Visual Clutter Reduction through Hierarchy-based Projection of High-dimensional Labeled Data

Dominik Herr[1,2] Qi Han[1] Steffen Lohmann[1] Thomas Ertl[1]

[1]Institute for Visualization and Interactive Systems (VIS)*
[2]Graduate School of Excellence advanced Manufacturing Engineering (GSaME)
University of Stuttgart, Universitätsstraße 38, 70569 Stuttgart, Germany

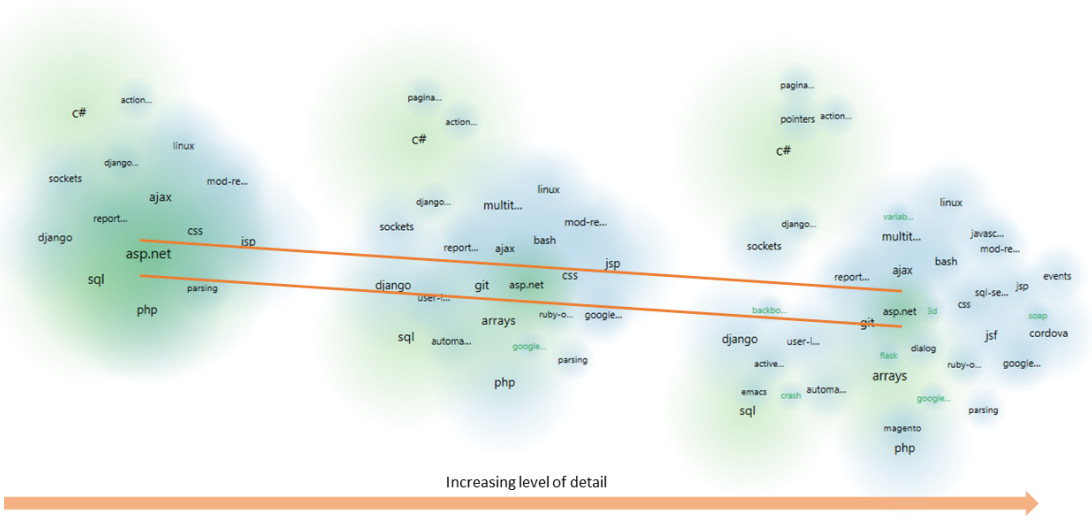

Figure 1: Our approach reduces visual clutter of projected labels by showing the user the relatedness of the labels on increasing levels of detail. A co-occurrence based hierarchy is used to generate the levels of detail. Here, the cluster *asp.net*, highlighted with a green background, and its related clusters, marked in with a slightly lighter green, are shown in increasing levels of detail. To follow the cluster more easily, we trace it with two lines in this depiction.

Abstract

Visualizing high-dimensional labeled data on a two-dimensional plane can quickly result in visual clutter and information overload. To address this problem, the data usually needs to be structured, so that only parts of it are displayed at a time. We present a hierarchy-based approach that projects labeled data on different levels of detail on a two-dimensional plane, whilst keeping the user's cognitive load between the level changes as low as possible. The approach consists of three steps: First, the data is hierarchically clustered; second, the user can determine levels of detail; third, the levels of detail are visualized one at a time on a two-dimensional plane. Animations make transitions between the levels of detail traceable, while the exploration on each level is supported by several interaction techniques. We demonstrate the applicability and usefulness of the approach with use cases from the patent domain and a question-and-answer website.

Index Terms: H.5.2 [Information Interfaces and Presentation]: User Interfaces—GUI; H.3.3 [Information Storage and Retrieval]: Information Search and Retrieval—Clustering

*{firstname.lastname}@vis.uni-stuttgart.de

Graphics Interface Conference 2016
1-3 June, Victoria, British Columbia, Canada

1 Introduction

The abundance of data produced nowadays is often too much to show all at once and its structure regarding content and relation is often either unavailable or unknown. A common practice to label the content is using keywords or tags that originate from predefined metadata, a classification structure, or from extracted content. In addition, the overview of the relation between data is often simplified by clustering the data and then showing the clusters' relations. The labels' relations span a high-dimensional space that is based on the co-occurrence of data objects' labels. To visually analyze the labels' coherence, the labels may be shown on a two-dimensional plane, for example through projection. Yet, the two-dimensional visualization of many labels leads to the problem of visual clutter. This visual clutter combined with a possibly complex relationship measure can cause heavy cognitive load for an analyzing user and hinders the efficiency of the analysis. One common approach to ease this problem is to introduce a hierarchy to the labels, wherein a child node specifies the description of its parent node.

However, the visualization of such hierarchies usually focuses on the relationship between the parent and its children. Apart from the intuition that siblings with the same parent are closer related to each other than other data objects on the same hierarchy level, the child-child relationship remains hidden in such visualization approaches.

We present an approach that aims to reduce the visual clutter of two-dimensional projections of a large number of labels through a smooth transition between overview and detail [31]. This is achieved by hierarchizing the labels based on a similarity matrix derived from the co-occurrence of the labels. Then we show

first highly aggregated clusters and then provide more details as the clusters are split apart. Also, our approach represents the child-child relationship of the generated hierarchy at a given level of detail through the projection of the relevant labels onto a two-dimensional plane. To do so, the approach is divided into three steps:

1. A hierarchy is generated from a given similarity matrix. Here, this matrix is based on the co-occurrence of the labels.

2. Based on the extracted hierarchy, varying levels of detail are determined for the information shown to the user; the user can change those levels at will.

3. The information contained in the topmost level of detail is projected onto a two-dimensional plane, and the user can explore the shown data or increase the level of detail.

2 Related Work

The following discussion of related work is divided into four parts. First, we examine works from the field of information visualization that use dimension reduction to visualize data spatially. Afterwards, we review visualization methods for hierarchical datasets and related interaction techniques. Then, we show visualization approaches for hierarchy-based data. At last, we present recent work in the area of word cloud visualization that are related to our work.

2.1 Dimension Reduction and Data Projection

One common approach to visualize the content in datasets, for example document collections, is to take extracted features, such as keywords or concepts, and compare the data based on those features. The natural language processing community developed many ways to extract keywords from documents [15]. The relatedness of such features is usually pairwise computed and therefore presents a high-dimensional reduction problem. Methods to solve this problem can be projection-based, for instance, by using Principal Component Analysis (PCA) [38], Multidimensional Scaling (MDS) [23], Least Squares Projection [27], or t-Distributed Stochastic Neighbor Embedding (t-SNE) [35]. Van der Maaten et al. [36] survey a number of techniques to project high-dimensional data onto a low-dimensional space. In such approaches, data is usually visualized as data points that take up almost no space. This representation suffers from visual clutter when the data is not clearly separable, which is even more problematic when the data is represented with labels.

Due to the high complexity of projection techniques, other potentially faster and more intuitive approaches, such as force-based layouts [12, 13] are often used when interactive visualizations are needed. However, García-Fernández et al. [14] concluded in their study that projection-based approaches are superior to force-based layouts, when a complete and large dataset needs to be visualized.

Some approaches are based on neural networks, such as hierarchical self-organizing maps [22]. When this approach is used for each created area, as proposed in [32] or [10], this approach creates a visualization seemingly based on a hierarchy. However, when generating a visual hierarchy this way, only elements contained in an area are mapped relative to each other. In our approach, we use a hierarchy not only to visualize the relation between parents and their children, but also between the siblings across the clusters.

2.2 Hierarchical Aggregation and Visualization

In case the data is already hierarchically structured, there are many approaches to visualize such data. If the user needs to inspect the data's distribution across the hierarchy tree, tree visualization techniques, such as dendrograms or icicle plots, are commonly used. However, dendrograms do not provide information about the relation between the elements across multiple clusters (even when they are at the same hierarchy level). Another prominent method to show a hierarchy's structure are treemaps [30]. Due to a treemap's layout, its ability to provide information about the relation of clusters with different parent clusters is limited and it is almost impossible to indicate possible shifts of the clusters' similarities on different levels of detail as tree maps assume a fixed hierarchy without considering a possible relation between the clusters, which we want to provide.

The goal of most hierarchical visualization techniques is to show the structure of the hierarchy. Elmqvist and Fekete [9] propose guidelines how hierarchical data can be used to limit the amount of shown information. We adapt some of the proposed guidelines in our work, such as taking the most important element of a cluster as cluster representative.

2.3 Visualization Approaches for Hierarchy-based Data

There are various approaches to provide information about more details about hierarchy-based data. For example, Dou et al. [8] generate topic models where the users can interactively modify the created hierarchical structure. Afterwards, the user can inspect the development of individual or groups of topics over time. In contrast to our approach, this attempt uses the hierarchies to aggregate topics for the analysis of changes over time. Instead, we use hierarchies to filter shown data and show the data's relation spatially. Like most hierarchical visualization approaches, they assume the availability of a hierarchical structure and use predefined hierarchy levels to show information. Our approach goes beyond that by enabling the users to set the shown hierarchy levels by themselves.

Liu et al. [24] developed an approach to build hierarchies based on topic graphs. They visualize the relations of extracted topics by using stacked trees in combination with force-based graph layouts. In contrast to our approach, they use hierarchies to distinguish between topics and not to set their content in a relation to other topics. Our approach aims not only to provide relational information across clusters, but also of their content when more information is shown.

Fried and Kobourov [11] presented a system, wherein they map the titles of papers in the DBLP database onto a two-dimensional landscape based on a hierarchy. The users can create a search profile whose results are highlighted on the landscape using a heat map visualization. They focus on temporal aspects of the data.

Wise et al. [37] proposed to show large document collections through a galaxy metaphor, in which every document is represented by a star. By doing so, the user gets a more intuitive understanding of the relations between documents. Similarly, SPIRE [34] and INSPIRE [39] use the same metaphor, but they combine it with a visual analytics approach to enable users to further analyze the data. The STREAMIT system [1] uses force-based layouts, clustering discovery techniques and topic modeling to visualize document streams in real-time. The clustering is based on the graph layout and does not create a hierarchy to examine the document streams on a semantical level. An early version of the recently published *Overview* system [4] projects documents onto a two-dimensional scatterplot, whilst showing a hierarchical tree structure of the documents in another view. The system uses brushing and linking to connect those two visualizations. However, the selected elements of the hierarchy view are not represented in the amount of data shown in the scatterplot.

Only a few of these landscape-based approaches support hierarchical data, and most that do assume the hierarchy to be given. Thom et al. [33] use hierarchical topic clustering combined with a treemap-based visualization to show Twitter data on different levels of detail. This level of detail can be interactively steered by the user during an analysis run, which enables the user to see more details about a specific topic. However, the visualization has similar drawbacks as described in subsection 2.2. As a result, the positions of the topics is based on the hierarchy, but the relation of the various topics is hard to comprehend.

In addition to approaches that try to depict the global relatedness

of data, there are approaches that show the local relatedness, depending on a user-chosen subset of the data. Often such approaches are realized through the use of focus+context techniques. One example for the interactive exploration of locally relevant information is the Proxilens approach by Heulot et al. [20]. Here, a lens draws related data towards the data in the lens's center and pushes irrelevant information out of the bounds of the lens.

2.4 Word Cloud Visualization

Also related to our work are word cloud visualizations that show the most frequent words of a text as a weighted list in some specific spatial arrangement, such as a sequential, circular, or clustered layout [25]. Several variations and advancements of word clouds have been proposed in recent years. For instance, Seifert et al. [29] developed algorithms for space-filling word clouds based on a set of heuristics, while related layout algorithms have also been presented in a number of other works.

Some layout strategies consider word relationships and implement spatial arrangements where strongly related words are placed in close proximity, similar to our attempt. The layout strategies range from simple line-by-line approaches [16] to treemap-like layouts [21] and force-directed placements in combination with Venn diagrams [6]. Some works even apply projection techniques, such as the aforementioned MDS, to reflect the relatedness of words [28]. There are also attempts to explicitly depict the relationships in word clouds, either by adding links between related words or via interactive highlighting [18]. Prefix Tag Clouds [5] make use of prefix trees to group different word forms, whereas the Word Cloud Explorer uses advanced NLP processing to link word forms and to support the visual analysis of text documents via interactive word clouds [18].

However, we are not aware of any word cloud visualization that projects the words on multiple layers and allows for a seamless interaction and exploration on and between the layers.

3 Approach

The main goal of our approach is to reduce visual clutter caused by the projection of labels onto a two-dimensional plane. We do this by introducing a hierarchy in the projection, providing a smooth transition between overview and detail. At the same time, we aim to preserve and indicate the relationships between labels on different levels of detail. Those levels of detail are based on the hierarchy generated in a preprocessing step, while the distribution of the labels on each level of detail supports the users' intuition that related labels tend to be placed in close proximity.

As aforementioned, our approach consists of three steps, shown below. In the following, these steps are explained in more detail.

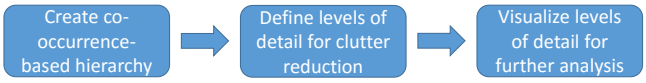

Figure 2: Workflow of our approach

3.1 Data Preprocessing

The creation of a hierarchy through clustering is expensive regarding computation time. As there is no need to recompute a dataset's hierarchy unless the dataset itself changes, we compute the hierarchy in a preprocessing step and store it for later use. This ensures a quick and smooth entry point into the analysis, as information about the hierarchy of the dataset can be shown right away.

Step 1: Hierarchization To create the hierarchy, we use Hierarchical Agglomerative Clustering (HAC) [26], which initially treats every data element as an individual cluster. Afterwards it iteratively merges the two most similar clusters based on a linkage criterion. Often, HAC-based clustering approaches use single-linkage as a linkage criterion. In single-linkage, the distance of two clusters is defined by the distance of the two closest elements of the clusters. However, it is unclear what merging clusters this way means on a semantic level [17, p. 525].

Therefore, we used medoid-linkage as the linkage criterion. The similarity of two clusters is defined by the similarity of the clusters' medoids. A medoid is the element within a cluster with the least total distance to all other elements within the cluster. As the calculation of a medoid includes all elements within a cluster, it is a better way to describe a cluster's content compared to the elements used by single-linkage. Further, medoid linkage implicitly compensates for a high variance within the cluster and is more robust to outliers within the cluster [26, p. 392;398].

One drawback of using a medoid linkage is the lack of monotonicity regarding the similarity of the merged clusters, as the medoid of a merged cluster may differ from the clusters that were merged. Therefore, a cluster's medoid usually needs to be recomputed as soon as the cluster changes.

3.2 Recurring Steps of Analysis

The following steps are executed every time an analysis is performed. As users may be interested in defining different levels of detail for each run of an analysis, the visualized information is likely to change and, hence, needs to be computed on demand.

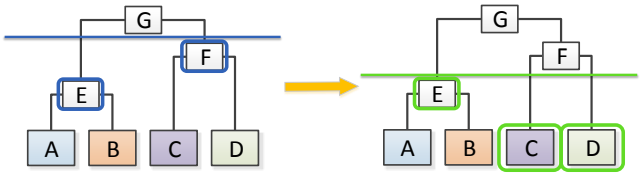

Figure 3: The labels shown on a given level of detail is similar to a cut in the dendrogram of the hierarchy. The first nodes after the cut represent relevant clusters for that level of detail (they are highlighted by a border in the figure). These clusters' labels are then used in the subsequent visualization step.

Step 2: Setting the Levels of Detail As our hierarchy is generated through a hierarchical agglomerative clustering, the hierarchy's branching is binary. Thus, a given level of depth of the hierarchy describes the amount of shown clusters at the same time. An analysis of the hierarchy by stepping through it one level of depth at a time can be extremely tedious as only one cluster would split apart and therefore is not viable for our approach. To reach our before mentioned goal, we enable the user to step through multiple levels of depth at once. In our approach, the chosen levels of depth are called levels of detail. A level of detail corresponds to a cut at the level of depth within the hierarchy. Only the clusters directly below the cut are being shown to the user. An example illustrating the idea of this is shown in Figure 3.

Therein, the cut is at first after the cluster G and the clusters E and F are the next clusters directly below the cut. In the second step, the cut is below F. Cluster E remains, but cluster F, which is now above the cut, is replaced by the clusters C and D.

Initially, we automatically set the levels of detail to represent an increase of 20 levels of depth. This ensures that each level of detail shows a comprehensible amount of new information. Afterwards, the user may add, change or remove any number of levels of detail. To support the user in this task, we show two plots that contain information about the hierarchy (see Figure 4 for an example).

Figure 4ⓐ shows the merged clusters' similarity. Every point of the plot represents the merging of two clusters. The similarity is expressed in the y-value of the points, whereas the clustering step is equal to the points' x-value. A higher similarity value indicates that the clusters are much alike, whereas a low value indicates little overlap regarding the co-occurrence of clusters' medoids.

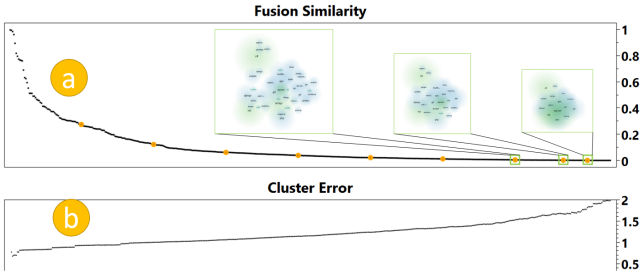

Figure 4: We initially present two plots. The fusion similarity (ⓐ) indicates the similarity of the clusters that have been merged over the course of the hierarchization. The cluster error (ⓑ) shows the clusters' correctness using the Davies-Boulding index (a lower value indicates better clusters). Also, initial cuts for the levels of detail are proposed to the user. For demonstration purposes, we use a reduced number of levels of detail and added small depictions of the visualizations that result on three different levels. The same visualizations are shown in larger size in Figure 1.

Figure 4ⓑ gives an indication of the quality of the clustering at a given step. As an indication measure, we use a variation of the Davies-Bouldin index [7]. The Davies-Bouldin index is designed to have a high value when the distance between clusters is low and the distance within the clusters is high. We assume that the similarity value s is normalized and calculate the distance d of two clusters C_i and C_j as $d_{i,j} = 1 - s_{i,j}$. Since we use medoid linkage, whereas the Davies-Bouldin index uses centroid distances, we slightly adapted the index. Our modified Davies-Boulding index DBI_{mod} is calculated as

$$DBI_{mod} = \frac{1}{n} \cdot \sum_{i=0}^{n-1} \cdot \max_{i \neq j} \left(\frac{\sigma_i + \sigma_j}{d_m(C_i, C_j)} \right),$$

with i and j being the cluster indexes, σ_x representing the average distance between the elements within cluster x and $d_m(C_i, C_j)$ being the distance between the medoids of the clusters C_i and C_j.

This way, the user gets a visual insight into the clustering and can decide whether an adjustment of the levels of detail is necessary and useful. For instance, a sudden change of the similarity or of the Davies-Boulding index could be a reason for an adjustment.

Step 3: Visualizing the Clusters In the third step, the user can visually explore the hierarchy and the relationships of the shown clusters on increasing levels of detail. At first, the labels from the upmost level of detail are taken and projected onto a two-dimensional plane, which we henceforth call landscape visualization, by using the t-SNE projection technique [35]. We use t-SNE as it is possible to apply it with or without an initial spatial mapping of the data. This will be further elaborated in Section 4.1.

For every level of detail, we create a separate distance matrix that is used by t-SNE. We calculate the cost to traverse the hierarchy tree between all shown clusters and use the results as the distances between the clusters. Once the positions of the clusters have been determined, representative labels can be shown in the landscape view. Details about the projection and positioning of the labels will be given in Section 4.1.

We encoded several key aspects of the data and its structure into the landscape visualization:

Representative of the cluster: Every cluster is represented by a label. Following the recommendation of Elmqvist and Fekete [9], we use the cluster's element with the highest overall occurrence frequency as the cluster label.

Font size for importance: As the font size is perceptually prominent, we use it to indicate a cluster's importance, similar to word clouds. Since clusters usually consist of several elements, we map the accumulated occurrence frequency of the elements onto the font size. We opted to represent the clusters' importance on the currently shown level of detail by normalizing the accumulated frequencies of all shown clusters.

Numbers of elements within a cluster: To give the users an idea of the general distribution of the labels, we added a colored radial background to every label. The size of the radius corresponds to the amount of elements that are contained within the cluster. The color scale is a linear gradient that starts with a light blue in the center and fades out towards the outside. In case the cluster is selected or marked to be relevant, the light blue is replaced by a dark or light green (see Section 4.2.3).

Cluster density: When two or more clusters overlap, their colors add up and a heat map-like visualization is created. This helps to get a better visual impression of the clusters' distribution. The radius of the clusters remains constant when the users zoom in or out of the landscape. Thus, smaller clusters are aggregated into bigger ones.

Position of the labels: As with all projection techniques, the position of the labels follows the Gestalt principle that visual closeness correlates with similarity. More precisely, due to the way t-SNE works, closeness correlates with the likelihood that two elements are related to each other. We incorporated this aspect implicitly by using the similarity as the projection measure in the second step of our approach (see Section 3.2). In order to keep the cognitive load low, it is important to keep the positions of the already visualized labels as stable as possible when changing the level of detail. We elaborate on the specific aspects of this in the following section.

4 Interactive Exploration

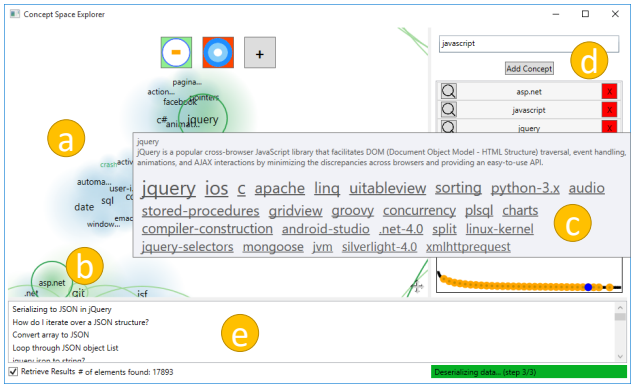

Figure 5: Screenshot of our prototype showing data from the question-and-answer website StackOverflow. The landscape view ⓐ shows the projected level of detail's clusters. Some clusters are selected and highlighted with a dark green background. All selected and related clusters are indicated by a green background color and halos. Some clusters are not visible in the focused viewport, but the halos indicate their position (ⓑ). A tooltip ⓒ shows the content of the focused cluster *jquery*, including its name, description and a word cloud with the most frequent elements contained in that cluster. The selected labels are also shown and highlighted in the search component on the right ⓓ. On the bottom ⓔ, questions are listed that contain at least one element from every selected cluster.

As outlined in Section 2, there are different ways to interact with two-dimensional representations of high-dimensional data. In the following, we describe how to switch between levels of detail and ways to interactively explore the landscape within a given level.

4.1 Switch Between Levels of Detail

To view the data on different levels of detail, the user needs to be able to switch between the chosen levels. However, projection algorithms are not designed to support the iterative projection of a dataset with subsets of increasing levels of detail and do not make

use of previously positioned clusters. However, it is important to support the user's mental map regarding the positions of the clusters in order to minimize the cognitive load during the change of a level of detail. At the same time, it is necessary to visually inform the user about the relation of the newly available subclusters in the context of the already visualized clusters. We achieve this by minimizing the movement of unchanged data as well as animating the movement of new data points from their respective parent cluster's position to their final destination.

At the first level of detail, we do not have any prior data about the previous projection step. Therefore, we apply the standard implementation of t-SNE, which uses a t-student distribution of the projected data as an initial mapping before optimizing the clusters' positions [35]. We use several means to stabilize the subsequent projections, which we will explain in detail in the following:

1. Initial map: Originally designed to pause and resume a projection run of t-SNE in order to show intermediate results, we use the option to 'resume' a t-SNE run with a superset of the data used in the previous projection step. We search for all clusters that split into smaller clusters in the next level of detail. Every unchanged cluster keeps its position and all newly generated clusters inherit the position from the cluster they originate from. The resulting map is used as the initial map of t-SNE, replacing the t-student distribution of the clusters that is used by default and therefore stabilizing the clusters' positions.

2. Perplexity: We also adapt the perplexity parameter used by t-SNE. The perplexity can be interpreted as a factor that controls the size of the neighborhood considered in the high-dimensional space during the projection of a data point. Typically, this is a constant value which must be adapted, if the results are not satisfying. Van der Maaten and Hinton [35] recommend a value between five and 50 for this parameter. Usually, this value is robust and small differences do not heavily impact the visualization. However, in the case of a modified initial mapping, a too high or too low perplexity value can lead to visual artifacts. In case the perplexity is too high, the visualization will look like all clusters center around one point, because every cluster tries to optimize with regard to all other clusters. If the perplexity is too low, the clusters will not move at all because they only consider themselves as relevant and therefore all newly added clusters overlap at their parent's position. To keep the user from following a trial and error approach to find a proper value, we decided to dynamically approximate the perplexity value depending on the number of shown clusters. Our perplexity function $p(x) = 6.929 \cdot x^{0.252710}$ is designed to initially increase fast in order to ensure that clusters in the first few levels of detail already consider some of their neighboring clusters. The more clusters are shown the lower is the increase of the perplexity compared to the previous level of detail. The values are based on the results we achieved with the datasets we describe in the use cases.

We compared our function to static perplexities on different levels of detail. We used the Kullback-Leibler divergence as a performance measure, since the authors of the t-SNE algorithm proposed it as a good statistical quality measure. The results based on the dataset of use case 2 are shown in Figure 6. It becomes clear that our function-based perplexity performs slightly better than most static perplexities. When we tested a static perplexity of $p = 15$, we noticed a lot more overdraw of the labels and concluded that lower perplexities will amplify this problem even more.

3. Scaling of target projection space: Once the clusters are projected, their positions are min-max normalized. To prevent a shift of all clusters due to outliers, we first calculate the clusters' geometric center. Afterwards, we normalize all clusters, but we discard the 10% of the clusters that are the farthest away from the center. Then, we rescale the target projection space. The scaling factor is a root function that depends on the amount of visible clusters.

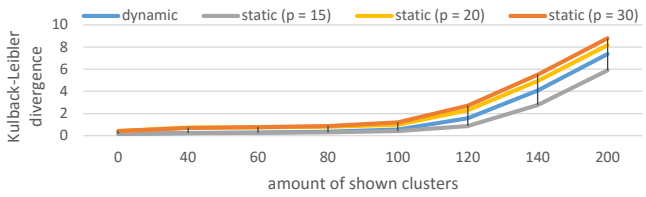

Figure 6: Comparison of our dynamic against static perplexities

4.2 Explore Cluster Relationships

In case the users want to explore a given level of detail, there are two scenarios: either they already have an initial idea what they are looking for, or they want to get an overview of the chosen level of detail without any premises. In both cases, the next step is to decide on interesting clusters to inspect. In the following, we will describe the means we provide in order to accomplish this.

4.2.1 Search for Specific Clusters and Elements

We provide the user with a search box that provides an autocomplete feature, suggesting every label contained within the dataset. Once an element has been found, it can be added to the list below the search box (Figure 5ⓓ). The selections made in the search list are linked to the landscape view (Figure 5ⓐ). Sometimes the searched element is not visible in the landscape view because it is hidden within a cluster. In this case, the cluster that contains the element will be selected. It is also possible to focus on the element or cluster that contains the element in the center of the view.

4.2.2 Freely Explore the Landscape View

The landscape view supports zooming and panning to let the user freely explore the visualized level of detail. As described in Section 3.2, the size of the background color of the clusters is independent of the visual zoom. This way, the clusters are visually aggregated when the user zooms out, as the cluster backgrounds' size increases compared to the clusters' labels. This helps the user to distinguish between areas with a higher cluster density and regions that are sparser.

When the users decide on one or more interesting clusters, they can select them in the landscape view. The selected elements will then be added to the aforementioned search list (Figure 5ⓓ).

4.2.3 Navigation Aids in the Landscape View

Once a set of clusters and/or elements is selected, the user may be interested in other clusters related to the selected ones. However, this information may not be available directly, as some clusters that are related in the high-dimensional space may not be close to each other in the low-dimensional space or vice versa. We differentiate between two kinds of relatedness: *global* and *local*. The global relatedness is depicted by the spatial arrangement of the clusters, wherein closer clusters are more likely to be related than clusters that are farther apart. The local relatedness is available for selected clusters and shows which clusters are similar to those clusters. To measure the local relatedness, we calculate the average similarity s_{avg} between the set of selected clusters and the cluster they are compared with by calculating

$$s_{avg}(C_{other}) = \frac{1}{n}\sum_{i=0}^{n-1}\left(\frac{1}{m}\cdot\sum_{j=0}^{m-1}\frac{1}{o}\cdot\left(\sum_{k=0}^{o-1} sim(C_{i,j}, C_{other,k})\right)\right),$$

wherein i is the index of the cluster within the cluster set, j is the index of the compared element within the cluster and k is the index of the element within the cluster that is compared with the set of selected clusters.

As the locally related clusters may lie spatially apart from the selected clusters, we implemented several measures to compensate for this information loss.

Halos: To support users in finding clusters that are related to the selected clusters, we provide them with a navigation aid introduced by Baudisch and Rosenholtz [2]. Therewith, the users get a visual notion of the selected and other related clusters' position by drawing a circle, which is called *halo* in this concept, around these clusters. In case the cluster is outside of the visible area, the halo's radius gets expanded, so it barely stays within the view. The curvature of the Halo fragment indicates direction and distance of the corresponding clusters. An example of the visual indication provided by halos is shown in Figure 5ⓑ.

Color coding: To make selected, related and other clusters distinguishable, we color the backgrounds and halos of selected clusters with a noticeable shade of green. Analogously, we colored the backgrounds and halos of related clusters with a shade of light green. The other clusters' background is drawn in a light blue. We decided to use color coding over approaches such as isolines or glyphs because we want to give the user an impression of the clusters' different statuses, whilst indicating the uncertainty of the clusters' positions due to the projection's information loss.

Darts View: Furthermore, we added the Darts View Visualization introduced by Herr et al. [19]. By using a darts game metaphor, users can quickly spot related clusters, as the selected clusters are shown in the middle of the darts view and the related clusters are arranged around them.

The Figures 5ⓐ and 8 depict how these aspects look like in our implementation of the concept.

4.3 Analyzing a Cluster's Content

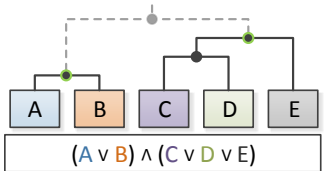

Figure 7: For each selected cluster (here indicated with a green circle), the leaf nodes are retrieved. Then, data is requested from the dataset, which contains at least one of the leaf nodes from every selected cluster.

Once the users found a relevant cluster, they may be interested in further information about it. We provide several means to inspect the content of a cluster. First, the user may request additional information about the cluster by looking at the cluster's tooltip. In case only the name of the elements is available, the tooltip shows the name of the representative label as well as a word cloud. The labels shown in the cloud belong to the elements contained in the cluster. They are ordered by their weight ($\hat{=}$ occurrence frequency), which is also encoded into the words' font size. In case the elements also contain a textual description, the description of the representative label is shown at the top. The word cloud contains the terms that occur most often in the descriptions of all elements. An exemplary depiction of the tooltip is shown in Figure 5ⓒ.

Second, the user can select one or more clusters. When doing so, our approach retrieves individual documents which contain at least one term from each selected cluster. A schematic demonstration of such a request is depicted in Figure 7. Here, the two highlighted clusters were selected. As the elements *A* and *B* are contained within the first cluster, only one of them has to be contained within the resulting document. By selecting an individual document, further details can be retrieved.

5 Use Cases

In the following, we will illustrate the applicability and usefulness of the approach with two scenarios using real world data.

5.1 Use Case 1: Analysis of StackOverflow

In the first use case, we assume the role of the head of a newly founded department of a software company that used to develop server-sided software solutions. Our new department is tasked to supplement a client-sided solution to the software suite. The company already supplied us with some of its software developers, but we still need to recruit a developer that is well versed with web development. As we only possess basic knowledge of this field, we need to take an explorative approach in order to gain knowledge, which aspects are important. To create a list of key skills for our recruit's profile, we will use our approach to explore the question-and-answer website *StackOverflow*'s[1] tags and their relations. Since StackOverflow contains data that comprises more than ten million questions, we limit our inspection to tags that have been assigned to at least 5000 questions. The similarity between the tags is based on Jaccard coefficient [26, p. 56].

We decide to deselect the first four proposed levels of detail, because we want to see some detailed information at the top level already. We know that our company's server-sided software returns JSON formatted answers and we also know that JavaScript is a common solution for web clients. Therefore, we select the clusters containing *JSON* and *JavaScript*. To see if our interview candidates have some background knowledge, we note some of the most often viewed questions in StackOverflow that contain elements from the JSON and JavaScript clusters. After selecting the clusters, the tool marks the cluster *jQuery* to be relevant. Once reading the tag's description in the tooltip and looking at its contents, we decide to add it to our profile and select the cluster (shown in Figure 5ⓒ). We write down questions such as 'How to iterate over a JSON structure?' for the interview of candidates for our team. At last, we freely explore the space around our selected clusters and notice the cluster *d3.js*. After looking at the cluster's description, we decide to add *d3.js* to our profile, as a developer with knowledge in web-based visualization may be useful in later software development stages.

By now, we have a profile that we can use to search for a software developer that has knowledge in key technologies used in web development, which fits our companies existing technologies.

5.2 Use Case 2: Patent Analysis

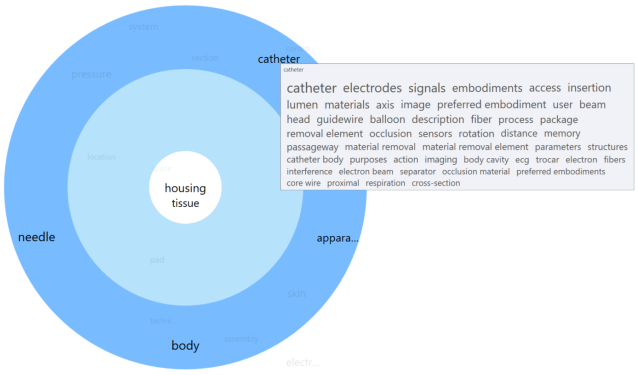

Figure 8: Depiction of the concepts related to the clusters *adhesive* (contained in *housing*) and *tissue*. There are five concepts that are strongly related to those clusters. One is catheter, which makes sense, as a catheter may be used to apply the adhesive.

In the second use case, we take the point of view of an intellectual property analyst at a large company that produces and sells medical apparatuses. The company plans to develop a new product to apply an adhesive to open tissue during a surgery and wants to avoid conflicts with existing patents.

[1] http://stackoverflow.com/

The analyst's task is to limit the number of patents that have to be inspected individually to a reasonable amount. At first, the analyst limits the number of patents by searching for a code from the International Patent Classification (IPC), which is specific for medical surgery and diagnosis. Also, the patents need to contain the keyword *glue* or *adhesive* in their description. This results in a patent set of about 300 patents, which is too much to analyze manually. Thus, the patents are automatically processed and concepts are extracted. The similarity between concepts is calculated by measuring the concepts' co-occurrence within sentences and comparing them using a cosine similarity. These concepts will then be loaded into our prototypical implementation to limit the patents even further.

The analyst may choose to change the initially proposed levels of detail, but in this case we assume that the expert will leave the levels as proposed. Once the first level is projected, the analyst searches for the clusters that contain the concepts *adhesive* (which is hidden inside the cluster *housing*) and *tissue*. By doing so, the analyst notices that the concept *catheter* is marked to be relevant to the selection. This becomes even clearer by switching to the darts view, which is shown in Figure 8. As it makes sense to apply an adhesive with a catheter, the analyst adds it to the set of selected clusters.

When looking at the resulting patents, the analyst is not satisfied with the quality of the results. Therefore, he or she adds the cluster containing the concept *apparatus* to the selected concept list and increases the level of detail several times. By doing so, the selected clusters get more specific as dissimilar elements split apart which results in a more specific patent request. Hence, the results of the leftover patents becomes better. By increasing the levels about five more times, the clusters become more specific, as less similar clusters split apart. Therefore, the analyst is left with 27 patents, which is a reasonable amount for a manual analysis.

6 Discussion

When analyzing a new dataset, the user is confronted with a chicken and egg problem: on the one hand, the zoom levels are supposed to help users in understanding the possible relations within the dataset by only showing information on a very coarse level. On the other hand, without knowledge about what levels of detail may be interesting, it is very hard to decide how many levels of detail should be shown and what information granularity they should have. We address this dilemma by proposing the user a preset number of levels of detail that can be used as a starting point for the exploration. Also, we show the user the error index of the clustering to support an easier manipulation of the levels. However, the shown information is of little help when the user is interested in which clusters are actually merged at a specific step. Even in case the user had that knowledge, it is still impossible to infer any further knowledge about the previous or the next cluster fusion step. Moreover, the user only gets an abstract measure of the quality of the clustering, which is hard to comprehend. This problem may worsen when the binary tree of the clustering algorithm is simplified, for example, by using Bayesian Rose Trees [3]. It is hard for a user to comprehend how many clusters will be shown in the next level of depth of the tree, as every step may imply multiple aggregation steps. As all of the mentioned aspects are important in choosing levels of detail that meet the user's needs and expectations, we will look into possible solutions in the future. For example, instead of proposing a constant step size within the hierarchy, it may be possible to further analyze the hierarchy and dataset to make a more appropriate recommendation for relevant levels of detail.

Like all dimensionality reduction approaches, our approach suffers from the curse of dimensionality. The more elements are contained within the matrix and the more they are related, the less will be their measurable distance. This causes problems in the projection step, as it will not be able to distinguish which labels should be clustered together anymore. This correlates with the problem to pick a proper perplexity value in our approach. When the value is too low or too high, the visualization will not be able to represent the data's relation anymore.

We tried to design the approach in a way that an analysis run can be done as smoothly as possible, even as more and more clusters are shown. As our approach is based on the original implementation of t-SNE, the time complexity is $O(n^2)$, where n is the number of shown clusters. This may cause a performance issue when many labels have to be projected. We measured how the increase of one level of detail in use case 2's dataset takes during the actual analysis (after preprocessing) on a computer with an Intel Core i7-4700MQ processing unit. It took 0.5 seconds for 40 clusters and 4.0 seconds for 480 clusters from the point of the user interaction to the point when the labels started to move to their new positions. Such issues can be tackled, for instance, by calculating the next level of detail in a background thread while the user explores the actual level of detail, by using a better performing implementation of t-SNE, or by calculating the projection information of the visible portions of the screen first and compute other needed information on demand. It is important to note that the latter will implicitly impact the quality of the visualization. Similar clusters should be projected close to each other, even if they originate from different parent clusters. When one of these parents is not visible, this information will be lost.

Moreover, the abundance of clusters on a high level of detail causes an increased information loss during the projection onto a two-dimensional space. Labels that are intuitively close to each other may be far apart. Also, our approach to place new clusters at the position of their parent clusters may bias the projection's result. However, we think this is tolerable as the similarity matrix provided to the projection contains all visible clusters and the original t-student distribution does not use any previously gained information. This is a general challenge and can only be slightly alleviated by indicating the position of the labels that are close in the high-dimensional space. We plan to extend our approach's variety of navigation aids by adding more visualizations to support in-depth analysis of the selected and related clusters.

As our visualization shows the mode of a cluster, it seems plausible to use those modes for clustering. However, we decided against such a linkage, because it is susceptible to outliers as it does not consider other elements in a cluster, similar to single linkage. Also, it implies that an element often occurring in the complete dataset is also important within a cluster, which is problematic.

Furthermore, the overlap of some clusters still poses an issue when many very similar clusters are projected on the landscape. Our approach tries to find a trade-off between the increasing white space produced when scaling the target space to prevent overlapping and a slight overlapping where the labels are still readable. We decided not to use an overlap removal after the projection step, as we wanted to keep the spatial distance of the clusters as intact as possible. However, we think that using more features to scale the projections target space may further help to prevent overlapping.

7 Conclusion

In this paper, we presented a visualization approach for the hierarchical exploration of labeled data. It reduces the problem of visual clutter and information overload that can quickly occur in the two-dimensional visualization of large amounts of labels. Overall, it consists of three steps: 1) the data is hierarchically clustered, 2) distinct levels of detail are defined, and 3) the levels of detail are projected on a layered landscape visualization.

In contrast to related work, our approach enables the users to explore the labeled clusters in two different ways: 1) by analyzing an individual level of detail to understand the clusters' child-child relation, and 2) by switching between different levels of detail to understand the hierarchical structure and composition of the clusters. This allows users to choose the level of abstraction that is most use-

ful for their analysis goals, and to explore a large amount of labeled data. We tailored the t-SNE projection algorithm to make use of the hierarchical structure of the data and to reuse already computed positional information. Animations help the users to understand the transitions between different levels of detail. Interaction techniques support the exploration on a given level of detail and ease the identification of related clusters in the visualization.

We demonstrated the applicability and usefulness of the approach with two use cases. In the first one, we showed the exploration aspect of the approach by analyzing the tag structure of the question-and-answer website StackOverflow to extend our understanding about a domain. In the second use case, we applied the approach to the field of patent analysis to support an analyst in filtering a set of patents.

Key challenges for future work include the appropriate labeling of merged clusters and the automatic detection of useful levels of detail. Furthermore, improved methods to even better reflect the global and local relatedness of clusters in the spatial layout need to be examined.

8 Acknowledgments

This work has been supported by the EU funded project iPatDoc (grant no. 606163).

References

[1] J. Alsakran, Y. Chen, D. Luo, Y. Zhao, J. Yang, W. Dou, and S. Liu. Real-time visualization of streaming text with a force-based dynamic system. *IEEE Comput. Graph. Appl.*, 32(1):34–45, 2012.

[2] P. Baudisch and R. Rosenholtz. Halo: A technique for visualizing off-screen objects. In *Eurograph. Ass. SIGCHI Conf. on Human Factors in Computing Systems*, pages 481–488. ACM, 2003.

[3] C. Blundell, Y. W. Teh, and K. A. Heller. Bayesian rose trees. In *Proc. 26th Conf. Uncertainty in Artificial Intelligence*, pages 65–72. AUAI Press, 2010.

[4] M. Brehmer, S. Ingram, J. Stray, and T. Munzner. Overview: The design, adoption, and analysis of a visual document mining tool for investigative journalists. *IEEE Trans. Vis. Comput. Graph.*, 20(12):2271–2280, 2014.

[5] M. Burch, S. Lohmann, D. Pompe, and D. Weiskopf. Prefix Tag Clouds. In *Proc. Int. Conf. Information Visualisation*, pages 45–50, 2013.

[6] Y.-X. Chen, R. Santamaría, A. Butz, and R. Therón. TagClusters: Semantic aggregation of collaborative tags beyond TagClouds. In *10th Int. Symp. Smart Graphics*, pages 56–67. Springer, 2009.

[7] D. L. Davies and D. W. Bouldin. A cluster separation measure. *IEEE Trans. Pattern Anal. Mach. Intell.*, 1(2):224–227, 1979.

[8] W. Dou, L. Yu, X. Wang, Z. Ma, and W. Ribarsky. HierarchicalTopics: Visually exploring large text collections using topic hierarchies. *IEEE Trans. Vis. Comput. Graph.*, 19(12):2002–2011, 2013.

[9] N. Elmqvist and J.-D. Fekete. Hierarchical aggregation for information visualization: Overview, techniques, and design guidelines. *IEEE Trans. Vis. Comput. Graph.*, 16(3):439–454, 2010.

[10] M. Endo, M. Ueno, and T. Tanabe. A clustering method using hierarchical self-organizing maps. *J. VLSI Signal Process. Syst.*, 32(1-2):105–118, 2002.

[11] D. Fried and S. G. Kobourov. Maps of computer science. In *IEEE Pacific Visualization Symposium*, pages 113–120. IEEE, 2014.

[12] T. M. Fruchterman and E. M. Reingold. Graph drawing by force-directed placement. *Softw. Pract. Exper.*, 21(11):1129–1164, 1991.

[13] E. R. Gansner and S. C. North. Improved force-directed layouts. In *Proc. 6th Int. Symp. Graph Drawing*, pages 364–373. Springer, 1998.

[14] F. J. Garca-Fernndez, M. Verleysen, J. A. Lee, and I. Daz. Stability Comparison of Dimensionality Reduction Techniques Attending to Data and Parameter Variations. In *EuroVis Workshop Visual Analytics using Multidimensional Projections*. Eurograph. Ass., 2013.

[15] K. S. Hasan and V. Ng. Automatic keyphrase extraction: A survey of the state of the art. In *Proc. 52nd Annual Meeting of the Association for Computational Linguistics*, pages 1262–1273. ACL, 2014.

[16] Y. Hassan-Montero and V. Herrero-Solana. Improving tag-clouds as visual information retrieval interfaces. In *Int. Conf. Multidisciplinary Information Sciences and Technologies*, pages 25–28, 2006.

[17] T. Hastie, R. Tibshirani, J. Friedman, and J. Franklin. The elements of statistical learning: data mining, inference and prediction. *Math. intell.*, 27(2):83–85, 2005.

[18] F. Heimerl, S. Lohmann, S. Lange, and T. Ertl. Word cloud explorer: Text analytics based on word clouds. In *47th Hawaii Int. Conf. System Sciences*, pages 1833–1842. IEEE, 2014.

[19] D. Herr, Q. Han, S. Lohmann, S. Brügmann, and T. Ertl. Visual exploration of patent collections with ipc clouds. In *Proc. 1st Int. Workshop Patent Mining and Its Applications*, volume 1292. CEUR-WS, 2014.

[20] N. Heulot, M. Aupetit, and J.-D. Fekete. ProxiLens: Interactive Exploration of High-Dimensional Data using Projections. In *EuroVis Workshop Visual Analytics using Multidimensional Projections*. Eurograph. Ass., 2013.

[21] O. Kaser and D. Lemire. Tag-cloud drawing: Algorithms for cloud visualization. In *Workshop Tagging and Metadata for Social Information Organization*, 2007.

[22] T. Kohonen. The self-organizing map. *Proc. IEEE*, 78(9):1464–1480, 1990.

[23] J. B. Kruskal and M. Wish. *Multidimensional scaling*. Sage, 1978.

[24] S. Liu, X. Wang, J. Chen, J. Zhu, and B. Guo. TopicPanorama: A full picture of relevant topics. In *IEEE Conf. on Visual Analytics Science and Technology (VAST)*, pages 183–192, 2014.

[25] S. Lohmann, J. Ziegler, and L. Tetzlaff. Comparison of tag cloud layouts: Task-related performance and visual exploration. In *12th IFIP TC 13 Int. Conf. Human-Computer Interaction*, pages 392–404. Springer, 2009.

[26] C. D. Manning, P. Raghavan, H. Schütze, et al. *Introduction to information retrieval*, volume 1. Cambridge University Press, 2008.

[27] F. V. Paulovich, L. G. Nonato, R. Minghim, and H. Levkowitz. Least square projection: A fast high-precision multidimensional projection technique and its application to document mapping. *IEEE Trans. Vis. Comput. Graph.*, 14(3):564–575, 2008.

[28] F. V. Paulovich, F. M. B. Toledo, G. P. Telles, R. Minghim, and L. G. Nonato. Semantic wordification of document collections. *Comput. Graph. Forum*, 31(3):1145–1153, 2012.

[29] C. Seifert, B. Kump, W. Kienreich, G. Granitzer, and M. Granitzer. On the beauty and usability of tag clouds. In *12th Int. Conf. Information Visualisation*, pages 17–25, 2008.

[30] B. Shneiderman. Tree visualization with tree-maps: 2-d space-filling approach. *ACM Trans. Graph.*, 11(1):92–99, 1992.

[31] B. Shneiderman. The eyes have it: A task by data type taxonomy for information visualizations. In *Proc. IEEE Symp. Visual Languages*, pages 336–343. IEEE, 1996.

[32] P. Suganthan. Hierarchical overlapped som's for pattern classification. *IEEE Trans. Neural Netw.*, 10(1):193–196, 1999.

[33] D. Thom and T. Ertl. TreeQueST: A treemap-based query sandbox for microdocument retrieval. In *48th Hawaii Int. Conf. System Sciences*, pages 1714–1723. IEEE, 2015.

[34] J. J. Thomas, P. J. Cowley, O. Kuchar, L. T. Nowell, J. Thompson, and P. C. Wong. Discovering knowledge through visual analysis. *J. Univers. Comput. Sci.*, 7(6):517–529, 2001.

[35] L. Van der Maaten and G. Hinton. Visualizing data using t-SNE. *J. Mach. Learn. Res.*, 9(85):2579–2605, 2008.

[36] L. J. van der Maaten, E. O. Postma, and H. J. van den Herik. Dimensionality reduction: A comparative review. *J. Mach. Learn. Res.*, 10(1-41):66–71, 2009.

[37] J. A. Wise, J. J. Thomas, K. Pennock, D. Lantrip, M. Pottier, A. Schur, and V. Crow. Visualizing the non-visual: Spatial analysis and interaction with information from text documents. In *Proc. IEEE Symp. Information Visualization*, pages 51–58. IEEE, 1995.

[38] S. Wold, K. Esbensen, and P. Geladi. Principal component analysis. *Chemometrics Intell. Lab. Syst.*, 2(1):37–52, 1987.

[39] P. C. Wong, B. Hetzler, C. Posse, M. Whiting, S. Havre, N. Cramer, A. Shah, M. Singhal, A. Turner, and J. Thomas. IN-SPIRE infovis 2004 contest entry. In *Proc. IEEE Symp. Information Visualization*. IEEE, 2004.