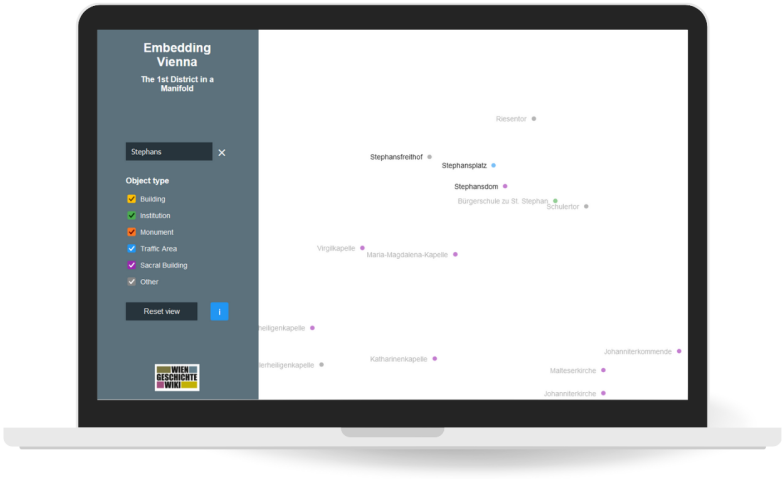
Embedding Vienna

The 1st District in a Manifold

universität wien



Felix Krause

Supervisor: Arndt Niebisch, Privatdoz. MA PhD

July 16, 2024[1]

Abstract

This project presents an innovative interactive visualization tool designed to explore the historical objects of Vienna's first district based on "Wien Geschichte Wiki" data. Leveraging advanced natural language processing and embedding techniques, the tool organizes hundreds of historical objects into a meaningful 2D space, enabling users to intuitively and playfully discover and analyze their relationships. Through an iterative development process, user-friendly features such as zooming, panning and a dynamic search function were implemented, guided by principles of effective visualization design. Despite some limitations, including scope and performance challenges, the tool provides a compelling glimpse into the potential of digital humanities, making historical data more accessible and engaging for a broader audience.

Live demo: f-krause.github.io/wien_geschichte_network/webpage/d3-based/

[1]This report describes the work until 16.07.2024. However, the project might be further developed.

Contents

1 Introduction

In an era where artificial intelligence (AI) is transforming how we engage with data, the intersection of history and technology offers new avenues for exploration and education. This project introduces a novel tool designed to facilitate the discovery and understanding of historical objects within Vienna's first district. The project leverages state-of-the-art sentence transformers to embed articles from the "Wien Geschichte Wiki" (WGW) [Wie24], visualizing them in an intuitive, interactive scatter plot.

The concept of embedding refers to representing textual data in a high-dimensional space where semantically similar items are positioned closer together [Hen23]. This technique, coupled with t-SNE for dimensionality reduction [MH08] and D3.js [Bos11] for dynamic visualization, creates a manifold – a complex, structured representation of Vienna's historical landscape. The goal of this research is to provide an innovative, easy-access and engaging method for exploring and understanding the rich historical tapestry of Vienna's first district.

The approach is grounded in the motivation to make historical information accessible and engaging for a broad audience, including laypeople and students. The primary research question explores how embedding and interactive visualization can enhance the search and discovery process within the WGW. By allowing users to start with a known object and explore related items or by enabling zooming into clusters of interest, this tool promotes a deeper understanding of the first district's historical context.

For example, students might begin their journey with St. Stephen's Cathedral and discover nearby historical sites such as the Virgilkapelle or the Maria-Magdalena-Kapelle, ideally learning about so far unknown objects.

Moreover, this tool represents a significant departure from conventional search methods, where users typically rely on predefined queries and linear exploration ("bottom up"). Instead, it offers a panoramic view of all objects, including museums, churches and coffee houses, enabling a playful and interactive exploration of the city's history ("top down"). Ultimately, this project aims to forward users to WGW pages they might not have otherwise encountered, broadening their knowledge of Vienna's first district.

Overall, this project presents a pioneering approach to historical exploration of Vienna's first district. By embedding and visualizing the rich content of the WGW, this tool transforms the way users interact with historical data, fostering an educational and engaging experience for anyone interested in the historical treasures of Vienna's heart.

This paper documents the project development process. First, it describes the data (sections 2.1–2.2) and data pre-processing (Section 2.3). This is followed by the embedding creation process in Section 2.4, details about visualization in Section 2.5 and the development approach in Section 2.6. Subsequently, the final results are presented in Section 3, highlighting the main features of the tool. Finally, challenges (Section 4) and limitations (Section 5) of this approach are discussed, before concluding the project and presenting future work in Section 6.

2 Methodology

2.1 About Wien Geschichte Wiki

”Wien Geschichte Wiki” (WGW) [Wie24] is a geo-referenced historical knowledge platform established in 2014 by the archive of the city of Vienna[2] and the library in Vienna’s city hall[3]. Its primary objective is to consolidate historical knowledge about Vienna from city administration sources and contributions by experts.

The foundation of the wiki consists of over 27.000 articles from the six-volume ”Historisches Lexikon Wien” (Historical Encyclopedia of Vienna) by Felix Czeike [Cze04]. The wiki encompasses all areas that have historically been part of Vienna’s city limits. As a historical platform, it includes only deceased individuals, unless living persons have demonstrably long-term significance for Vienna. Similar criteria apply to construction projects, urban planning and other entries, excluding current political topics.

Every text and image uploaded is reviewed by a team of city history experts to ensure adherence to scientific standards. All claims must be verifiable through sources and literature, and entries must be written in an objective and professional manner.

2.2 About the Data

To obtain an embedding of the first district in Vienna, the respective WGW pages are necessary. To achieve this, the entire metadata of WGW was downloaded from the governmental open data platform ”data.gv.at” [Mag24]. This metadata collection includes information about all WGW pages, such as ID, name, GPS location, address, year, tags, object type and URL to the web page. The dataset contains metadata for 15.597 articles, from which only those concerning objects in the first district of Vienna were retained for further processing.

The dataset comprises several object types, from which the five largest groups were selected to be shown as unique groups, as illustrated in Appendix 7.1. The other groups were combined into the category ”Other”, which also includes all objects without a specified object type. The selected object types are: ”Building”, ”Institution”, ”Monument”, ”Traffic Area” and ”Sacral Building”.[4] Note that before the final assignment, the object types ”Verein”, ”Firma” and ”Behörde” were added to ”Institution”.

2.3 Data Pre-Processing

The CSV file from ”data.gv.at” [Mag24] was loaded and only entries with ”BEZIRK” equal to 1 were retained. However, after several iterations, it became apparent that some objects were missing from the data, as the district information was sometimes unavailable. In these cases, the ”BEZIRK_TXT” feature often contained the correct district information in text form. Therefore, the district data was imputed from this feature, completing the final selection of objects in the first district.

Next, for each object, the web page was accessed via the provided URL and the raw HTML corpus was stored. To extract only the text portion of the corpus, a specific method was required. Unfortunately, the main corpus was not encapsulated in a single ”div” of a specific

[2] Wiener Stadt- und Landesarchiv (MA 8)

[3] Wienbibliothek im Rathaus (MA 9)

[4] Original categories: ”Gebäude”, ”Institution”, ”Denkmal”, ”Verkehrsfläche”, ”Sakralbau”

class or ID but instead started with a series of empty divs and ended with an HTML comment, as illustrated below.

```
</div></div></fieldset></div></div></div></div></div></div>
</div></div></div>
MAIN TEXT
<!--
```

The raw string of the HTML corpus was thus cropped between these two identifying strings. Further pre-processing included detecting and transforming separate paragraphs within the div into one connected string. References (i.e., ”href” elements) were extracted for further analysis. Finally, all HTML tags were removed to obtain a clean text.

2.4 Creating Embeddings

2.4.1 About Embeddings

An embedding is a technique used to represent data, such as words or documents, as points in a multidimensional space where their relationships reflect similarities or meanings. This space is often conceptualized as a manifold, a complex mathematical structure, which allows data points to be organized in a way that preserves their intrinsic relationships. For instance, documents with similar meanings might cluster together on this manifold, aiding computers understand and process their underlying semantic connections more efficiently. In summary, similar items should have similar embedding vectors and thus be closer to each other, while dissimilar items should be far apart [Hen23].

2.4.2 Embeddings Model Selection

To create suitable embeddings, several different approaches were tried. Initially, smaller neural network architectures were used, which could not handle the raw text from the WGW pages directly. This was evident as the resulting embedding space was essentially just a random cloud of items without any meaningful structure. Consequently, the model was only fed a small list of relevant tags. Various types of input information were experimented with, such as name, extracted suffix of the name (e.g., ”kirche”, ”museum”, ”café”, ”markt”), address, WGW tags and the first few references to other pages. This approach appeared more promising, with similar items clustering together, as another qualitative review of the results revealed. However, the most promising results were obtained using the most downloaded sentence transformer from the Hugging Face platform.[5] This model, ”all-MiniLM-L12-v2” [Fac24], uses the first 256 tokens of the input text and maps it to a 384-dimensional dense vector space, representing Vienna’s first district manifold that will be visualized later. Figure 1 shows a sample text with exactly 256 tokens to illustrate the amount of information the model receives to place the objects in the landscape.

The ”all-MiniLM-L12-v2” sentence embedding model is based on Microsoft’s MiniLM architecture, specifically the MiniLM-L12-H384-uncased variant. MiniLM is a compact transformer model [VSP+17] designed for efficient language understanding and generation tasks, distilling knowledge from larger models like BERT while maintaining competitive performance [WWD+20]. The model architecture includes 12 layers, each with 384 hidden units and 12 attention heads, totaling 33 million parameters [Mic24].

[5] Last accessed on 01.07.2024 from https://huggingface.co/models?library=sentence-transformers&sort=downloads; the model was ranked first with 82.4 million downloads in the last month.

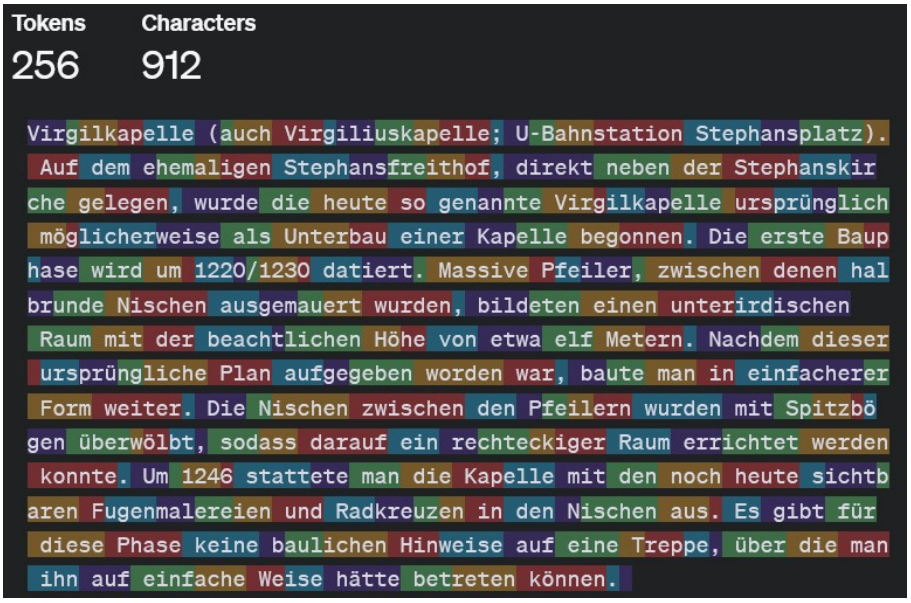


Figure 1: 256 tokens based on OpenAI tokenizer for illustration [Ope24].

The sentence embedding model is pre-trained on a corpus of more than one billion sentences. During this phase, the model learns to predict masked tokens in sentences and encode contextual information. This involves optimizing parameters to enhance the model's ability to understand and generate coherent text. Following pre-training, the model is fine-tuned for specific tasks such as sentence embedding using contrastive objectives. This involves further optimizing the model parameters by comparing the cosine similarity of sentence pairs within batches and applying cross-entropy loss to improve embeddings [Fac24]. The final pre-trained model is then published on Hugging Face and used in this project to create the embeddings.

2.5 Visualization & Frontend

As the created embedding consists of vectors of length 384, dimensionality needs to be reduced for visualization in 2D. To achieve this, t-Distributed Stochastic Neighbor Embedding (t-SNE) is used [MH08]. t-SNE is a popular technique for visualizing high-dimensional data. It aims to map complex data to two or three dimensions while preserving the local structure of the data points.

To avoid information overload and bad performance, the scope of visualization was reduced to focus on the most relevant objects. From the approximately 2.500 objects on WGW assigned to the first district, only around 1.000 were displayed. First, all objects ending in "gasse" or "straße" were removed, eliminating about 500 less relevant items. Then, for the object type "Building" (labeled "Gebäude" in the original database), only objects with WGW pages longer than 2.000 characters were retained. For all other object types, the page length had to be at least 300 characters. This distinction was made because buildings comprised about one-fifth of all data points. Lastly, political parties were excluded from the dataset. This process resulted in a final set of 981 objects, including 236 buildings, achieving a slightly more balanced distribution among different object types (see Appendix 7.1). To prevent overlapping labels in the final visualization, objects very close to each other were slightly shifted away from each other in the y-direction.

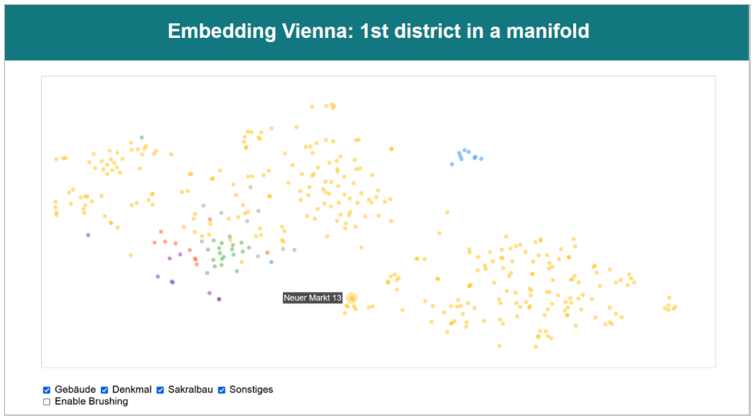
For the visualization, a static HTML page was created with dynamic features implemented

using JavaScript. The JavaScript library D3.js (Data-Driven Documents) [Bos11] was used for this purpose, as it is well-suited for creating dynamic, interactive data visualizations in web browsers. D3.js leverages modern web standards such as HTML, SVG and CSS, enabling users to bind data to a Document Object Model (DOM) and apply data-driven transformations to the document. This flexibility allows for the creation of highly customizable, interactive and efficient scatter plots, as needed for this project.

2.6 Development Process

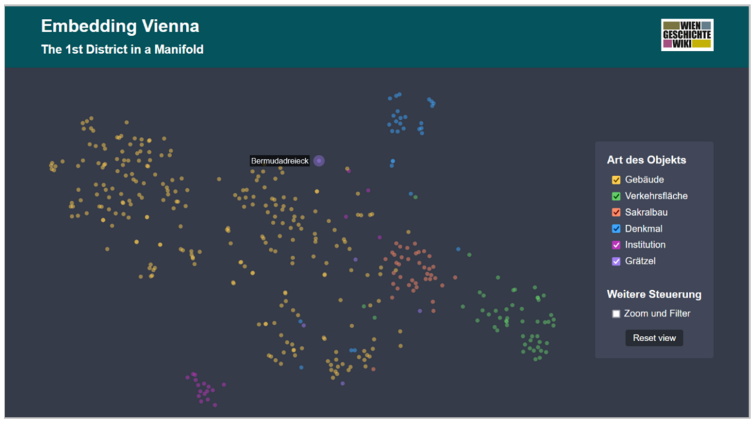
The development process followed an Agile-inspired iterative and incremental approach, which is commonly used in visualization design processes [Mun14]. This methodology emphasizes continuous refinement through successive cycles of development, testing and evaluation. Initially, a basic web page was created using D3.js to visualize initial attempts at embedding the objects. Based on user feedback and preliminary evaluations, iterative enhancements were made to both the embedding models and the user interface. This process involved experimenting with different input data and various natural language processing (NLP) models for embeddings, using frameworks like TF-IDF [PVG+24], spaCy [AI20], Doc2Vec [LM14] and Hugging Face Sentence Transformers [RG19] to improve the relevance and accuracy of the embeddings.

The approach was data-driven, meaning that features and embeddings were created based on the available data. Concurrently, the UI was incrementally enriched with additional features to enhance usability and interactivity based on user feedback and performance evaluations. An overview of the main UI development stages can be found in Figure 2.



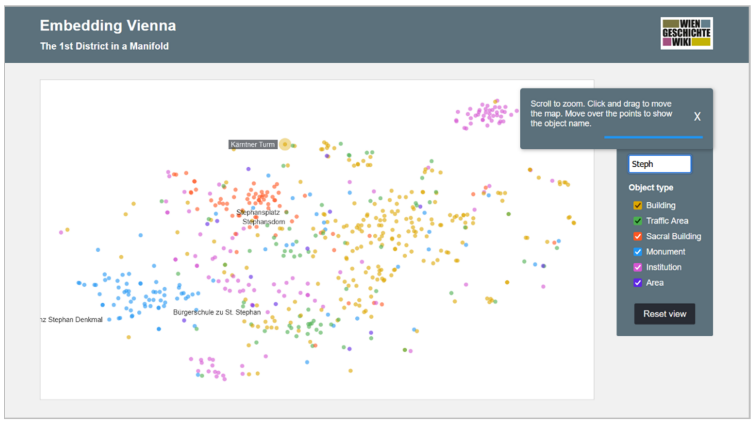
28.05.2024

First working version: hover effects, colouring and filtering by object type



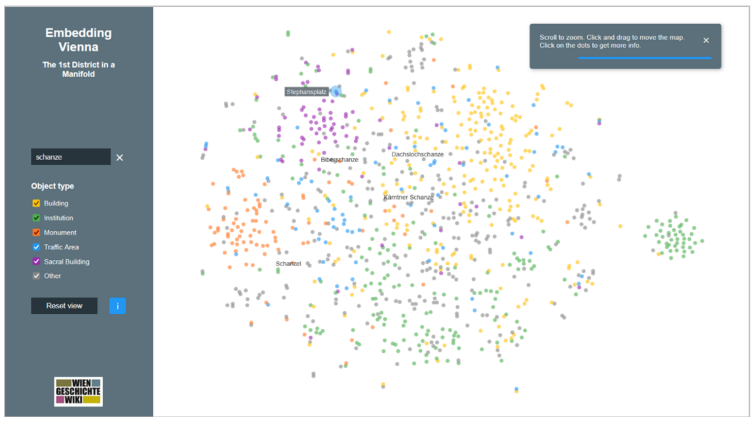
04.06.2024

Design updates, first working but unintuitive zoom (“brushing”), reset button



24.06.2024

Design updates, toast notification, first simple search, lots of bugs when zooming/filtering and searching at the same time



12.07.2024

Final design: info button with more information, clear search input button, bug free

Figure 2: Development process. Main stages of UI/UX design and embedding changes.

3 Final Result

Ben Schneiderman's mantra [Sch96], "Overview first, zoom and filter, then details on demand", encapsulates the principles of effective information visualization. It emphasizes the importance of presenting an initial overview, allowing users to zoom in on areas of interest and providing detailed information upon demand. This approach helps users manage complexity and explore data at different levels of granularity. Moreover, Krug's [K+14] principle "Don't Make Me Think" emphasizes creating intuitive and user-friendly designs where users can navigate and interact with a website or application effortlessly, without having to stop and figure out how things work. The goal is to minimize cognitive load and make interactions as straightforward and self-evident as possible. The final visualization design tries to follow these two principles, providing an appropriate level of granularity at each stage of the user journey and an intuitive, simple to use UI.

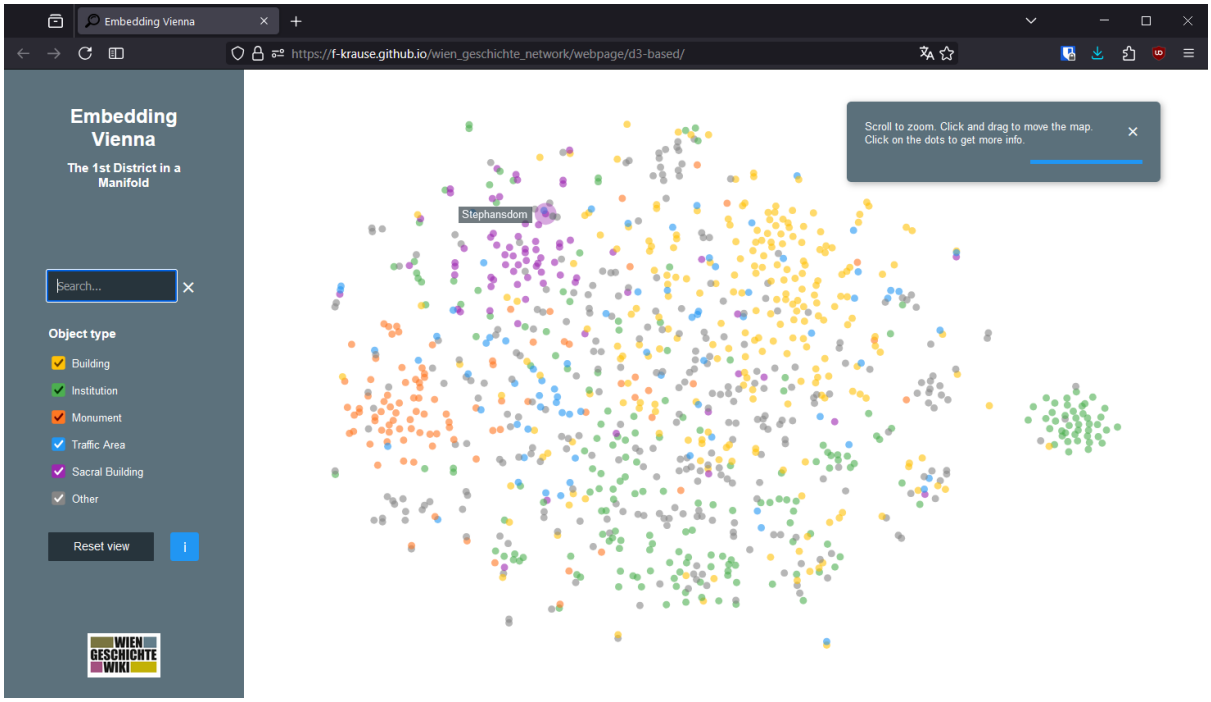


Figure 3: Final UI

The final, interactive result can be accessed at f-krause.github.io/wien_geschichte_network/webpage/d3-based/. In the following subsections, its features will be explained in more detail.

3.1 Landing Page

At first, the user is presented with a dot cloud, providing a rough overview and clustering of object types, as shown in Figure 3. This aligns with Schneiderman's mantra [Sch96] of creating an overview first, allowing users to identify clusters of similar objects quickly. For instance, coffee houses are clustered on the right, monuments are mostly on the left, sacral buildings are in the top left, buildings are in the top right and institutions are located at the bottom and to the right.

Using a scatter plot colored by object type adheres to the expressiveness and effectiveness principles defined by Tamara Munzner [Mun14, p. 101]. The expressiveness principle states

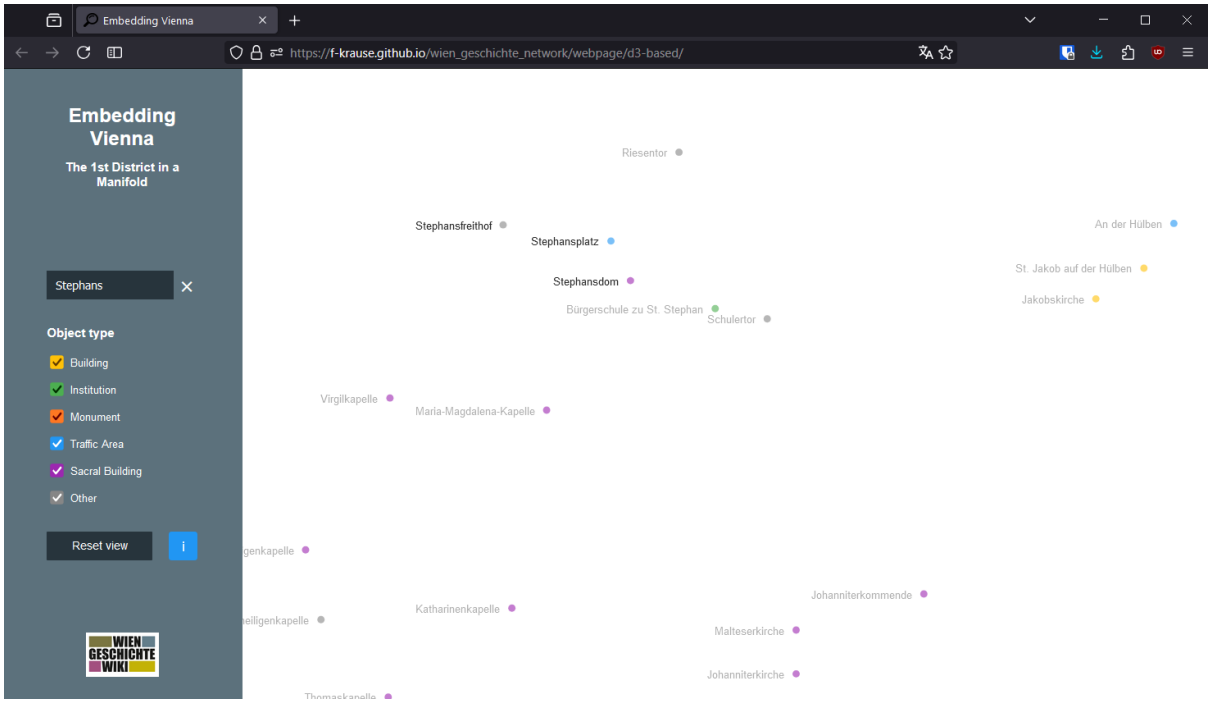


Figure 4: Zoomed UI – showing displayed labels of objects

that the visual encoding should express all and only the information in the data, while the effectiveness principle states that the importance of the attribute should be matched with the salience of the channel. These concepts are fulfilled by the current visualization. Information is easily retrievable as the relatedness of objects is encoded by position and the categorical object type is encoded by color.

Additionally, the design follows Edward Tufte's design principles [Tuf90, Tuf01], which advocate for reducing non-essential elements (often referred to as "chartjunk"), ensuring the data itself is the focal point. Thus, the visualization initially only shows the colored dots by object type. Tufte promotes high data intensity and high data-ink ratios, meaning that a large portion of the ink used should display the data rather than decorative or redundant features. Here, nearly all the "ink" visualizes the dots, which are sized to avoid excessive overlap while being large enough to be seen clearly. Compared to older designs in Figure 2, the final one uses much more of the available screen space by consolidating all design, informative and control elements into a single bar on the left, freeing up space for data display.

Aligned with the last part of Schneiderman's mantra, providing details on demand, users can click on the dots to be directed to the respective WGW page and hover over the dots to display the object name. To improve the user experience (UX), a transparent, larger circle is positioned above each dot. This way, the user does not need to position the cursor precisely on the small dot but only in its proximity. When hovering over a dot, this transparent larger circle is colored, providing instant feedback.

On the left-hand side of the plotting area, the navigation bar is displayed. This bar provides features to filter, search, reset and get more details, which will be described in more detail in the following chapters.

3.2 Zoom & Pan

By implementing zooming and panning, users can filter specific objects and explore areas of interest in greater detail, as shown in Figure 4. This feature aligns with the middle statement of Schneiderman's mantra [Sch96]. When zooming in sufficiently, all object names are displayed simultaneously, providing a clearer overview. Additionally, if the user enters a search term, objects not relevant to the search string will still be displayed but in a lighter grey color.

Zooming and panning, achieved through scrolling and click-and-drag actions, exemplify the "Don't Make Me Think" principle by Krug [K+14], which prioritizes user-friendly design to minimize cognitive load. This approach facilitates seamless navigation in an intuitive manner, allowing users to focus on tasks and content rather than interface mechanics and promoting efficiency and user satisfaction.

Implementing zoom and pan in a user-friendly manner was challenging. The difficulty lay in maintaining the zoom and pan state (i.e., leaving the display unchanged) whenever the user changes the selected object types or searches for an object, as these actions update the database and thus the display.

Triggering the display of object labels next to the dots is based on a simple heuristic dependent on the number of selected object types. The assumption is that deselecting object types reduces the number of displayed dots, creating more space for labels to be displayed sooner without excessive overlapping.

3.3 Object Type Selection

As described in Section 2.2, five object types and the catch-all group "other" have been used to color the objects on the map. In the navigation bar to the left, users can select which types to display by clicking on the checkboxes (see Figure 3). This functionality is designed to be self-explanatory and intuitive. The checkboxes also serve as the legend for the dots, consolidating all relevant information about the object types in one place. This design follows the previously mentioned principles, enabling users to filter objects of interest [Sch96] while maintaining a low cognitive load for interactions [K+14].

3.4 Search

The last major feature is the search bar. By typing a search term into the bar located in the navigation panel (see Figure 4), matching objects will be highlighted. If the user is at a zoom level where no labels are shown, only the labels of matching objects will be displayed. If the user is already zoomed in far enough that all labels are shown, only the matching ones will remain in black while the others will be greyed out. The search input can be deleted anytime by clicking on the "x" next to the search bar.

To enhance the user experience, several tweaks were made to the simple string-matching search algorithm. Firstly, labels are only displayed after two characters have been entered, preventing the view from becoming cluttered with overlapping labels starting with the same initial letter. Additionally, the input string is converted to lowercase and normalized, meaning special characters are converted to their "base version". For example, "Ö" becomes "o" and "é" becomes "e". This normalization is also applied to object names before matching, allowing users to find "Café Français" by typing "cafe francais" into the search bar.

Implementing this feature was challenging too, as the display needs to update (i.e., the displayed labels) with each character typed or deleted, depending on the current zoom level.

The search function must determine if the user is currently zoomed out and does not see any labels or if the user is zoomed in and sees all labels, requiring irrelevant ones to be greyed out.

3.5 Additional Features

In addition to the main features previously mentioned, several smaller enhancements have been made to further improve the user experience.

To provide initial guidance for new users, a small toast notification appears for 15 seconds, explaining the most basic controls (see Figure 3, top right corner). Additionally, clicking on the blue ”i” button in the navigation bar triggers a pop-up, as in Appendix 7.2, that offers concise information about the display, tool usage, methods and attribution. Users can find additional information about objects by clicking on the respective dots. Clicking on the WGW logo at the bottom left redirects to the WGW landing page. A ”reset view” button allows users to reset the object type selections and the main view.

Since the website is primarily optimized for devices with a mouse and large screen (i.e., classic PC setup), a warning is shown to mobile users, as can be seen in Figure 5. However, the application still functions on smaller devices. For very narrow screens, the navigation bar is removed, leaving only a header with the title and WGW logo. Despite this, the embedding remains fully functional: users can zoom, labels will be displayed and clicking on dots redirects to the respective WGW page. When the device is rotated or for devices with wider screens, the navigation bar will reappear.

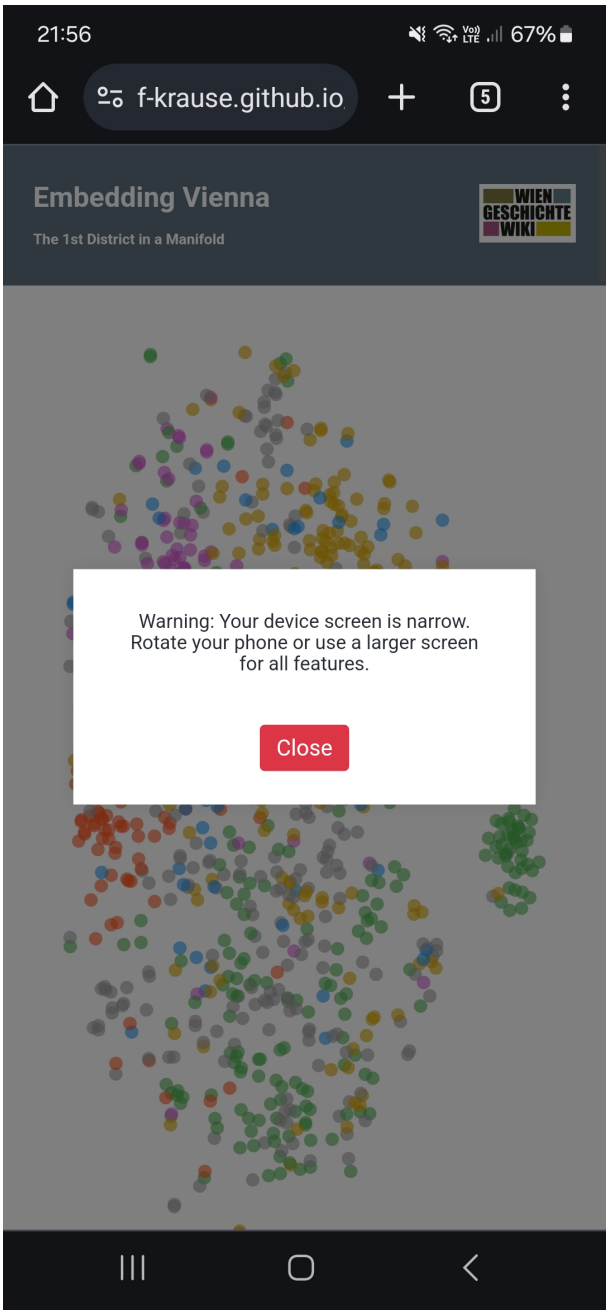


Figure 5: Mobile device users warning.

4 Challenges

One of the primary challenges in developing this tool was creating embeddings that accurately captured the semantics of the WGW articles, thereby appropriately representing the relatedness of historical objects. Initially, handcrafted input lists of relevant keywords were used, which led to promising results after several iterations of improvement and trial and error. Transitioning to a large sentence transformer model that processes the beginning of an article as input significantly improved semantic accuracy.

Another substantial challenge was ensuring a seamless user experience (UX). A critical aspect of this was maintaining the position of plotted points and their labels whenever the user interacted with the interface – whether through object type selection, using the search bar, zooming, or panning. Implementing this feature required meticulous attention to detail to prevent the view from resetting or labels from disappearing with each interaction. For instance, managing the dynamic display of labels so that all labels stay visible when changing the search string while zoomed in required extensive experimentation and complex logic.

Furthermore, a significant bug surfaced late in the development cycle, which led to the omission of around 250 objects of type ”Other” from the visualization. This issue was discovered and rectified only in the project's final days, underscoring the challenges inherent in handling and visualizing datasets.

5 Limitations

Despite the tool's innovative approach and interactive features, several limitations need to be considered.

Firstly, the current scope of the tool is restricted to approximately half of the objects in Vienna's first district. The WGW contains a far greater number of objects, not only within the first district but across the entirety of Vienna. Consequently, the tool only offers a partial view of the historical landscape.

There is also the issue of potential irrelevant object positioning. The embeddings were generated using only the first 256 tokens of each article, a relatively small amount of information that may not fully capture the context and nuances of the historical objects. Additionally, the embedding fully relies on the content in the articles and hence is impacted by faulty content and misleading keywords. This limitation means that some objects might be inaccurately positioned within the manifold, potentially leading to confusing or irrelevant associations.

Moreover, while the chosen method for obtaining embeddings proved effective, it is by no means the only approach. Numerous other techniques could potentially yield more accurate or contextually rich embeddings. Future iterations of the tool could explore alternative models and methodologies to enhance the accuracy of the embeddings further.

Performance issues also pose a limitation, particularly on older and weaker hardware. The visualization involves nearly 1.000 plotted objects and their labels, resulting in a high number of DOM elements that can strain the browser's rendering capabilities. This can lead to lag and reduced responsiveness, negatively impacting the overall user experience. Addressing these performance concerns will be crucial for making the tool more accessible and user-friendly across a wider range of devices. However, this would require a more fundamental framework change.

6 Conclusion & Outlook

The development of the interactive visualization tool for Vienna's first district was a data-driven, iterative and incremental process. The primary objective was to create a tool that effectively represents the relationships and similarities among historical objects through embeddings. The project combined advanced natural language processing techniques with principles of visualization design to deliver an informative and user-friendly interface.

The initial phase involved creating simple embeddings from handcrafted lists of keywords. Although these provided some structure, the transition to a sentence transformer model significantly improved the accuracy and relevance of the embeddings. This model, processing the first 256 tokens of each WGW article, was instrumental in mapping objects into a meaningful multidimensional space. Dimensionality reduction using t-SNE allowed for a visual representation in 2D, maintaining the local structure of the data points.

A decent user experience (UX) was a core focus throughout the development process. The final user interface (UI) incorporates features like zooming, panning, a search bar and dynamic label displays to enhance usability. Implementing these features while maintaining performance and preventing interface resets was challenging but necessary for a seamless UX. User feedback was instrumental in refining these elements, ensuring the tool met usability and accessibility standards. Principles of visualization design, such as Schneiderman's mantra [Sch96], Krug [K+14] and Tufte's [Tuf90, Tuf01] emphasis on clarity, guided the development process, ensuring the tool is both informative and user-friendly.

Despite its achievements, the tool has several limitations. It currently only covers a portion of the historical objects in Vienna's first district and the embeddings rely on a limited amount of text from each article. Performance issues, particularly on older hardware, also pose challenges. However, these limitations highlight areas for future improvement.

Looking ahead, there are several exciting prospects for enhancing and expanding this tool. Future work may include the development of more sophisticated search algorithms and improved embeddings to enhance the accuracy and relevance of the displayed information. Rewriting the code with canvasJS [Fen24] or WebGL [Con24] could significantly improve performance, enabling better scaling and more fluid interactions also on older hardware. Additionally, incorporating tooltips with the most relevant information upon clicking on dots would enrich the user experience by providing immediate context. These tooltips could for example contain an image, a date, some tags, a short description and a link to its WGW page. One could also experiment with different embeddings tailored for specific tasks such as temporal analysis, geographic location or object type, coupled with more extensive filtering options by tags like "early modern period", "cafe", "chapel", year ranges and others.

In conclusion, this project successfully developed an innovative and interactive tool for exploring the manifold of historical objects in Vienna's first district. While there are areas for future improvement, the tool represents a novel approach compared to traditional search methods, offering a panoramic and engaging exploration experience. By integrating advanced NLP with effective visualization techniques, it provides a powerful means of exploring historical data. This enhances the understanding and appreciation of Vienna's rich historical heritage, making it more accessible and engaging for everyone. The project's success demonstrates the potential of combining cutting-edge AI with thoughtful design to unlock new ways of playful interaction with Vienna's historical landscape.

7 Appendix

7.1 Object Types and Counts in Final Display

Note that in the final display only the first six groups were shown. All other groups were added to ”Sonstiges” (”other”).

Object type	**Count**
Gebäude	236
Sonstiges	124
Institution	118
Denkmal	94
Verkehrsfläche	76
Sakralbau	72
Verein*	39
Firma*	34
Behörde*	18
Grätzel	14
Grünfläche	10
Brücke	9
Anstalt	9
Markt	3
Friedhof	3
Gewässer	2
Bezirk	2
Passagen	2
Park	2
Fonds	1
Vorstadt	1
Konfessionelle Verwaltungseinheit	1
Berg	1
Total	981

*) these groups were added to ”Institution” for the broader picture

7.2 Additional Info Pop-up

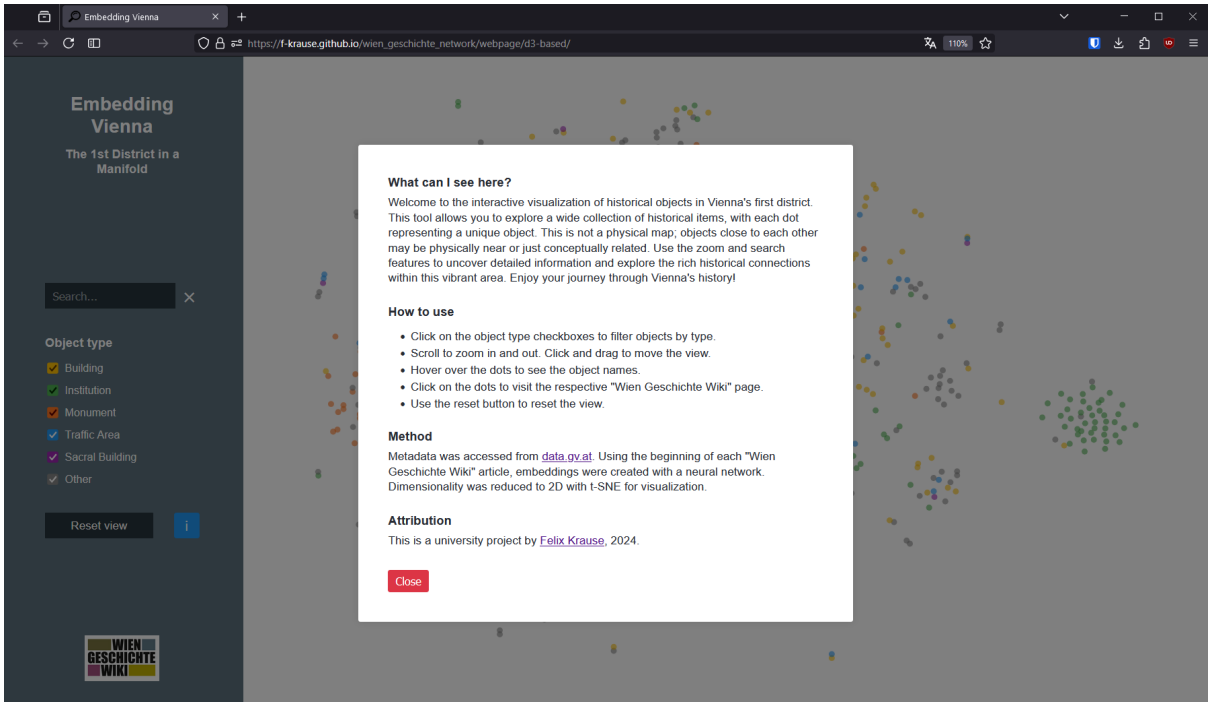


Figure 6: Pop-up displayed when a user clicks on the blue info button in the navigation bar.

References

[AI20] Explosion AI. spacy: Industrial-strength natural language processing in python. https://spacy.io, 2020. Version 2.3.

[Bos11] Mike Bostock. D3.js - data-driven documents, 2011. Accessed: 05.07.2024.

[Con24] Mozilla Contributors. WebGL API. https://developer.mozilla.org/en-US/docs/Web/API/WebGL_API, 2024. Accessed: 14.07.2024.

[Cze04] Felix Czeike. *Historisches Lexikon Wien*, volume 2. Kremayr & Scheriau, 2004.

[Fac24] Hugging Face. sentence-transformers/all-MiniLM-L12-v2: Transformer model pre-trained on diverse text data for sentence embeddings using MiniLM-L12 architecture. https://huggingface.co/sentence-transformers/all-MiniLM-L12-v2, 2024. Accessed: 01.07.2024.

[Fen24] Fenopix. CanvasJS: JavaScript Charts Made Easy. https://canvasjs.com/, 2024. Accessed: 14.07.2024.

[Hen23] Kevin Henner. An Intuitive Introduction to Text Embeddings. https://stackoverflow.blog/2023/11/09/an-intuitive-introduction-to-text-embeddings/, November 2023. Accessed: 01.07.2024.

[K+14] Steve Krug et al. Don't make me think, Revisited. *A Common Sense Approach to Web and Mobile Usability*, 2014.

[LM14] Quoc Le and Tomas Mikolov. Distributed representations of sentences and documents. In *Proceedings of the 31st International Conference on Machine Learning*, pages 1188–1196. PMLR, 2014.

[Mag24] Magistrat Wien - Magistratsabteilung 8 - Wiener Stadt- und Landesarchiv. Wien Geschichte Wiki - Semantische Daten. https://www.data.gv.at/katalog/de/dataset/wien-geschichte-wiki, 2024. Downloaded: 28.04.2024.

[MH08] Laurens van der Maaten and Geoffrey Hinton. Visualizing data using t-SNE. *Journal of machine learning research*, 9(Nov):2579–2605, 2008.

[Mic24] Microsoft. microsoft/MiniLM-L12-H384-uncased: MiniLM-L12 model, a small and fast Transformer pre-trained with MLM on English text, lowercased. https://huggingface.co/microsoft/MiniLM-L12-H384-uncased, 2024. Accessed: 01.07.2024.

[Mun14] Tamara Munzner. *Visualization analysis and design*. CRC press, 2014.

[Ope24] OpenAI. OpenAI Tokenizer. https://platform.openai.com/tokenizer, 2024. Accessed: 07.07.2024.

[PVG+24] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay. Scikit-learn: Machine Learning in Python. https://scikit-learn.org/stable/modules/generated/sklearn.feature_extraction.text.TfidfVectorizer.html, 2024.

[RG19] Nils Reimers and Iryna Gurevych. Sentence-bert: Sentence embeddings using siamese bert-networks. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing*. Association for Computational Linguistics, 11 2019.

[Sch96] Ben Schneiderman. The eyes have it: A task by data type taxonomy for information visualizations. *Visual Languages, 1996. Proceedings., IEEE Symposium on*, pages 336–343, 1996.

[Tuf90] Edward R Tufte. *Envisioning Information*. Graphics Press, 1990.

[Tuf01] Edward R Tufte. *The Visual Display of Quantitative Information*. Graphics Press, 2001.

[VSP+17] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. *Advances in neural information processing systems*, 30, 2017.

[Wie24] Wien Geschichte Wiki. Über das Wien Geschichte Wiki. https://www.geschichtewiki.wien.gv.at/%C3%9Cber_das_Wien_Geschichte_Wiki, June 2024. Accessed: 01.07.2024.

[WWD+20] Wenhui Wang, Furu Wei, Li Dong, Hangbo Bao, Nan Yang, and Ming Zhou. Minilm: Deep self-attention distillation for task-agnostic compression of pre-trained transformers. *Advances in Neural Information Processing Systems*, 33:5776–5788, 2020.