Prefix Tag Clouds

Michael Burch, Steffen Lohmann, Daniel Pompe, Daniel Weiskopf
VIS/VISUS, University of Stuttgart, Germany
{*michael.burch,daniel.weiskopf*}*@visus.uni-stuttgart.de*
steffen.lohmann@vis.uni-stuttgart.de

***Abstract*—Tag clouds are a popular way to visually represent word frequencies. However, one major limitation is that they do not relate different word forms but treat every form as an individual tag. This results not only in a non-efficient use of screen space but, in particular, leaves the viewer with no indication whether there are other forms of a word or not. To overcome this limitation, we introduce prefix tag clouds: a visualization technique that uses a prefix tree to group different word forms and visualizes the subtrees as tag cloud. The grouping is emphasized by color, while the relative frequencies of the word forms are indicated by font size. A circular tag cloud layout supports the quick identification of the most frequent words and word forms. We show the usefulness of the approach for a large dataset of paper titles from the computer science bibliography DBLP.**

***Keywords*-tag cloud; prefix tree; word forms; circular layout; word cloud**

I. Introduction

Having their roots "outside the world of computers" [1], *tag clouds* (also known as *word clouds*) became popular in the context of community-oriented websites that use tagging as indexing method [2], [3]. Meanwhile, they have crystallized themselves as a core technique of information visualization that is applied in many different contexts [1], [2]. One popular application area is text summarization, where tag clouds are typically used to depict the words that occur most often in texts. The font sizes of the tags indicate the word frequencies, i.e. the larger a tag the more often it occurs in the text [4].

A major limitation of tag clouds is that they treat different forms of the same word as individual tags. Typical examples are inflections (e.g. singular and plural), nominalizations (e.g. the gerund "-ing"), or spelling differences (e.g. British vs. American English). Such variations of words either also appear in the tag cloud or are not shown at all if not frequent enough. In the first case, they take up screen space that could better be used to display other information or additional tags. In the second case, the viewer has no indication whether there are other forms of a word in the text or not.

Especially the latter is problematic: Assume, for example, an information retrieval context where a user is interested in all documents related to the topic of "visualization". Searching for this term would likely not return all relevant documents, as the term would not have been used by all authors. However, if other word forms like "visualisation" or "visually" are less frequent in the documents, they might not be shown in the tag cloud. Hence, there would be no hint that would tell the user what else to try as search term.

To overcome this limitation, we have developed an extension of the tag cloud visualization that we call *prefix tag cloud*. It creates a prefix tree that groups different word forms and visualizes the subtrees as tag cloud. Color is used to emphasize the grouping, while the relative frequencies of the word forms are indicated by font size, as it is common in tag clouds. The grouped word forms are arranged in a circular tag cloud layout that supports the quick identification of the most frequent words and word forms.

II. Related Work

Several extensions to the basic tag cloud visualization have been presented in the last couple of years. Many approaches address layout issues of tag clouds, such as large white spaces or the restriction to specific boundaries. As just one example, Kaser and Lemire tackle the problem of wasted and unbalanced space and present models and algorithms to improve the display of HTML-based tag clouds [5]. They consider tag relationships and use slicing trees, nested tables, and rectangle packing to optimize the distribution of space in the clouds. Seifert et al. [6] also propose algorithms for space optimization in tag clouds. The resulting layouts are more compact and clear and can even feature shaped boundaries for tag clouds. Advanced layouts are also provided by popular tag cloud generators, such as Wordle [4], [7], Tagxedo [8], or Tagul [9].

While there are several works on layout issues of tag clouds, only few address the grouping of different word forms. One related approach is that of Cui et al. who use a porter stemmer to collapse words with the same stem and only include the most frequent word form in the tag cloud [10]. A similar approach is taken by Dörk et al. [11]. However, the viewer gets no idea which word forms have been collapsed, as this information is not shown in the tag clouds. Also related to our idea is the work on Word Trees by Wattenberg and Viégas [12]. They use a suffix tree to present search terms in their textual context. Instead of single words, they analyze complete sentences and visualize the sentence structure as tree. It is the only work we found that uses a data structure similar to prefix trees for visualization purposes. It is conceptually related to our approach but

To appear in:
Proceedings of the 17th International Conference on Information Visualisation (IV 2013)
Los Alamitos, CA, USA: IEEE, 2013.

serves a different purpose by displaying keywords in context instead of grouping word forms in tag clouds.

There are also visualization techniques that combine tag clouds and trees. For instance, Tree Clouds arrange tags on a tree to visually depict their semantic relatedness [13]. Other approaches, such as clustered tag clouds [2], [5], [14], do not explicitly express semantic relatedness of tags by visual links but implicitly by spatial proximity. Parallel Tag Clouds [15] combine the ideas of tag clouds and parallel coordinates to allow for a direct comparison of changes in word use. In contrast, SparkClouds add sparklines (i.e. simplified line graphs) to the tags to depict changes in word use over time [16]. Yet other works propose geographical variants of tag clouds, using either virtual tag landscapes [17] or real geographical space [18]. Finally, there are also 3D variants of tag clouds, such as WP-Cumulus [19] that provides a rotating, three-dimensional sphere of tags. All these variations are compatible with, and complementary to, prefix tag clouds, so that they can be integrated with our approach.

III. Creation of Prefix Tag Clouds

The process of creating prefix tag clouds consists of three main components: First, a prefix tree is generated from a set of tags. This requires us to order the tags lexicographically and to compute a prefix hierarchy from the resulting list. In a second step, the prefix subtrees are rendered as node-link diagrams. The font sizes of the prefixes in the diagrams are scaled according to the tag frequencies. Finally, the tag cloud is composed from the subtrees and placed in a given drawing area. We use a circular tag cloud layout that makes efficient use of the available screen space. In the following, we will detail these three components of the visualization process.

A. Prefix Tree Generation

Initial input is a set of tags $T := \{t_1, \ldots, t_n\}$ with individual tags t_i, $1 \leq i \leq n$. The tags are composed of a finite sequence of characters from the alphabet $\Sigma := \{\sigma_1, \ldots, \sigma_m\}$. Since $t_i \in \Sigma^+$, each tag t_i contains at least one character and is not an empty string.

In addition, each tag is associated with a quantitative value expressing its occurrence frequency. This mapping is given by the function $f_{freq} : T \longrightarrow \mathbb{N}$. Likewise, we define T_{freq} to denote that the tag set T is ordered by frequency, while we use T_{lex} if it is ordered lexicographically.

The prefix tree P is generated from the set of tags T with the function $f_{tree} : T \longrightarrow P$. Algorithm 1 provides the pseudo code for that function which consists of the following steps:

- **Adding the empty string λ to T:** It serves as prefix[1] for all tags $t_i \in T$ that do not have another tag $t_j \in T$ as prefix.

[1]Note that the term *prefix* is not used in its linguistic sense but in its meaning related to data structures in this work.

Algorithm 1 Prefix Tree Generation

```
PrefixTree($T$):
  // $T$: set of tags
  $T := T.add(\lambda)$;                  // add empty string $\lambda$ to $T$
  $T_{lex} := lex\_order(T)$;          // order $T$ lexicographically
  $n := size(T_{lex})$;
  $V := T_{lex}$;                       // vertices of prefix tree
  $E := \emptyset$;                     // edges of prefix tree
  for $i := n$ to 1 do
    $j := i - 1$;
    while $(!(t_j \prec_{pref} t_i) \wedge (j \geq 1))$ do
      $j--$;
    end while
    $v_i := t_i - t_j$;                 // subtract tag $t_j$ from tag $t_i$
    $E := E \cup (v_i, v_j)$;           // add edge to prefix tree
  end for
  return $P = (V, E)$;                  // prefix tree $P$
```

- **Ordering T lexicographically:** Fast sorting algorithms such as merge sort can do this in $\mathcal{O}(n \log n)$ time. As a result, we obtain the lexicographically ordered list T_{lex} with the empty string λ as first element.
- **Generating the prefix tree from T_{lex}:** This step processes the entire list T_{lex} once and checks if tag $t_j \in T_{lex}$ is a prefix of tag $t_i \in T_{lex}$, where $j < i$. If true, it subtracts t_j from t_i and saves the resulting string (i.e. the prefix) as vertex $v_i \in V$. In addition, it creates an edge between v_i and $v_j \in V$. Since it uses two nested loops, one being conditional, it has $\mathcal{O}(n^2)$ time complexity in worst case and $\mathcal{O}(n)$ in best.

The result of Algorithm 1 is the prefix tree $P := (V, E)$ for T, i.e. a graph where the vertices V represent the prefixes and the edges $E \subset V \times V$ describe the hierarchical structure of the prefix tree. The root node $v_{root} \in V$ of the tree is the empty string λ.

Figure 1 illustrates the generation of the prefix tree P_x for a sample list of tags T_{lex_x}. The list contains a small number of lexicographically ordered tags starting with the letter "v". The occurrence frequencies f_{freq} of the tags are also depicted, as they will later be used to scale the font sizes in the prefix subtrees. Figure 1b depicts the hierarchical structure of the prefix tree, while Figure 1c shows the actual prefix tree as it results from Algorithm 1. The tree is displayed in a left-to-right orientation that is later also used to display the subtrees in the prefix tag cloud.

The mapping between prefixes and tags is described by the bijective function $f_{map} : V \longrightarrow T$, i.e. each prefix $v_i \in V$ is associated with exactly one tag $t_i \in T$ and vice versa. In the example of Figure 1, it means that the prefix "visual" is associated with the tag "visual", while the prefix "ization" is mapped to the tag "visualization". Note that the ending

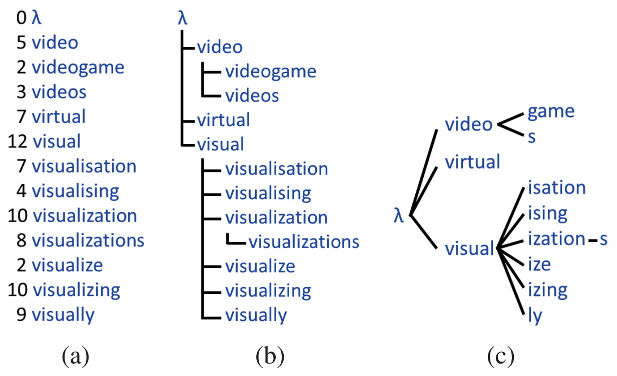

Figure 1. Transforming a list of tags into a prefix tree: (a) Lexicographically ordered list of tags and their frequencies. (b) Node-link diagram illustrating the prefix tree structure. Tags are shown in full length. (c) Node-link diagram of the vertically aligned prefix tree. Tags are split into prefixes.

"s" following the prefix "ization" is also called *prefix* in our case. It represents the tag "visualizations" and is treated like the other prefixes.

B. Subtree Rendering

Before we generate the prefix tag cloud, we split the prefix tree P at its root node v_{root} into a set of subtrees $P' := (V', E')$ with $V' \subset V$ and $E' \subset E$. Each subtree P'_i represents either a group of word forms (if $|V'_i| > 1$) or a single word (if $|V'_i| = 1$). The latter is the case for tags that have v_{root} as prefix and no child node, such as the word "virtual" in Figure 1.

The subtrees of P' are visualized as node-link diagrams in a left-to-right orientation. Parent nodes are placed in the vertical center and to the left of their child nodes, as it was sketched for the whole prefix tree in Figure 1c. The diagrams have different background colors so that they can be more easily distinguished. However, color is not necessary to read and understand the visualization.

The font sizes of the prefixes in the subtrees are scaled according to the frequency values (given by f_{freq}) of the associated tags (given by f_{map}). Since a linear change of the font size has a roughly quadratic effect on the text area, we use the square root of the frequency values for scaling. Using a linear scale instead would overstate the larger tags, while a logarithmic scale would understate them. There may be cases where such scales are more suitable, even though they increase the "lie factor" [20] of the visualization. For instance, a logarithmic scale is often used if the frequency values of the tags tend to follow a power law distribution, as in many folksonomies [3].

Figure 2 shows two alternative renderings of subtree $P'_{visual} \subset P_x$ that was already depicted in Figure 1c: One visualizes the tags $t_i \in T_x$ and the other the prefixes $v_i \in V_x$. As the latter diagram is less redundant and more space-efficient, we use it for the prefix tag clouds.

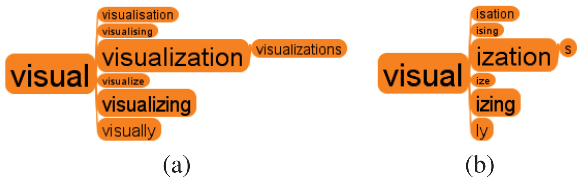

Figure 2. Alternative renderings of the same prefix subtree: (a) showing the complete tags, (b) showing only the prefixes.

C. Tag Cloud Generation

The subtrees are visualized in a circular tag cloud layout with the most frequent tags in the center and tags with decreasing frequencies towards the boundary. According to Lohmann et al. [2], this layout supports particularly well the identification of popular tags, which we consider a key task related to prefix tag clouds.

We create the circular layout with a simple yet effective algorithm that is similar to the one presented in [4]. It places the subtrees of P' along a spiral path, as sketched in Figure 3.[2]

First, the drawing area is defined, which can, in principle, be of any shape. As graphical user interfaces are usually based on 2D grid layouts, a rectangle may be the 'shape of choice' in most cases. Drawing starts with the subtree that contains the tag with the highest frequency value, i.e. tag t_1 from the descending ordered list of tags T_{freq}. This first subtree is placed in the center of the drawing area. In the example of Figure 3, it consists of the tags "system" and "systems", i.e. the diagram shows the prefixes "system" and "s" accordingly.

Subsequently, subtrees with decreasing frequencies are placed along the spiral path. This is done by traversing T_{freq} and rendering the associated subtrees as described above. As long as a subtree would intersect with any previously placed subtrees in the drawing, it is moved further along the spiral path. It is also moved if it would be placed outside the drawing area. If there is not enough free space left to place a subtree in the drawing area, it is continued with the next subtrees until some predefined threshold or until the last element of T_{freq} is reached. Rendered subtrees are marked as rendered to avoid multiple placement of the same subtree.

If we would strictly adhere to the circular layout as described in [2], we would not be allowed to place tags with small font sizes close to the center of the cloud. However, we would waste a lot of space in such a strict layout, as we do not only place tags of rectangular shape but also subtrees with more complex shapes. Hence, we decided to fill free space between larger subtrees with smaller ones. We do this by starting the spiral placement for each subtree in the center of the drawing area so that smaller subtrees

[2]Note that the actual distance between turns of the spiral must be much smaller than sketched to get results like in this paper.

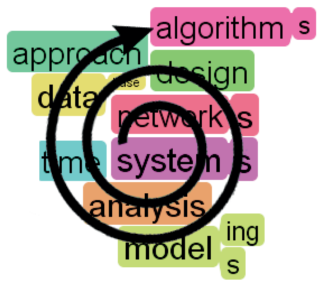

Figure 3. Spiral placement of the prefix subtrees to create the circular tag cloud layout.

are rendered whenever there is enough space, also between already drawn subtrees.

Subtree drawing continues until one of three conditions is met: 1) all subtrees of P' have been rendered, 2) a user-defined number of subtrees of P' has been rendered, or 3) the drawing area is completely filled with subtrees. The first two conditions result in a circle-shaped tag cloud due to the spiral placement of the subtrees. The circle may be cut by the border of the drawing area if it does not completely fit. The third condition results in a tag cloud with the shape of the drawing area.

Finally, we implemented a method that combines the above conditions. It automatically scales the font sizes of the prefixes so that a given number of subtrees fits exactly in the pre-defined drawing area. Different scaling factors are tested until a solution is reached, while the relative differences between the font sizes are retained as much as possible. In the next section, we provide examples of a circle-shaped tag cloud and a rectangle-shaped one.

IV. Application Example

We applied prefix tag clouds in the context of text summarization to evaluate their usefulness. The investigated text corpus is a large dataset containing publication information of the computer science bibliography DBLP [21]. We used prefix tag clouds to create visual summaries of the words that appear most often in the publication titles. Such summaries provide an overview on the topics of the field and can serve as a starting point for analysis [22]. As we will show in this section, prefix tag clouds may provide additional insight in such analyses.

We used the XML file of the DBLP dataset, dating from February 25, 2013. It contains bibliographic information for more than 2.1 million publications in the field of computer science. We transformed the publication titles into individual tags with techniques common in tag cloud generation [4], [11]: We first used regular expressions to remove special characters and separate words by spaces. We then converted the tags to lowercase and removed stop words, i.e. common words like “the”, “is”, or “at” that do not carry relevant meaning. Finally, we counted the frequencies with which the words appear in the titles. As a result, we got the data required to generate prefix tag clouds, i.e. a set of tags T and the mappings $T \longrightarrow \mathbb{N}$ defined by f_{freq}.

Figure 4a depicts a rectangle-shaped prefix tag cloud showing the 150 words that appear most frequently in all publication titles. Very frequent words, such as “system”, “network”, and “analysis”, become immediately apparent. These words occur around 100,000 times in the titles. Words with small font sizes, such as “integration” or “random”, occur around 14,000 times. Words below this frequency value are not displayed in this highly filtered view.

The grouping in the prefix tag cloud reveals that the singular and plural forms of several words occur with similar frequencies in the paper titles, as the plural “s” has often nearly the same font size (e.g. “system” or “network”). Other words occur more often in either singular (e.g. “image” or “method”) or plural (e.g. “graph”). The word “model” is not only used in singular and plural but additionally in its gerund form. The gerund is also popular for the words “process” and “test”. Finally, the words “data” and “database” are grouped, while “data” is used more frequently, as indicated by its larger font size.

Figure 4b depicts another prefix tag cloud generated from the DBLP dataset. It has a circle shape and shows all words that occur most frequently together with the word “visual” or any other word having “visual” as prefix. We generated it from the subset of publication titles that resulted after filtering for the string “visual”. As this string appears in all publication titles of the subset (which is 31,802 times), it has the largest font size in the prefix tag cloud.

The prefix tag cloud contains several subtrees that are similar to the ones visualized in Figure 4a. There are also new subtrees, such as one consisting of the prefixes “real” and “ity” or another grouping “audio” and “visual”. Furthermore, we can observe that the word “program” is often used in its gerund form while “technique” is mostly used in plural. The subtree of “visual” is similar to the one that already served as an example in Figure 2. The prefix is most often used in the word “visualization”, while it also appears frequently in the words “visualizing”, “visualisation”, “visually”, and the plural form of “visualization”. Interestingly, even the German word “visualisierung” is part of the subtree, even though publications in computer science are usually written in English.

V. Conclusion and Future Work

We have introduced the prefix tag cloud—a tag cloud variant that makes use of prefix trees. Different word forms are visually grouped by color and space, facilitating their identification and comparison in the tag cloud. The left-to-right orientation of the prefix subtrees leads to a well-readable tag cloud layout consistent with the dominant reading direction in English and other languages. From preliminary, informal user feedback, we have indication

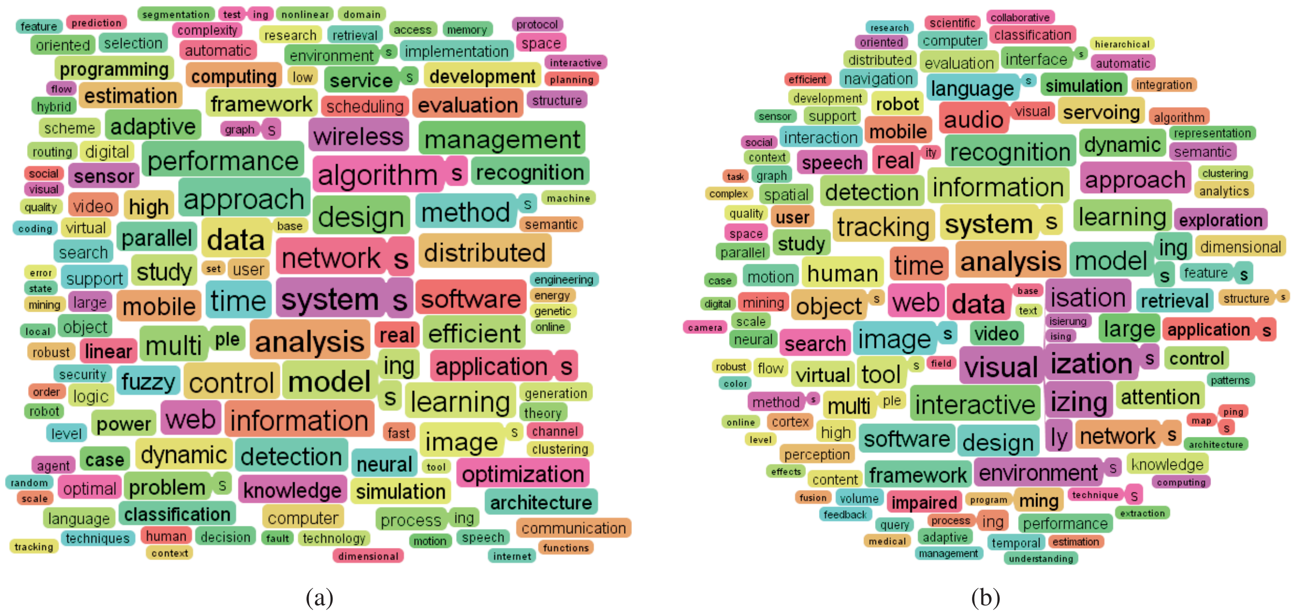

Figure 4. Differently shaped prefix tag clouds, each showing 150 words from publication titles indexed in the computer science bibliography DBLP: (a) most frequent words of all publication titles, b) most frequent words of all publication titles containing the string "visual".

that prefix tag clouds can be learned fast (even without any instruction) and are easily understandable to viewers. Furthermore, we have illustrated the usefulness of prefix tag clouds for the example of a large database with bibliographic information.

In contrast to other work, the approach does not require any semantic analysis of the tags and their relationships but relies purely on lexicographical ordering, which makes it versatile and easy to implement. On the other hand, the lack of semantic information comes with limitations; for example, words may be grouped that are not related (like "gene" and "general"). However, we do not expect these rare cases to cause serious problems when using prefix tag clouds.

In future work, we plan to equip prefix tag clouds with interaction techniques that provide additional support for the visual analysis of texts, such as the interactive highlighting of co-occurring words [23]. We will also consider other application domains and types of data that may benefit from prefix tag clouds. In addition, we will investigate further extensions to the layout. One idea is to combine the circular layout with a clustered one to additionally support the visual identification of semantically related tags.

References

[1] F. B. Viégas and M. Wattenberg, "Tag clouds and the case for vernacular visualization," *interactions*, vol. 15, no. 4, pp. 49–52, 2008.

[2] S. Lohmann, J. Ziegler, and L. Tetzlaff, "Comparison of tag cloud layouts: Task-related performance and visual exploration," in *Proceedings of the 12th IFIP TC 13 International Conference on Human-Computer Interaction: Part I*, ser. INTERACT '09. Springer, 2009, pp. 392–404.

[3] J. Sinclair and M. Cardew-Hall, "The folksonomy tag cloud: when is it useful?" *Journal of Information Science*, vol. 34, no. 1, pp. 15–29, 2008.

[4] J. Feinberg, "Wordle," in *Beautiful Visualization*, J. Steele and N. Iliinsky, Eds. O'Reilly, 2010, pp. 37–58.

[5] O. Kaser and D. Lemire, "Tag-cloud drawing: Algorithms for cloud visualization," in *WWW' 07 Workshop on Tagging and Metadata for Social Information Organization*, 2007. Available: http://www2007.org/workshops/paper_12.pdf

[6] C. Seifert, B. Kump, W. Kienreich, G. Granitzer, and M. Granitzer, "On the beauty and usability of tag clouds," in *Proceedings of the 12th International Conference on Information Visualisation*, ser. IV '08. IEEE, 2008, pp. 17–25.

[7] "Wordle – Beautiful Word Clouds," http://www.wordle.net.

[8] "Tagxedo – Word Cloud with Styles," http://www.tagxedo.com.

[9] "Tagul – Gorgeous Tag Clouds," http://tagul.com.

[10] W. Cui, Y. Wu, S. Liu, F. Wei, M. X. Zhou, and H. Qu, "Context-preserving, dynamic word cloud visualization," *IEEE Comput. Graph. Appl.*, vol. 30, no. 6, pp. 42–53, 2010.

[11] M. Dörk, D. M. Gruen, C. Williamson, and M. S. T. Carpendale, "A visual backchannel for large-scale events," *IEEE Trans. Vis. Comput. Graphics*, vol. 16, no. 6, pp. 1129–1138, 2010.

[12] M. Wattenberg and F. B. Viégas, "The word tree, an interactive visual concordance," *IEEE Trans. Vis. Comput. Graphics*, vol. 14, no. 6, pp. 1221–1228, 2008.

[13] P. Gambette and J. Véronis, "Visualising a text with a tree cloud," in *Classification as a Tool for Research*, H. Locarek-Junge and C. Weihs, Eds. Springer, 2010, pp. 561–569.

[14] Y.-X. Chen, R. Santamaría, A. Butz, and R. Therón, "TagClusters: Semantic aggregation of collaborative tags beyond tagclouds," in *Proceedings of the 10th International Symposium on Smart Graphics*, ser. SG '09. Springer, 2009, pp. 56–67.

[15] C. Collins, F. B. Viégas, and M. Wattenberg, "Parallel tag clouds to explore and analyze faceted text corpora," in *Proceedings of the IEEE Symposium on Visual Analytics Science and Technology*, ser. VAST '09. IEEE, 2009, pp. 91–98.

[16] B. Lee, N. H. Riche, A. K. Karlson, and S. Carpendale, "SparkClouds: Visualizing trends in tag clouds," *IEEE Trans. Vis. Comput. Graphics*, vol. 16, no. 6, pp. 1182–1189, 2010.

[17] K. Fujimura, S. Fujimura, T. Matsubayashi, T. Yamada, and H. Okuda, "Topigraphy: visualization for large-scale tag clouds," in *Proceedings of the 17th International Conference on World Wide Web*, ser. WWW '08. ACM, 2008, pp. 1087–1088.

[18] A. Jaffe, M. Naaman, T. Tassa, and M. Davis, "Generating summaries and visualization for large collections of geo-referenced photographs," in *Proceedings of the 8th ACM International Workshop on Multimedia Information Retrieval*, ser. MIR '06. ACM, 2006, pp. 89–98.

[19] "WP-Cumulus – Word Press Plugin," http://wordpress.org/extend/plugins/wp-cumulus/.

[20] E. R. Tufte, *The Visual Display of Quantitative Information*, 2nd ed. Graphics Press, 2001.

[21] M. Ley, "DBLP: Some lessons learned," *Proceedings of Very Large Data Bases*, vol. 2, no. 2, pp. 1493–1500, 2009.

[22] S. Liu, M. X. Zhou, S. Pan, W. Qian, W. Cai, and X. Lian, "Interactive, topic-based visual text summarization and analysis," in *Proceedings of the 18th ACM International Conference on Information and Knowledge Management*, ser. CIKM '09, 2009, pp. 543–552.

[23] S. Lohmann, M. Burch, H. Schmauder, and D. Weiskopf, "Visual analysis of microblog content using time-varying co-occurrence highlighting in tag clouds," in *Proceedings of the International Working Conference on Advanced Visual Interfaces*, ser. AVI '12. ACM, 2012, pp. 753–756.