Abbreviating Text Labels on Demand

Mariana Shimabukuro* Christopher Collins†

University of Ontario Institute of Technology – Canada

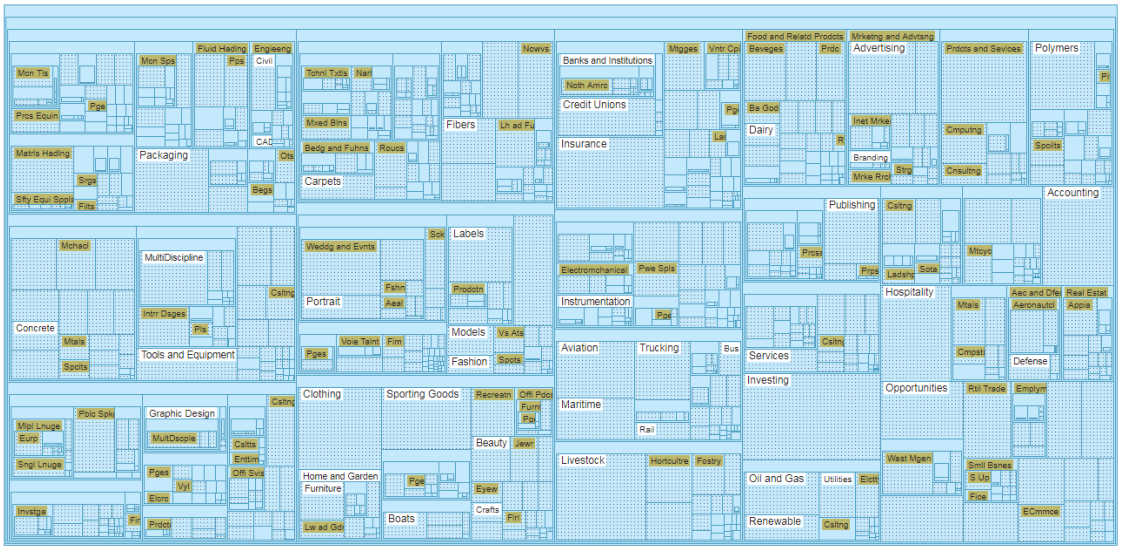

Figure 1: A visualization of the DMOZ dataset. Each of the highlighted labels are being abbreviated by our algorithm, which drops as many letters as needed to fit the text. It chooses the least important letter based on the character and its position within the word.

Abstract

Long text labels is a known challenge in information visualizations. There are some techniques used in order to solve this problem like setting a very small font size. On the other hand, sometimes the font size is so small that the text can be difficult to read. Wrapping sentences, dropping letters and text truncation are some techniques do deal with this problem. In order to investigate a solution for labeling long words we ran a study on how people create and interpret word abbreviations. Based on the study data we designed a new algorithm to automatically make words as short as they need to fit the text. Examples applications of this algorithm are presented in this paper.

Keywords: Text visualization, word abbreviation and text labeling.

1 Introduction

Labeling is a difficult challenge in text visualization [3]. Often long words or phrases are displayed in small font sizes, or overlaping with other labels [2, 6, 8] compromising the visualization readability [4]. Labeling in visualization is defined by Bertini et al. [1] as text labels attached to graphical marks to associate semantic information to data items. Labels contain textual description that characterizes the object, along with visual features (e.g., color, size, etc.).

Techniques to optimize label placement have been studied for decades, mostly for cartographic purposes [5]. Fekete and Plaisant [3] also presented common practices regarding long word labeling. Depending on the visualization, the font size can be as small as needed in order to make the text fit. Other visualizations may apply truncation or omission of the text. Using a shorter label as a substitute when needed (e.g., acronyms) might be helpful. Breaking long labels in multiple lines is also possible. However, there are cases where the text simply overflow/overlap with no special treatment.

In order to solve long text labeling we ran an adaptive crowdsourced study on how people create and understand English word abbreviations [7]. Based on the study results we designed the "Abbreviation on Demand" algorithm, which aims to drop the least important letters of a word based on the study data, shortening labels while maintaining readability. The algorithm uses the probability of dropping letters based on their position within the word and the identity of the characters themselves.

Here we present some results on applying the "Abbreviation on Demand" algorithm in visualizations (Figure 1) and some other scenarios where we compare the algorithm performance with other abbreviation techniques (Figures 2 and 3).

2 Adaptive Crowdsourced Study

We designed and implemented an adaptive crowdsourced study on how people create (encode) and understand (decode) abbreviations, and if semantic context has any affect on these tasks [7].

The experiment had 80 tasks divided in two types: 40 encoding and 40 decoding tasks. Using semantic distance based on the word2vec[1] model, we selected 80 different words with length varying from 10–16 characters long from Corpus of Contemporary American English (COCA)[2] divided in 4 contextual groups. Within each group we had 20 words that were semantically related between each other (e.g. words about astronomy or education).

The study had 2 conditions: contextual – where the participant had 4 screens with 5 words belonging to the same context; and

*e-mail: marianaakemi.shimabukuro@uoit.ca

†e-mail:christopher.collins@uoit.ca

[1]https://github.com/3Top/word2vec-api

[2]http://corpus.byu.edu/coca/

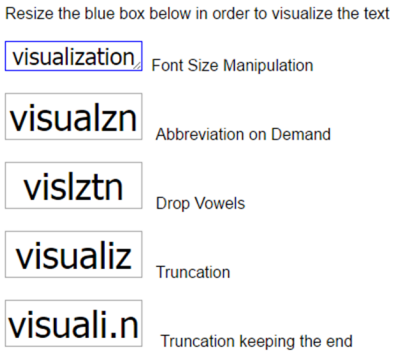

Figure 2: Application for real time comparison of letter dropping choices from each of the different abbreviation techniques (font size manipulation, abbreviation on demand, drop vowels, truncation and truncation while keeping the end).

non-contextual – having 5 screens with 4 words, each word from a different contextual group.

We ran the adaptive study on a crowdsourcing platform called Crowdflower[3] with a total of 100 participants. By adaptive, we mean using the fast crowdsourcing recruitment and being able to evaluate the abbreviations created in the encoding task using the decoding task in close to real time. To achieve this, we implemented a ranking algorithm that selects the most relevant abbreviations from the encoding task and fed it into the decoding task. Submitted abbreviations which were difficult to decode were automatically dropped from the study, allowing us to gather more data on promising approaches.

From the study, we extracted data that allowed us to determine which letters are the most dropped, as well as the most dropped positions within a word. The data also taught us that participants have higher confidence levels for contextual tasks when compared to the non-contextual tasks. This is a relevant fact when considering that text labels in a visualization usually belong to the same context (e.g., data about medicine).

3 "Abbreviation on Demand" Algorithm

In the study, we observed that dropping letters from a word is the best general approach. However, the strategy of simply dropping the vowels did not match human behavior in creating abbreviations. Thus we created the "Abbreviation on Demand" algorithm that, given a word and a desired size, will drop the least important letters based on a score calculated using letter position and context, until the abbreviation's length matches the specified size.

For the score calculation we use the correlation measure given by $corrMx(word[i-1]word[i])$ which is the probability a participant would drop letter i following letter $i-1$ [7] in our encoding task. Also, the $pPos(i)$ probability of dropping a letter depending on its position within the word. In a word, the score for each letter in position i from 0 to the word length is given by:

$$score_{word[i]} = \begin{cases} monoDropProb(word[i]) * pPos(i) & \text{if } i = 0, \\ corrMx[word[i-1][word[i]] * pPos(i) & \text{if } i > 0. \end{cases}$$

where the $monoDropProb(word[i])$ is the probability of a individual letter $word[i]$ being dropped based on the study data. Considering that the correlation measure of a letter depends on the letter that came before, we cannot apply it to the first letter of a word.

We can also make small modifications to the algorithm to consider different use cases. For example, in Figures 1 and 2 we present a modified version of our algorithm to abbreviate words based on their screen size instead of the number of characters. Also, for readability reasons we do not want the font size to be smaller than a parameterized minimum size. So, all we need is how much space is available, the minimum font size, the font name and the word to be place in the screen. The algorithm can drop letter by letter until it fits into the specified available space.

[3] www.crowdflower.com

Original	**Abbreviation**	Drop Vowels	Truncation	Truncation keep end	TOP 1 decoded abbreviaton	TOP 2 decoded abbreviaton	TOP 3 decoded abbreviaton	Original length	60% of length
academically	**acdmcly**	acdmcll	academi	academ.y	acad	academi	acdmcll	12	7
accelerating	**accelng**	acclrtn	acceler	accele.g	acclrtn	accelerat	acceler	12	7
acceleration	**acceltn**	acclrtn	acceler	accele.n	acc	accel	acceler	12	7
adventurers	**advturs**	advntrr	adventu	advent.s	advntrs	adventu	advnturers	11	7
assignments	**assgnms**	assgnmn	assignm	assign.s	assgnmnts	assign	assignments	11	7
atmospheric	**atmsphc**	atmsphr	atmosph	atmosp.c	atmsphr	atmsphrc	atmpheric	11	7
automotive	**autmtv**	autmtv	automo	autom.e	auto	automtv	autom	10	6
circumstance	**circmsc**	crcmstn	circums	circum.e	circums	crcstance	circmstnc	12	7
collisions	**colsns**	cllsns	collis	colli.s	collsns	collsn	collisi	10	6
colonization	**colnzan**	colnztn	coloniz	coloni.n	colonztn	colniztn	colnzation	12	7

Figure 3: Abbreviations except TOP 1, 2 and 3 were created by dropping 40% of the letters from each word. Column "Original" is the original word, "Abbreviation" is the abbreviation created by our algorithm followed by other techniques. TOP 1, is the most accurate abbreviation from our study followed by TOP 2 and 3.

4 Applications

Figure 1 shows the "Abbreviation on Demand" algorithm applied to a treemap visualization in D3 using the category "Business" from the DMOZ data set [9]. When we resize the treemap the words get re-arranged and different words might be abbreviated instead.

Figure 2 compares in how each of the abbreviation techniques work while we resize the text box. Lastly, Figure 3 shows our algorithm performance compared to other techniques when we drop 40% of the letters, as well as abbreviations entered in the study. We have also the API for this algorithm available in our website (https://abbreviation.vialab.ca).

5 Discussion

We presented our work in progress on the "Abbreviation on Demand" algorithm for visualization labeling, based on the probability of dropping letters using character position and context as features. We treated features as independent when multiplying probabilities in the scoring function. In future, we could consider feature dependencies, other features, such as tri-graph, part-of-speech, word root, etc.

Future work includes a mixed approach with font size manipulation to first make smaller, then start dropping. We also want to conduct an evaluation of readability in context.

References

[1] E. Bertini, M. Rigamonti, and D. Lalanne. Extended excentric labeling. In *Computer Graphics Forum*, vol. 28, pp. 927–934. Wiley Online Library, 2009.

[2] M. Bostock. Treemap: D3 example, 2016. Accessed 2017-02-27.

[3] J.-D. Fekete and C. Plaisant. Excentric labeling: dynamic neighborhood labeling for data visualization. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, pp. 512–519. ACM, 1999.

[4] P. Goffin, W. Willett, J.-D. Fekete, and P. Isenberg. Exploring the placement and design of word-scale visualizations. *Visualization and Computer Graphics, IEEE Transactions on*, 20(12):2291–2300, 2014.

[5] K. Mote. Fast point-feature label placement for dynamic visualizations. *Information Visualization*, 6(4):249–260, 2007.

[6] S. Plath. General graded evaluation scale of Sylvia Plath's corpus, 2014. Accessed 2015-11-19.

[7] M. Shimabukuro. An adaptive crowdsourced investigation of word abbreviation techniques for text visualizations. Master's thesis, UOIT, 2017.

[8] N. Y. Times. How the giants of finance shrank, then grew, under the financial crisis - interactive - nytimes.com, 2015. Accessed 2015-11-19.

[9] R. Veras and C. Collins. Optimizing hierarchical visualizations with the minimum description length principle. *IEEE Transactions on Visualization and Computer Graphics*, 23(1):631–640, 2017.