Read the Bible in Context with BibleViz

Haeji Jung haejij@cs.ubc.ca
Yuri Kim yurikim1@cs.ubc.ca

I. Introduction

MANY Bible readers today encounter Scripture through digital apps or daily verse feeds that emphasize short, isolated excerpts. While these fragments are easily accessible and shareable, they often obscure the broader narrative, historical, and theological contexts that give each passage its meaning. Reading in context, however, is essential to understanding the Bible as an intentional and coherent text rather than a collection of moral sayings or inspirational quotes.

BibleViz aims to support readers who wish to engage with Scripture more deeply by providing an interactive visualization of contextual relationships within the Bible. The tool enables exploration at both verse and word levels, helping users discover semantically related verses, trace thematic distributions across books, and examine linguistic associations among words. Through this multi-layered interface, users can move fluidly between coarse-grained (verse-level) and fine-grained (word-level) views, fostering reflection on how individual passages relate to the larger biblical narrative.

Technically, the system combines natural language processing (NLP) and interactive visualization. We utilize semantic similarity modeling to map relationships between verses, and embedding-based representations to position words according to contextual proximity. The resulting explorer allows users to query, compare, and interpret verses interactively—promoting an experience of reading that is both analytical and contemplative.

Beyond traditional concordances or curated thematic analyses—which rely heavily on human annotation and interpretive decisions—our approach introduces a computational perspective on biblical context. By leveraging language models trained on Scripture itself, BibleViz uncovers patterns of semantic affinity that may not be immediately visible through manual study. While such representations are not directly explainable and may lack explicit theological rationale, they offer a complementary lens: one that is empirical, scalable, and capable of revealing emergent relationships across the text. This data-driven approach does not replace traditional exegesis, but rather augments it—inviting readers and scholars alike to explore how algorithmic embeddings might reframe familiar passages and generate new questions about meaning and connection within the Bible.

Personal Expertise. The project team consists of Haeji Jung and Yuri Kim. Haeji is a first-year PhD student specializing in NLP, with prior experience in language model training, tokenization, and other text analysis techniques. Haeji will take the lead on the more advanced NLP-related components in the analytical work. Yuri has prior experience as a data analyst and is proficient in data processing and analysis using Python. Both members have little to no experience in front-end development. While Haeji, as a Christian, is familiar with the Bible, Yuri is not, which may offer complementary perspectives in interpreting and presenting the data.

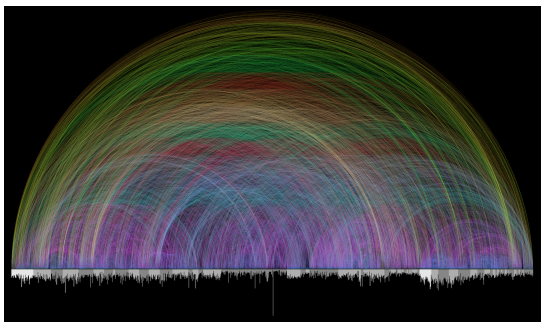

Fig. 1: *Bible Cross-References Visualization* by Chris Harrison.

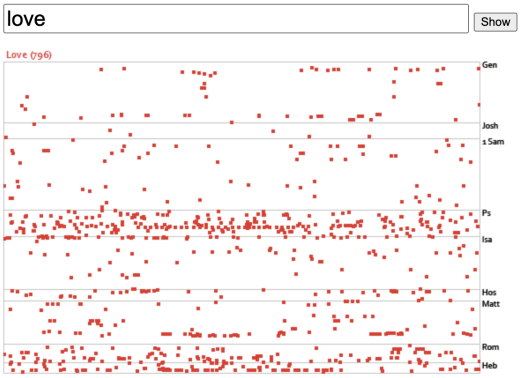

Fig. 2: *Bible Word Locator* from *OpenBible.info*.

II. Related Work

A. Bible-Focused Visualization Tools

Early efforts to visualize relationships within the Bible have focused primarily on thematic or structural connections rather than on semantic context. *Bible Cross-References* by Chris Harrison (1) (see Fig. 1) presents a well-known visualization of cross-referenced verses using arc diagrams, effectively illustrating the interconnected nature of scripture but without interactive or semantic exploration. Other tools such as the *Bible Word Locator* (see Fig. 2) and *Applying Sentiment Analysis on the Bible* from *OpenBible.info* (2; 3) provide keyword-based and sentiment-based analyses, respectively, but they remain limited to static or search-driven exploration

rather than supporting contextual understanding. Similarly, *Viz.Bible* (4) offers genealogical visualizations such as bloodlines, focusing on narrative lineage rather than linguistic or semantic relationships between verses.

B. Visualization for Reading and Interpretation of Texts

Beyond Bible-specific applications, a growing body of work has explored how textual data—ranging from individual words to full corpora—can be visualized to support reading and interpretation. In digital humanities, Benito et al. (2017)(5) modeled dictionary entries as interconnected nodes in a semantic network, allowing users to explore complex relationships among linguistic units. Similarly, Campbell et al. (2018)(6) supported multi-granular textual analysis by visualizing named entity networks in a large corpus of women's writing. These works exemplify how graph-based and coordinated views can be used to support both close reading and distant reading. Other approaches incorporate sequential or structural cues in textual visualization. For instance, *TextArc* (7) and the *Word Tree* (8) visualize words and phrases while preserving aspects of reading order or syntactic flow. While such systems focus on different facets of text—frequency, sequence, or repetition—they collectively illustrate how visualization can surface latent patterns and structures in large bodies of text. Our work builds on these traditions by modeling semantic similarity among verses and words using embedding-based representations, and mapping those relationships into interactive graph and spatial views tailored to the biblical corpus.

C. NLP and Embedding Visualization

Recent advances in visualizing high-dimensional embeddings have provided powerful ways to explore latent semantic relationships in text. Tools and techniques such as t-SNE (9) have been widely used to map high-dimensional linguistic representations into interpretable two-dimensional spaces. Interactive embedding explorers (e.g., Grant Custer's Embedding Explorer (10)) also demonstrate how users can intuitively navigate semantic structures through visual layouts based on similarity. These techniques inspire our word-level view, which visualizes contextual word embeddings derived from Bible text using t-SNE for dimensionality reduction.

III. Data and Task Abstractions

A. Data

1) Bible Data: We use the text of the **Bible**, which has a multiscale hierarchical structure from *books* to *words*. Additionally, each book of the Bible can be classified into one of the ten themes: The Pentateuch (Law), History, Wisdom, Major Prophets, and Minor Prophets, which belong to the Old Testament, Gospels, Acts, Pauline Epistles, General Epistles, and Prophecy, which belong to the New Testament.[1] We incorporate theme attribute into our visualization, resulting in one additional layer above the *books* level. The expanded hierarchical structure is therefore: *themes – books – chapters – verses – words*. Specifically, the Bible comprises a total of 66 *books*. Each book contains, on average, approximately 18 *chapters*, and each chapter consists of multiple *verses*. There are 31,298 verses in total, with each verse containing an average of 25 *words*. Our project focus on two hierarchical levels: verse-level and the word-level data. We therefore construct two datasets—one that treats each *verse* as an item, and another that treats each *verse*×*word* pair as an item. This enables users to trace occurrences of the same word across different verses and supports fine-grained, interactive exploration of the corpus.

2) Neural Language Model: We employ a neural language model trained on the Bible text to obtain latent representations of each textual unit, following the architecture of (12). The text is first tokenized into a predefined set of tokens, each mapped to a *token embedding*—a vector representation used as input to the model[2]. The model then takes these embeddings as input and produces a contextualized vector for each token, which we refer to as a *verse*×*token embedding*. Unlike the token embeddings, these vectors capture the contextual information of surrounding words within the same verse. As a result, the same word can have different representations across different verses. A *verse embedding* is then obtained by averaging all verse×token embeddings within a verse.

3) Data Abstraction: Table I summarize the attributes and their types along with which dataset they are included in. As described in Section III-A1, the Bible has a multiscale hierarchical structure, and we construct two datasets that focus on verses and verse×token pairs, respectively. We refer to each as *verse dataset* and *token dataset* henceforth, for simplicity. Theme, Book, Chapter, Verse, and Verse ID, are included in both datasets as an attribute for each item. While we primarily use these attributes as categorical, Book, Chapter, and Verse are defined based on the order that they appear in the Bible, inherently possessing an ordinal characteristic. Verse ID acts as a shared key across our datasets, enabling us to interconnect the different views that are to be presented in our visualization. Verse/Token/Verse×Token Positions are derived by applying dimensionality reduction to vectors extracted from the language model we trained. We obtained high-dimensional (768-dimension) token vectors from the model, and applied UMAP to yield a 2-dimensional coordinate set. This 2D projection provides the coordinates for visualizing the latent space.

B. Task

Our tool is designed to help readers engage with the Bible in context. Specifically, it targets users who seek to deeply reflect on and interpret individual verses. To support this, the tool allows users to query and browse verses or explore words of interest.

[1]This has been a consensus for themes of the Bible. Another general way to classify the books are by seven: Law, History, Wisdom, Prophecy, Gospel, Letter, and Apocalypse (11). We start with ten categories.

[2]Since the basic unit representing a word—typically separated by spaces—is a token, we use the terms word and token interchangeably throughout the project. The term token, however, more precisely denotes the unit used for computational analysis.

Attribute	Dataset	Description	Type	Cardinality/Range
Theme	Both	Thematic classification of a biblical book (e.g., History, Wisdom, Gospels).	Categorical	10
Book	Both	Name of the book of the Bible (e.g., Genesis).	Categorical (O)	66
Chapter	Both	Chapter number of the item (e.g., 1, 16).	Ordinal	[1, 150]
Verse	Both	Verse number of the item (e.g., 1, 16).	Ordinal	[1, 176]
Verse Text	Both	Textual content of a numbered subdivision within a chapter.	Categorical (Textual)	31,298
Verse ID	Both	Unique identifier assigned to each verse in the Bible, defined by the combination of book, chapter, and verse number (e.g., GEN_1_1).	Categorical	31,298
Verse Position	Verse	Dimensionally-reduced version of the language model output that represents each verse.	Quantitative	2-dim vector
Verse Similarity	Verse	Similarity scores computed with each other verse.	Quantitative	[0,1]
Tokens	Verse	List of tokens (e.g., words or subwords) of the verse.	Categorical	10,000
Token IDs	Verse	List of Token IDs of each token in the verse (e.g., [10, 329]).	Categorical	10,000
Token	Token	Basic unit of text (e.g., words or subwords) used for computational analysis.	Categorical (Textual)	10,000
Token ID	Token	Unique identifier assigned to each token (e.g., 823).	Categorical	10,000
Token Position	Token	Dimensionally-reduced version of the word embedding that represents each unique token.	Quantitative	2-dim vector
Verse×Token ID	Token	Unique identifier assigned to each verse×token item.	Categorical	389,323
Verse×Token Position	Token	Dimensionally-reduced version of the language model output that represents each token within a verse.	Quantitative	2-dim vector
Verse×Token Similarity	Token	Similarity scores computed with each other verse×token.	Quantitative	[0,1]

TABLE I: Data abstraction for the Bible visualization. There are attributes that are shared across different datasets. Dataset column specifies the dataset that contains the attribute. (O) in the Type column indicates that the attribute possesses an ordinal characteristic.

1) Search: BibleViz invites users to search for either verses or words of interest. A verse can be searched by its *Verse ID*, which consists of the book, chapter number, and verse number. Once a target verse is selected, the tool displays similar verses from other parts of the Bible, along with their associated information, as described in Section III-B2.

A word can be searched with its text, prompting the system to suggest similar tokens from the vocabulary. For a word, users can explore its various occurrences and usages across the Bible, as detailed in Section III-B3.

2) Browse: For the verse-level task, an item is a verse. Users can browse verses that are semantically similar to a queried verse[3]. This aims to enable users to identify similar verses, and understand various aspects of the target verse by analyzing similar verses. Specifically, users can:

- *Select* the verse of interest (target verse) to display similar verses—verses with similarity scores above a certain threshold. This enables users to read individual relevant verses and enhance understanding.
- *Analyze* the book themes of similar verses. This helps users see which themes the similar verses are concentrated in, providing contextual clues for interpreting the verse and understanding broader patterns in the Bible.
- *Compare* the number of similar verses for each verse. This naturally invites users to identify verses with many connections and encourages further exploration of those that are highly connected.

3) Explore: For the word-level task, an item is a word in a certain verse. Users can select a target word to explore how its meaning varies depending on the verse context. The visualized word instances represent Verse×Token Embeddings obtained through a dimensionality reduction method. By viewing all instances of the word across the Bible rather than in isolation, users are encouraged to reflect on differences in usage, deepening their understanding of the text. Specifically, users can:

- *Select* a word of interest to visualize its occurrences across verses. This shows how each instance is positioned relative to all words in the Bible, and provides users the overview of the word usage in the Bible.
- *Identify* atypical usages of a word by comparing the relative positions of its instances. By looking into the instances farther apart, users can better understand the different usages of the same word in the Bible.
- *Search* for words that appear in similar positions to a chosen instance of the target word. This reveals words that share similar contextual or semantic patterns across the Bible.

[3]Semantic similarities across verses is computed based on Vector Embeddings

IV. Solution

Our solution provides two main views with a side panel that includes sub-views and controls.

A. Verse-level View

One of the main views is the verse-level view, which contains a graph of verses and their similarities (See Figure 4). This view employs the verse-level data, and is associated with the task in Section III-B2. The initial view will highlight only those verses that have a sufficient number of highly similar counterparts (i.e., their similarity scores are above a pre-defined threshold.) This is to lower the cognitive load of users by preventing them from initially viewing all 30K dots, and to guide them toward areas of high similarity to begin their exploration.

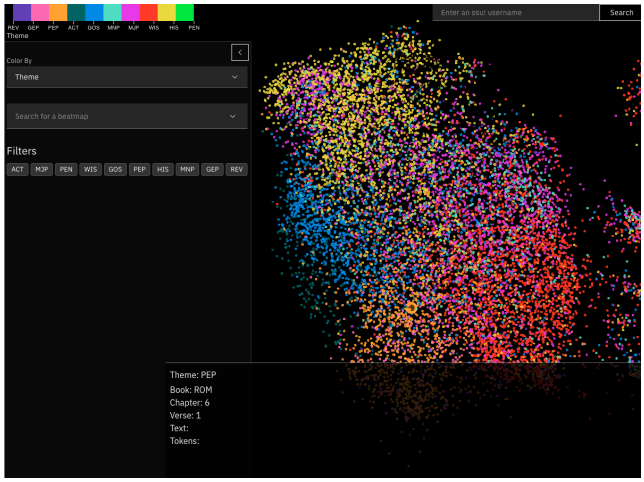

Fig. 3: Screenshot of the progress. While there are not much of our design applied, it shows how the control panel and the meta data for each selected item will be displayed. Current implementation has three color coding modes, and one filtering control based on themes. Our primary next step is to implement filtering functionality based on similarity scores.

1) Visual Encoding: The nodes of the graph represents verses, and the links represent similarities. Nodes are color-coded by book theme and size-coded according to the number of linked similar items. Edges are also size-coded based on their similarity rank to the selected node. Selected nodes and linked nodes are highlighted with full saturation, while selected nodes are identified with bolded edge.

2) Interaction: To select the node, users have to click on the node in the main view. Once selected, the selected item and linked items are highlighted. Also, hovering on the highlighted nodes shows the verse text for each item.

3) Linked Sub-views: The verse-level sub-view displays the selected verses along with their similarity scores. It is designed to accommodate users' intent to compare different verses, assuming that users may be interested in their relationships even when the verses are not directly linked on the graph. Another sub-view below displays the verse text of each selected item. The main view and the two sub-views form linked views, allowing cross-filtering.

4) Control: Users can set a similarity threshold to control how many linked nodes are highlighted. Also, users can toggle node size-coding based on the number of links and adjust the degree to highlight selected and connected nodes.

B. Word-level View

Another main view is the word-level view, which is comprised of two views: one displaying a graph of unique words ("graph view") and another visualizing a latent space derived through a dimensionality reduction method ("DR view") (See Figure 5). This view utilizes word-level data and corresponds to the task described in Section III-B3.

1) Visual Encoding: The nodes of the graph view (left side in Figure 5) represent unique words, and the edges represent similarities between them, which is size-coded based on their similarity rank. Selected items are highlighted as in Section IV-A. On the right side of the word-level view, each mark represents a word in a verse (i.e., verse×token embedding). Only the selected items are highlighted and popped-out, and are color-coded based on their Token IDs.

2) Interaction: To select a node, users can click on it in the graph view. Once selected, the node and its linked items are highlighted, and the corresponding word instances in the DR view are also emphasized. Users can select up to three words at once. Hovering over a highlighted mark in the DR view reveals the full verse corresponding to that node. Notably, users cannot interact directly with the DR view. Due to its high cardinality, each pixel may represent multiple instances rather than a unique one, making direct selection impractical. Instead, interaction is designed to occur through the graph view, where each node occupies more than a single pixel, providing sufficient space for precise user selection.

3) Control: As in the verse-level view, users can set a similarity threshold to control how many linked nodes are shown for the graph view.

C. Connection between Two Main Views

While each view supports a distinct task—browsing verses or exploring word usages—we also incorporate a connection between the two views. Using a shared attribute, the Verse ID, words appearing in the selected verse are displayed within the verse view. When a user clicks on one of these words, the corresponding word is automatically selected in the word-level view.

D. Narrative Exploration via Scrollytelling

To help users unfamiliar with the tool—especially those without prior knowledge in NLP or model-based visualization—we incorporate a narrative, scroll-based walk-through inspired by the concept of *scrollytelling* (13). This design aims to ease the onboarding process, help users build intuition about unfamiliar elements (e.g., similarity, number of neighbors), and inspire curiosity through guided examples.

The narrative interface is structured around a set of illustrative examples derived from our own data analysis using BibleViz. The narrative highlights several compelling findings that emerged from exploring the verse- and word-level views. For instance, we showcase verses that exhibit unexpected semantic similarity despite appearing in distant books, as well as words whose contextual meanings vary across different contexts.At the end of the narrative sequence, users are invited to continue their own exploration through an interactive link to the full visualization tool.

V. Scenario of Use

A user is a Christian, who meditates on Scripture every morning. They take their Bible out, with a laptop to make notes during the meditation. Their routine is to read an assigned chapter from the Bible, highlight verses that resonate with them, and reflect on those verses, often by searching the web or relating them to their own life experiences. After making

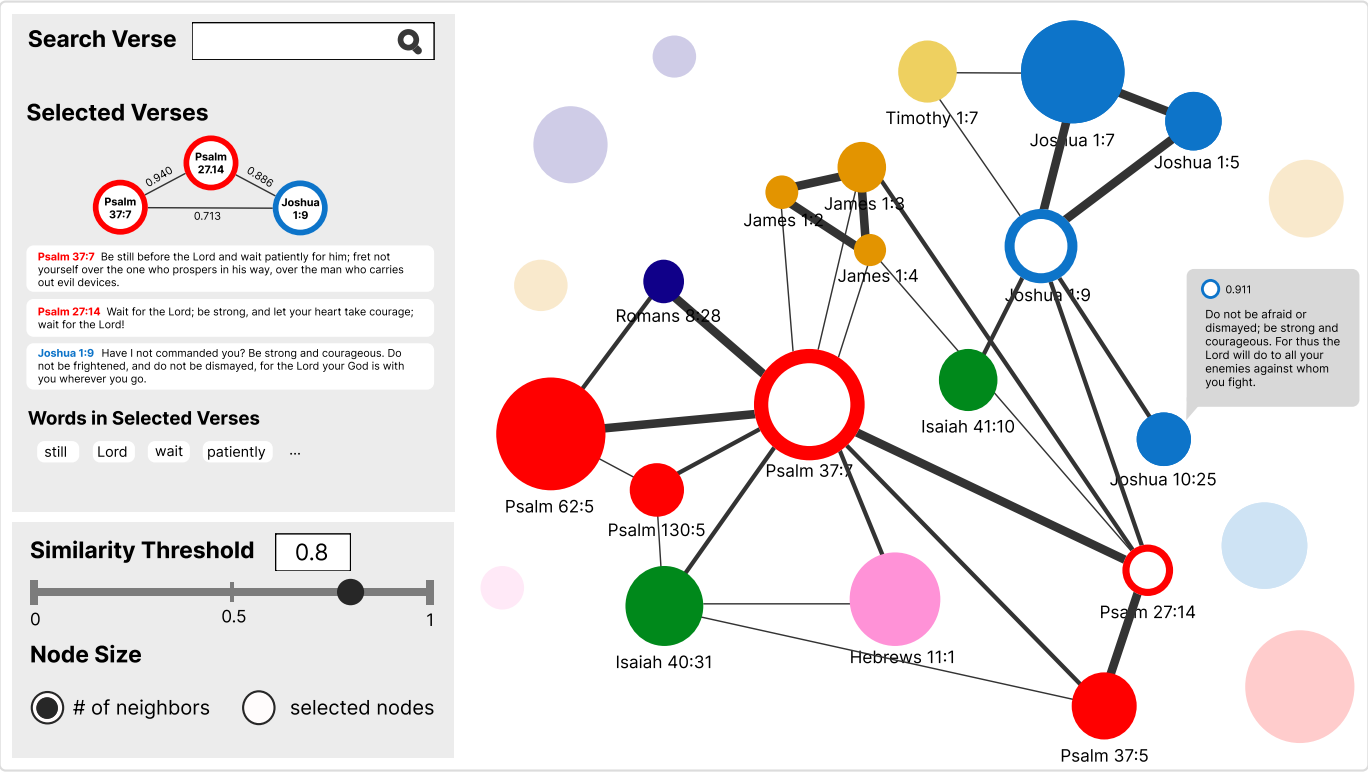

Fig. 4: Verse-level view mockup.

a pass through the chapter, the user revisits the verses they highlighted. One of the verses was particularly meaningful to the user because it closely related to a recent experience. The user became curious whether other verses in the Bible might convey a similar or complementary message.

The user enters the Verse ID, and sets the threshold to 0.9 to retrieve only highly relevant results. Upon clicking the verse, they find that over 10 similar verses highlighted. Also, the user finds that most of the similar verses are from the Gospels, and decide to read one of those books next. They then click the largest linked node to explore more related verses, and turn to the corresponding passage in their Bible. To better understand its context, they also skim through the chapter in which the verse appears. After another cycle of exploring similar verses, the user feels they have gained a deeper understanding of the verse.

The user takes notes on the similar verses they found to reflect on them throughout the day and also records their plan for the next book to read.

VI. Implementation

For implementation of BibleViz, we plan to employ D3 Observable to build the interactive visualization interface, due to its flexibility in rapid prototyping and seamless integration with the web. The visualization tool will be hosted on a publicly accessible website to allow easy exploration by users. Given that the team members have little to no experience with JavaScript, this plan may be revised. For text processing, tokenization, and language model training, we will utilize Python, incorporating libraries and APIs from huggingface[4]. The language model training will be done from scratch using only English Bible corpus available on public, which takes around 4 hours.

VII. Milestones

The project milestones and deadlines are shown in Table II. The total amount of estimated hours per person is roughly 80 hours per each member.

References

[1] C. Harrison, "Bible cross-references visualization," 2008, accessed: 2025-10-19. [Online]. Available: https://www.chrisharrison.net/index.php/Visualizations/BibleViz

[2] OpenBible.info, "Bible cross references tool," 2011, accessed: 2025-10-19. [Online]. Available: https://www.openbible.info/labs/cross-references/

[3] ——, "Applying sentiment analysis to the bible," 2011, accessed: 2025-10-19. [Online]. Available: https://www.openbible.info/blog/2011/10/applying-sentiment-analysis-to-the-bible/

[4] Viz.Bible, "Bible ancestry visualization (bloodline)," 2020, accessed: 2025-10-19. [Online]. Available: https://viz.bible/ancestry/

[5] J. Benito, L. Borràs, M. Calvo, and J. Vivaldi, "Exploring lemma interconnections in historical dictionaries," in *Proceedings of the 2nd International Workshop on Computational Methods for Digital Libraries (CMDL*

[4]https://huggingface.co/

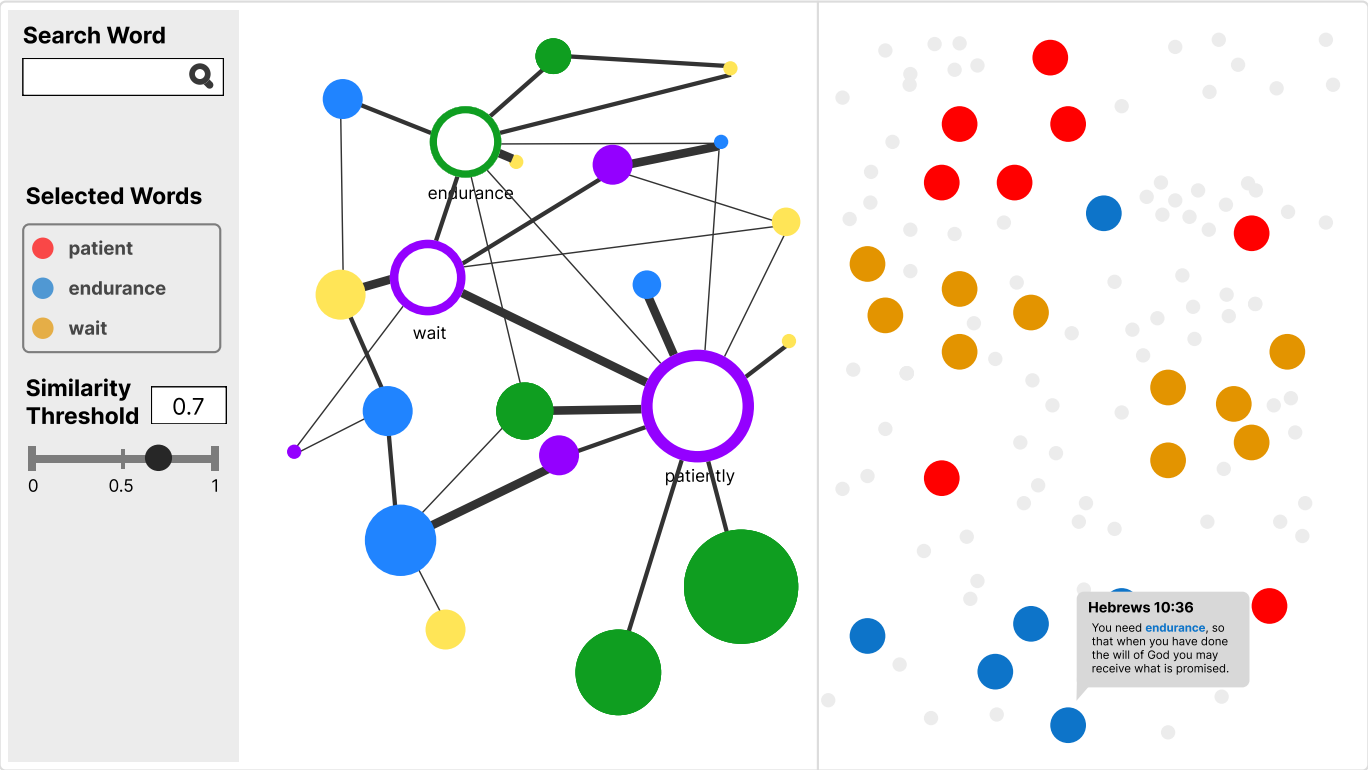

Fig. 5: Word-level view mockup.

2017). CEUR Workshop Proceedings, 2017. [Online]. Available: http://ceur-ws.org/Vol-2038/paper6.pdf

[6] S. Campbell, M. Jennings, and D. Thompson, "Close and distant reading via named entity network visualization," *Digital Humanities Quarterly*, vol. 12, no. 3, 2018. [Online]. Available: http://www.digitalhumanities.org/dhq/vol/12/3/000400/000400.html

[7] W. B. Paley, "Textarc: Showing word frequency and distribution in text," 2002, accessed: 2025-10-19. [Online]. Available: http://www.textarc.org/

[8] M. Wattenberg and F. B. Viégas, "The word tree, an interactive visual concordance," *IEEE Transactions on Visualization and Computer Graphics*, vol. 14, no. 6, pp. 1221–1228, 2008.

[9] L. van der Maaten and G. Hinton, "Visualizing data using t-SNE," *Journal of Machine Learning Research*, vol. 9, pp. 2579–2605, 2008. [Online]. Available: https://www.jmlr.org/papers/volume9/vandermaaten08a/vandermaaten08a.pdf

[10] G. Custer, "Interactive embedding explorer," Online: GitHub Pages, 2022, accessed: 2025-10-19. [Online]. Available: https://grantcuster.github.io/umap-explorer/

[11] J. Edson, "Complete guide to biblical genres [chart]," https://www.biblegateway.com/learn/bible-101/about-the-bible/biblical-genres/, 2025, accessed: 2025-11-12.

[12] Y. Liu, M. Ott, N. Goyal, J. Du, M. Joshi, D. Chen, O. Levy, M. Lewis, L. Zettlemoyer, and V. Stoyanov, "RoBERTa: A robustly optimized BERT pretraining approach," *ArXiv*, vol. abs/1907.11692, 2019. [Online]. Available: https://api.semanticscholar.org/CorpusID:198953378

[13] Christine M., "An introduction to scrollytelling," https://shorthand.com/the-craft/an-introduction-to-scrollytelling/, accessed: 2025-10-19.

Milestone	Task	By	Est.	Act.	Assignee	Description	Progress
Proposal	Set Specific Task	Oct 10	1	1	Haeji	Specify task and the idea	Done
	Research Relevant Viz Examples	Oct 11	1	1	Haeji	Find and analyze existing examples related to our task	Done
	Set Milestones	Oct 13	1	1	Together	Plan project timeline	Done
	Data and Task Abstraction	Oct 15	2	2	Together	Define tasks and organize data attributes by type and scale	Done
	Mock-up Brainstorming	Oct 18	2	2	Together	Sketch ideas and discuss potential visual designs	Done
	Create Scenarios and Mock-up	Oct 19	3	3	Together	Create user scenarios and initial mock-ups	Done
	Proposal Writing	Oct 19	4	4	Together	–	Done
Updates	Literature Review	Oct 22	3	4	Together	Review related research papers and tools	Done
	Tokenizer Exploration	Oct 22	3	3	Together	Search for appropriate tokenizer	Done
	Train Language Model	Oct 20	2	3	Haeji	Train LM on Bible data	Done
	Learn Basics of D3 Observable	Oct 29	10	6+	Together	Familiarize with D3 by making tweaks to existing examples	In Progress
	MVP Implementation	Nov 7	15	10+	Together	Implement basic graph views	In Progress
	Updates Writing	Nov 12	2	4	Together	–	Done
Final Presentation	Apply Feedback from Peer Review and Meeting	Nov 18	2	–	Together	Adjust scope and plans if needed	Not Started
	Implement Interactive Components	Nov 22	6	–	Together	Add user interaction functions to graph views	Not Started
	Explore Examples for Narrative Scrollytelling	Nov 22	3	–	Together	Look for interesting examples to show on Scrollytelling view	Not Started
	Implement Narrative Viz	Nov 25	6	–	Together	Build Scrollytelling view using examples	Not Started
	Fix Bugs, Polishing	Nov 27	4	–	Together	Debug and refine design	Not Started
	Make Slides for Final Presentation	Nov 30	3	–	Together	–	Not Started
	Final Presentation Preparation	Dec 1	2	–	Together	–	Not Started
	Final Report Writing	Dec 11	7	–	Together	–	Not Started

TABLE II: Project milestones. “Est.” = Estimated hours per person, “Act.” = Actual hours per person.