Visualising a Text with a Tree Cloud

Philippe Gambette, Jean Véronis

▶ **To cite this version:**

Philippe Gambette, Jean Véronis. Visualising a Text with a Tree Cloud. IFCS'09: International Federation of Classification Societies Conference, Mar 2009, Dresde, Germany. pp.561-569, 10.1007/978-3-642-10745-0_61 . lirmm-00373643v2

HAL Id: lirmm-00373643
https://hal-lirmm.ccsd.cnrs.fr/lirmm-00373643v2

Submitted on 7 Aug 2010

HAL is a multi-disciplinary open access archive for the deposit and dissemination of scientific research documents, whether they are published or not. The documents may come from teaching and research institutions in France or abroad, or from public or private research centers.

L'archive ouverte pluridisciplinaire **HAL**, est destinée au dépôt et à la diffusion de documents scientifiques de niveau recherche, publiés ou non, émanant des établissements d'enseignement et de recherche français ou étrangers, des laboratoires publics ou privés.

Visualising a Text with a Tree Cloud

Philippe Gambette[1] and Jean Véronis[2]

[1] L.I.R.M.M., UMR CNRS 5506, Université Montpellier 2.
[2] L.I.F., UMR CNRS 6166, Université de Provence.

Summary. Tag clouds have gained popularity over the internet to provide a quick overview of the content of a website or a text. We introduce a new visualisation which displays more information: the tree cloud. Like a word cloud, it shows the most frequent words of the text, where the size reflects the frequency, but the words are arranged on a tree to reflect their semantic proximity according to the text. Such tree clouds help identify the main topics of a document, and even be used for text analysis. We also provide methods to evaluate the quality of the obtained tree cloud, and some key steps of its construction. Our algorithms are implemented in the free software TreeCloud available at `http://www.treecloud.org`.

Key words: information visualisation, tag cloud, semantic proximity, hierarchical clustering, arboricity.

1 Introduction

Tag clouds have become very popular on the web. They allow the representation of entire websites in a compact way, through a set of tags whose size or colour reflects their frequency of use [18]. Tags are usually manually associated to the individual articles. However, *word clouds* have been propopsed, that can be built directly from a text using the word frequencies, after getting rid of stop words.

The words of a tag or word cloud are often sorted in alphabetical order. This ordering provides no information, although it could be used to express some semantic information on the displayed words, captured using their cooccurrence level. Such improvements of tag clouds have appeared in the literature, for example in [12], where unsupervised clustering, with the number of clusters given as a parameter, is first used to put similar tags on the same line, followed by a reorganization of the lines to group together similar clusters.

Graphs can also be used to display words as well as their cooccurrence relationships, for example in some text mining software like WordMapper [8], Blake Shaw's visualisation of Del.icio.us tags [16], or Chris Harrison's Bible

Visualisation [11]. Other approaches have considered multidimensional scaling [4, 6] or factor analysis [2, 19] to express the semantic proximity by displaying the most frequent words in two or three dimensions. However, these visualisations are often quite complex, and difficult to read and analyze quickly.

Here we propose to use a tree to reflect the semantic distance between words of a tag cloud, and we call it a *tree cloud*. Then, the distance between two words is given by the length of the path between them in the tree. In fact, the idea of using a tree in this context was already given in [13], but the tree was not used explicitly and was just a step in an algorithm to display the tag cloud in a compact way.

The problem of finding a tree which reflects a distance matrix was introduced in bioinformatics to reconstruct phylogenetic trees from the information on the distances between their leaves. This very active field has provided algorithms which were also used in text and information processing to represent for example proximity inside a set of texts. It has also been used to reflect the semantic distance between words, according to Google [3], or inside a text [2, 17], but was not used yet to enhance tag clouds.

We describe how to build such tree clouds in Section 2. For each step of the algorithm, we give alternative methods or formulas. Then, in Section 3, we present some possible test procedures to evaluate the quality of the obtained tree cloud, or the method choosed to generated it. We will focus on a corpus of 138 campaign speeches by Barack Obama, retrieved at `http://www.barackobama.com/speeches/`.

2 Constructing a Tree Cloud

We consider that we are given an input text containing t words, and detail how to build a tree cloud which describes it.

2.1 Building the List of Frequent Terms

The first step is to extract from this text the list of its most frequent words. Before this process, punctuation should be removed, and other changes in the text can be performed: conversion to lower case or lemmatization (sometimes it should not be applied, see for example “Americans” and “American”, which, interestingly, appear in different subtrees in Figure 1). Some words can also be grouped together, for example different ways to refer to a person: “Barack Obama”, “President Obama”, “Obama”...

Once the list of most frequent words is obtained, stop words (words unlikely to have a semantic value) may be removed to get a meaningful tree cloud. This operation is crucial in any word cloud as well, because stop words are among the most frequent, and even on top of the list. Finally, we consider that we obtain a list L of k words, with their frequency.

2.2 Building the Distance Matrix

We then compute some semantic distance between the words in L. Note that we use the word "distance" to refer in fact to a *dissimilarity*, that is, the triangle inequality may not be satisfied; we only guarantee that the distance matrices are symmetric and contain positive numbers, with 0 on the diagonal.

We use the classical principle that the semantic distance between two words in a text is well captured by their cooccurrence. However, there is no ultimate formula to compute the semantic distance, and many have been used in different contexts: more than 20 were gathered and uniformly defined in [5].

These formulas of cooccurrence distance between two words w_i and w_j are based on a set of portions of the text. $O^{11}_{i,j}$ (resp. $O^{12}_{i,j}$, $O^{21}_{i,j}$, $O^{22}_{i,j}$) counts the number of such portions which contain w_i and w_j (resp. w_i but not w_j, w_j but not w_i, neither w_i nor w_j). A portion can correspond to a sentence, a paragraph, or just a sliding window, depending on the type of text whose tree cloud is being built.

For sliding windows, two parameters have to be chosen: the width w of the window (by default, 30 words), and the size of the sliding step s between two consecutive windows (1 by default). We discuss the choice of these two parameters in section 3.

For the second parameter, a one word sliding step should be chosen to get the most accurate cooccurrence computation. In this case, our algorithm to compute the O^{ab} matrices consists in storing the set L_w of the words of L currently contained in the sliding window with their number of occurrences in the window, updating the content of the O^{11} matrices in $O(min(w, |L|)^2)$, and then updating L_w (in constant time) when the sliding window is shifted. This provides an algorithm of complexity $O(t.min(w, |L|)^2)$ which is in practice much faster than the naive algorithm which computes the cooccurrence for each pair of words in $O(|L|^2.t)$. Note that the beginning of the sliding window of width w starts at position $1 - w$, and stops at position t. This ensures that each word has the same weight in the cooccurrence distance.

2.3 Building the Tree

The most popular tree reconstruction algorithm is Neighbor-Joining [14]. For trees reconstructed from textual data, the method mostly used is a variant of AddTree [15] proposed by Barthélémy and Luong [1]. It is not clear whether this method is used because it is adequate for such data, or just because Neighbor-Joining was not popular enough when this field of research started. Other heuristics have been proposed to reconstruct a tree from a distance which is not close to a tree distance, for example a numerical procedure which consists in fitting the distance to a tree distance [7] or a more recent one based on quartets [3].

The bootstrapping methods to evaluate the quality of tree clouds presented in Section 3 can help choosing the most appropriate tree reconstruction

method for some data. Currently, our program uses only Neighbor-Joining as implemented in SplitsTree, but adding some format conversion functions, to make other tree reconstruction algorithms available, is ongoing work.

2.4 Building the Tree Cloud

The size of keywords can simply reflect the frequency of words, as it is usually the case in tag clouds, or, it can be used differently, for example, to reflect the statistical significance of the various words with respect to a reference corpus. For instance, the words trees representing Barack Omaba's and George Bush's discourses could be contrasted that way: the largest words would be those which are the most salient for each of them.

Keyword colours can also convey information. One obvious use is the categorisation according to topics (e.g. Sports, Politics, Business, etc. on a news website). Brightness could be used to show whether the word appears in one same place in the text or in many places (according to some dispersion coefficient). If the corpus is associated with dates, the most recent words can be displayed with the highest intensity or with a different colour, as in Figure 1.

Information can be conveyed also by edge thickness, length and colour. However, it remains to be seen how much information can be superimposed in the same tree without disturbing its overall readability. A good trick to improve the general aspect of the tree cloud is to force unit edge length. This avoids the long branch problem which occurs with most of the semantic distances: the branches leading to the leaves are very long and the structure of the tree is hidden in the center. The obtained visualization reflects the semantic distance less faithfully, but the subtree topology appears more clearly.

3 Evaluating the Quality of a Tree Cloud

Tree clouds are useful to get a quick glance at the content of a text. However, one could also use them for further analysis by looking more carefully at clusters of words associated the different subtrees. The tree should then give a good representation of the semantic distance between words in the text, although this distance is just approximated by the tree distance.

In this section, we give some methods to evaluate this quality. They can also be used to choose appropriate distance formulas or tree reconstruction methods for some given input data.

Note that, contrary to phylogenetic tree where the tree distance is supposed to have a biological interpretation (it can represent time, or the number of mutations), the semantic distance formulas which are reflected by the tree have no clear interpretation. In fact, applying an increasing function to the distances would not change their ordering, and the obtained distances can be considered as valid as the input. This explains why the quality of the tree

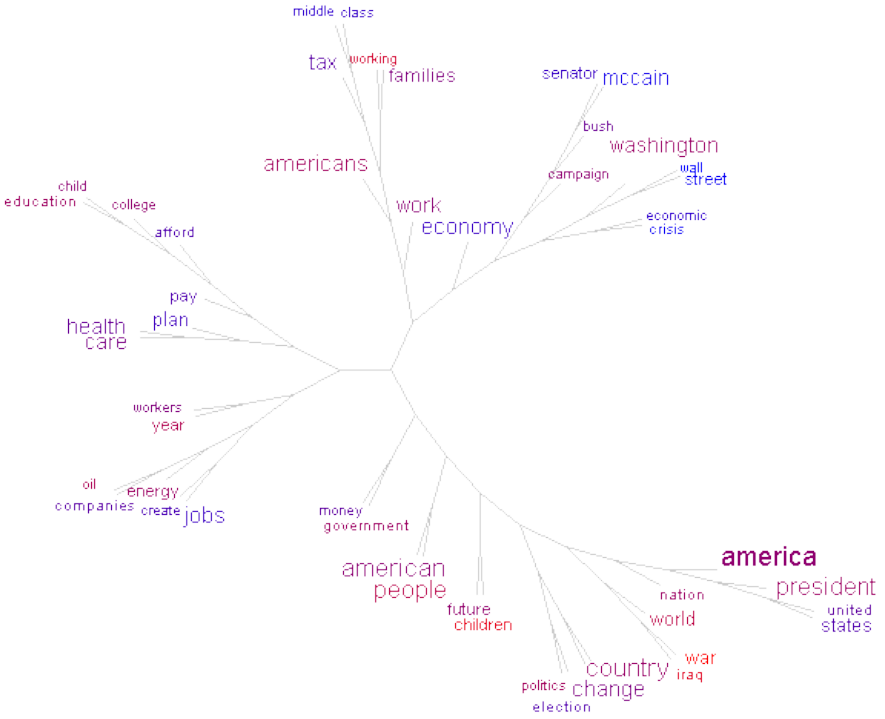

Fig. 1. 50 word tree cloud of Obama's presidential campaign speeches, with Jaccard distance, and chronology colouring. Red corresponds to the beginning of the campaign ("children", "Irak", "war", "world"), while blue corresponds to the end ("McCain", "Wall Street", "crisis", "taxes").

clouds should not be evaluated by direct comparison of the distance matrix and the tree distance.

Instead, we propose an bootstrap evaluation based on the stability of the results. If small changes in the input text provide a similar tree cloud, then it is *stable*, and the method to build it can be considered *robust*. We will also give another criterion, *arboricity*, which evaluates how close the semantic distance is to a tree distance, and discuss how it is related to stability.

3.1 Stability and Robustness

Evaluating the stability of a tree cloud requires two steps: altering the input text, and computing how much the tree has changed. For text alteration, we implemented two procedures: either each word is deleted with probability p, or the text is cut into 100 parts, and some of those parts are removed.

Then, to evaluate stability, we count how many edges of the tree built from the original text are present in the one built from the altered text, seeing each

edge as a *split*, i.e. a bipartition of the leaves into two separate clusters. Each edge leading to a leaf is trivially present in both trees, so we neglect those *trivial splits*, and define stability as the proportion of non-trivial splits which appear in both trees.

3.2 Arboricity

Tree reconstruction algorithms are more efficient on distances which fit a tree. Thus, one can expect that the tree cloud will be more stable for semantic distance formulas which provide distances with a good arboricity, that is, close to a tree distance. We first give two formulas which evaluate arboricity, and we will test below whether this can be an objective criterion to evaluate the quality of the tree cloud by avoiding the bootstrap procedure.

The *discrete arboricity*[3] [10] of a symmetric matrix $M \in [0,1]^{n\times n}$ is

$$Arb_d(M) = \frac{1}{\binom{n}{4}} |\{\{i,j,k,l\} \text{ such that } Smax - Smed < Smed - Smin\}|, \quad (1)$$

where $Smin$, $Smed$ and $Smax$ are the three sums $M_{i,j} + M_{k,l}$, $M_{i,k} + M_{j,l}$ and $M_{i,l} + M_{j,k}$, sorted in increasing order. The *continuous arboricity* [9] of M is

$$Arb_c(M) = \frac{1}{\binom{n}{4}} \sum_{i<j<k<l} \frac{Smed - Smin}{Smax - Smin}. \quad (2)$$

3.3 Distance Comparison on the Obama Corpus

We applied our quality control procedures on the tree clouds obtained on Obama's speeches with the 13 semantic distance formulas implemented in TreeCloud, with text alteration based on removing words with 5% probability.

The results are available as supplementary material for this article at `http://www.treecloud.org`, and a summary given in Table 1 shows that all distance formulas[4] perform approximately equally well, except except mutual information which is very bad, normalized Google distance, and oddsratio. Although oddsratio gives tree clouds with lower stability, it is still an interesting distance, because it provides nice trees even when the edge lengths are not forced to unit length.

The correlation between arboricity and stability is not very good (0.6 correlation coefficient). However, very bad arboricity (below 50%) implies bad stability, and very good arboricity (over 90%) implies good stability.

[3] This formula reflects how much the four point condition, characteristic of tree distances, is verified for each subset of 4 elements.

[4] The abbreviations correspond to: Liddell, geometric mean [5], Jaccard, Dice, minimum sensitivity, z-score, Hyperlex [17], χ^2, Poisson-Stirling, log-likelihood, oddsratio, normalized Google distance [3], mutual information.

distance:	Li.	g.m.	Ja.	Di.	m.s.	z.s.	Hy.	χ^2	P.S.	l.l.	od.	NGD	m.i.
Av. stability (%)	56.8	56.8	56.6	56.5	56.0	55.1	55.0	53.9	52.3	51.8	30.9	27.1	17.6
Arb_d (%)	67.2	64.3	65.3	64.4	64.8	66.0	64.4	68.9	53.4	61.8	55.6	59.0	42.1
Arb_c (%)	70.0	66.3	67.6	66.4	66.9	68.2	66.4	72.7	55.4	65.0	55.1	57.5	42.4

Table 1. Average stability (5% alterations) and arboricity of various semantic distances for the tree clouds of the 50 most frequent words of 138 campaign speeches, with sliding windows of size 30 and sliding gap 1.

3.4 Robustness to Parameter Variations

We evaluated stability to decreasing ($w = 10$ words) or increasing ($w = 100$ words) sliding window width, and to variations of the sliding step ($s = 5, 15, 30$) which give similar results for the different distances, shown in Table 2, except for loglikelihood and Poisson-Stirling which seem less robust to sliding step variations.

distance:	Li.	g.m.	Ja.	Di.	m.s.	z.s.	Hy.	χ^2	P.S.	l.l.	od.	NGD	m.i.
$w = 10$	36.2	35.8	35.8	35.5	34.7	35.4	34.6	35.3	34.1	34.4	17.8	10.4	1.2
$w = 100$	28.9	29.1	29.7	29.1	28.5	29.2	25.4	28.4	21.6	27.4	12.5	6.8	1.4
$s = 5$	68.9	70.7	71.0	69.2	69.9	67.0	67.6	61.5	50.1	52.3	27.2	42.1	25.6
$s = 15$	47.9	48.9	48.9	48.5	48.1	48.0	47.8	45.2	39.6	41.6	23.3	24.5	9.7
$s = 30$	34.0	35.0	35.5	35.5	35.6	34.3	33.1	33.9	31.5	32.5	17.7	14.0	3.1

Table 2. Average stability of various semantic distances, for the tree clouds of the 50 most frequent words of 138 campaign speeches, to changing the sliding window width (30 by default) or the sliding step (1 by default).

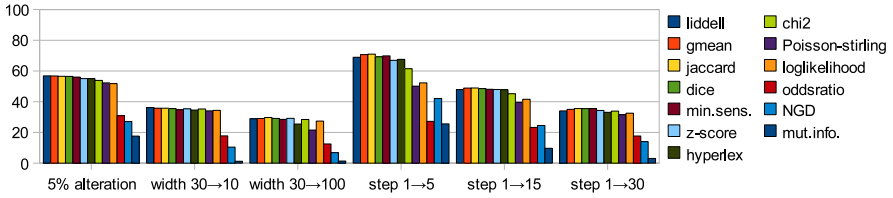

Fig. 2. Visualization of stability results of Tables 1 and 2.

4 Conclusion

We presented a new visualisation tool which improves word clouds to get a quick overview of the content of a text, as well as some quality control procedures to evaluate how much a tree cloud can be trusted for text analysis. The study of other uses in this context (topic-focused tree cloud, tree cloud comparison...) is ongoing work.

5 Acknowledgments

We thank the French ANR project ANR-08-EMER-011-01 (Phyl-ARIANE) for support. We thank Vincent Ranwez and Alain Guénoche for useful discussions, and Virginie Lethier for many references on discourse analysis and lexicometry.

References

1. J.P. Barthélémy, N.X. Luong. Sur la topologie d'un arbre phylogénétique : aspects théoriques, algorithmes et applications l'analyse de données textuelles, *Mathématiques et Sciences Humaines*, 100, p. 57-80, 1987.
2. E. Brunet. Un hypertexte statistique : Hyperbase. *JADT 1993*, p. 1-16, 1993.
3. R. Cilibrasi, P. Vitanyi. The Google Similarity Distance. *IEEE/ACM Transactions on Knowledge and Data Engineering*, 2007.
4. N.J. van Eck. *Towards Automatic Knwoledge Discovery from Scientific Literature*. MSc Thesis, 2005.
5. S. Evert. *The Statistics of Word Cooccurrences, Word Pairs and Collocations*. Phd Thesis, p. 75-91, 2005.
6. K. Fujimura, S. Fujimura, T. Matsubayashi, T. Yamada, H. Okuda. Topigraphy: Visualization for Large-scale Tag Clouds. *WWW2008*, 2008.
7. O. Gascuel and D. Levy. A Reduction Algorithm for Approximating a (Non-metric) Dissimilarity by a Tree Distance. *Journal of Classification*, 13(1), p. 129-155, 1996.
8. J.F. Grimmer. *WordMapper Text*.
9. A. Guénoche, P. Darlu. TreeOfTrees: a New Method to Evaluate Gene Tree Distances. Manuscript, 2009.
10. A. Guénoche, H. Garreta. Can We Have Confidence in a Tree Representation? *Lecture Notes in Computer Science*, 2066, p. 45-56, 2000.
11. C. Harrison. *Visualizing the Bible*. `http://www.chrisharrison.net/projects/bibleviz`, 2008.
12. Y. Hassan-Montero, V. Herrero-Solana. Improving Tag-Clouds as Visual Information Retrieval Interfaces. *InSciT2006*, 2006.
13. O. Kaser, D. Lemire. TagCloud Drawing: Algorithms for Cloud Visualization. *WWW2007*, 2007.
14. N. Saitou, M. Nei. The Neighbor-Joining Method: a New Method for Reconstructing Phylogenetic Trees. *Molecular Biology and Evolution*, 4, p. 406-425, 1987.
15. S. Sattah, A. Tversky. Additive Similarity Trees. *Psychometrika*, 42, p. 319-345, 1977.
16. B. Shaw. Semidefinite Embedding Applied to Visualizing Folksonomies. Manuscript, 2005.
17. J. Véronis. Hyperlex, Lexical Cartography for Information Retrieval. *Computer, Speech and Language*, 18(3), p. 223-252, 2004.
18. F.B. Viégas, M. Wattenberg. Tag Clouds and the Case for Vernacular Visualization. *ACM Interactions*, 15(4), p. 49-52, 2008.
19. J.-M. Viprey. Ergonomiser la Visualisation AFC dans un Environnement d'Exploration Textuelle : une Projection "Géodésique". *JADT 2006*, p. 981-992, 2006.