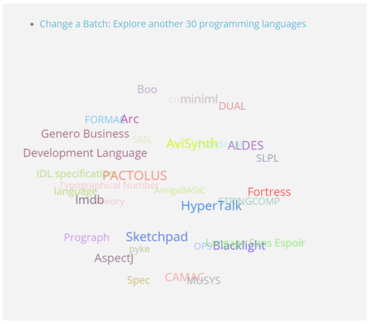

Figure 1. Dynamic Programming Languages Cloud

B. Geographical distribution

Figure 2 displays the distribution of developers for various programming languages across different countries. By hovering over a country, users can view details such as the top 3 most popular programming languages within that country. Clicking on a country will pin the details list for easy reference. Additionally, each programming language in the details list can be clicked, allowing users to navigate directly to the corresponding language's information in the table.

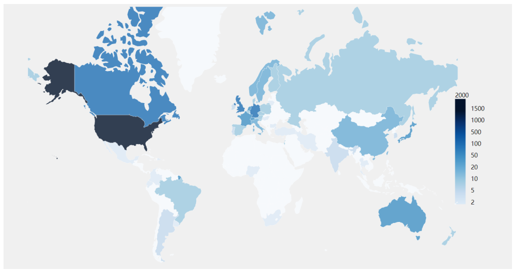

Figure 2. Distribution of Developers for Various Programming Languages Across Different Countries

C. Text Visualization

Figure 3 illustrates the text visualization section of the website. Users can use the slider to select the year, which will filter programming languages that appeared before that year in the selection box. Users can view the LDA visualization for the chosen language by selecting it from the selection box.

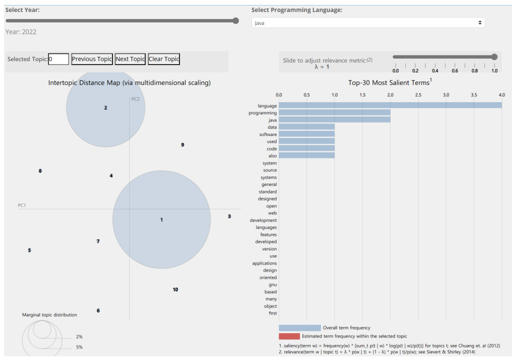
Select Year:
Year: 2022
Select Programming Language:
Java
Selected Topic
Previous Topic
Next Topic
Clear Topic
Slide to adjust relevance metric
Intertopic Distance Map (via multidimensional scaling)
Top-30 Most Salient Terms
Marginal topic distribution
Overall term frequency
Estimated term frequency within the selected topic

Figure 3. Text visualization sketch with the LDA method.

D. Language Network

Figure 4 illustrates the language network section of the website. Users can hover over each node to look into the details of this language and drag each node item. Languages of the same type are visualized with the same color, and the legend is to the right of the graph.

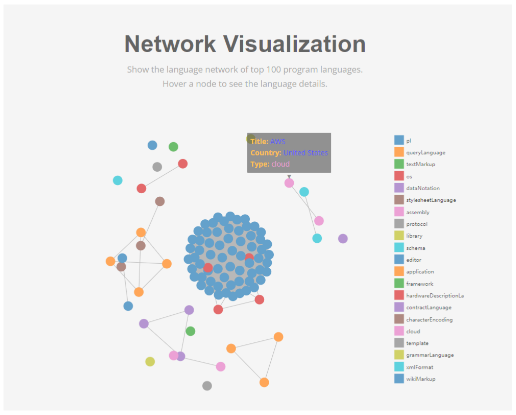

Figure 4. Network visualization for top 100 languages

III. Tools and Lectures

We plan to use the Bootstrap framework for the website and the D3.js [1] library for the visualizations. We will also use the JavaScript programming language and modify HTML and CSS files.

A. *Dynamic Programming Languages Cloud*

We employ the TagCloud [2] GitHub repository to generate a dynamic 3D text cloud for programming languages. Mouseover and click events are managed through dedicated JavaScript functions.

B. *Geographical distribution*

For geographical distribution, we utilize TopoJSON for rendering the countries' topologies. We then employ D3.js, a JavaScript-based library, to color the map and facilitate interactivity. Lecture 8.1 and Lecture 8.2 offer valuable guidance on visualizing data on maps, making them instrumental resources for this portion of the project.

C. *Text Visualization*

We will use a Python script and the pyLDAvis library [3] to save the LDA representations for each programming language in JSON format for the LDA visualisation. Then, we will use JavaScript to load and render the visualizations. Relevant lectures include those on text visualization. We refer to the lecture and switch the way of text visualization from word cloud to LDA visualization, as word cloud has bad visual encoding (size vs position) with no structure. Besides, word cloud encoding is inaccurate for long words that are bigger.

D. *Language Network*

For the network representation, we also use a python script to analyze and generate the inner relations in languages. Then, we use JavaScript to load and render the visualizations based on D3.js. Relevant lectures include those related to network visualizations.

IV. Core Visualization and Extra Ideas

A. *Core Visualization (Minimal Viable Product)*

Dynamic Programming Languages Cloud Visualization

- Create a dynamic 3D text cloud displaying a random batch of programming languages.
- Allow users to change the displayed programming languages to a new batch.
- Implement a tooltip that appears when hovering over each text element, showing detailed information. When a programming language is clicked, guide users to the corresponding detailed information in the table.

Geographical Distribution Visualization

- Develop an interactive global map.
- Implement a tooltip that appears when hovering over each country, displaying detailed information. When a programming language within the tooltip is clicked, direct users to the corresponding detailed information in the table.

Text Visualization

- Implement a time slider that filters programming languages based on the selected year.
- Create a selection box that displays programming languages appearing before the selected year.
- Load and render the LDA visualization for each programming language using the pyLDAvis library.

Network Visualization

- Load the network data for the whole top 100 languages, enable dragging nodes and zooming.
- Implement a tooltip when hovering over each node to see the detailed information.

B. *Extra Ideas*

General Part

- We can paginate the table to reduce loading overhead and the length of the page.
- We can enhance the appearance of the table displaying the details.
- We can add more styles to the words in the page.
- For the overall style, we can tailor it to suit our topic: Programming Languages.

Dynamic Programming Languages Cloud Visualization

- We can replace the text with programming language logos. To accomplish this, we may need to utilize a Python web crawler to download the logos.
- We can incorporate additional buttons, allowing users to pause or resume the animation of the text elements.

Geographical Distribution Visualization

- We can enhance our visualization by transforming it into a 3D spherical representation of Earth, offering a more dynamic and engaging visual experience.
- In cases where adjacent countries have not developed any programming languages, it may be beneficial to add borders to distinguish between them more effectively. Otherwise, it might be difficult to differentiate one country from the other in the visualization.

Text Visualization

- Integrate the world map and the network information to display programming languages based on the selected year, and allow users to select programming languages by clicking on them.
- Add tooltips or other interactive elements to the LDA visualization to provide more context or information about the topics and terms.
- Find ways to accelerate webpage response speed.

Network Visualization

- Implement double-click response of nodes: to zoom in and rearrange the network graph to center to specific nodes.
- Enrich the node relationships, add features other than type as parameters to define edge existence and strength.
- Accelerate webpage load and response speed.

V. Functional Project Prototype Review

We have developed an initial website running with the basic skeleton of the visualization/widgets. The prototype includes a functional time slider, programming language selection box, and LDA visualization, which can be filtered based on the selected year. The prototype demonstrates our core visualization components, and we plan to continue refining and expanding our project by incorporating the extra ideas mentioned above.

References

[1] M. Bostock, "D3.js - data-driven documents," 2011. [Online]. Available: https://d3js.org/

[2] C. Min, H. Wen, F. Mayer, and H. Ahrens, "3d tagcloud," https://github.com/cong-min/TagCloud, 2022.

[3] B. Mabey and R. Allen, "pyldavis: Interactive topic model visualization. port of the r ldavis package," 2014, python library. [Online]. Available: https://github.com/bmabey/pyLDAvis