An Adaptive Crowdsourced Investigation of Word Abbreviation Techniques for Text Visualizations

Mariana Akemi Shimabukuro

Faculty of Science
University of Ontario Institute of Technology

This thesis is a partial requirement for the degree of
Master of Science

April 2017

Declaration

I hereby declare that except where specific reference is made to the work of others, the contents of this dissertation are original and have not been submitted in whole or in part for consideration for any other degree or qualification in this, or any other university. This thesis is my own work and contains nothing which is the outcome of work done in collaboration with others, except as specified in the text and Acknowledgements.

Mariana Akemi Shimabukuro
April 2017

Acknowledgements

Firstly I would like to acknowledge my supervisor Dr. Christopher Collins. Dr. Collins has always helped me to developed both my personal and research skills. He consistently allowed this thesis to be my own work, but steered me in the right the direction whenever he thought I needed it.

I would also like to thank Dr. Jeremy Bradbury for his insights and feedback, mostly in the early phases of this project when I had the honour of being co-supervised by Dr. Bradbury as well.

I would also like to acknowledge all the members of Vialab for their encouragement in the gloomiest days and valuable comments during the development of this thesis.

Finally, I must express my very profound gratitude to my parents, my sister, the Sawal family who took me in as one of them, and to my partner for providing me with unfailing support and continuous encouragement throughout my years of study and through the process of researching and writing this thesis. This accomplishment would not have been possible without them. Thank you.

Abstract

A known problem in information visualization labeling is when the text is too long to fit in the label space. There are some common known 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 can also be used. However, there is no research on how these techniques affect the legibility and readability of the visualization. In other words, we don't know whether or not applying these techniques is the best way to tackle this issue. This thesis describes the design and implementation of a crowdsourced study that uses a recommendation system to narrow down abbreviations created by participants allowing us to efficiently collect and test the data in the same session. The study design also aims to investigate the effect of semantic context on the abbreviation that the participants create and the ability to decode them. Finally, based on the study data analysis we present a new technique to automatically make words as short as they need to be to maintain text legibility and readability.

Table of contents

List of figures

List of tables

Chapter 1

Introduction

In the information visualization field, we face challenges when adding text labels. Our study regards the words that are too long to fit in their respective space. A common practice is to make the font size smaller, sometimes so tiny that it is difficult to read (as shown in Figure 1.1). Some designers choose to truncate the text when wrapping (breaking the text in different lines) still does not solve the problem. We can also see some situations where the text labels are simply placed overflowing or overlapping other graphical elements.

From the literature review and a pilot study we categorized some abbreviation techniques:

Font size manipulation: when the font size is minimized.

Drop vowels: when the letters omitted are vowels.

Truncation: when the last letters of the word are removed.

Truncation keeping the end: when the last letters are removed with exception of the very last one.

In Figure 1.2 we can see examples of these techniques applied to the word 'Literature' when we set the minimum font size to 24px. It does not seem like the word needed to be abbreviated, however simulating a case where the space constraint is tighter (12px minimum font size) then the techniques seems to improve the readability.

We are interested in a number of research questions that relate to this project:

1. Can we find a pattern in how people create word abbreviations and how good are people at understanding them?

2. How does the semantic context affect word abbreviations?

3. Can we find an automatic approach to solve this problem?

Context dependency refers to investigate whether encoding (creating an abbreviation/abbreviating) or decoding (interpreting/reading an abbreviation) approach changes if a the word comes

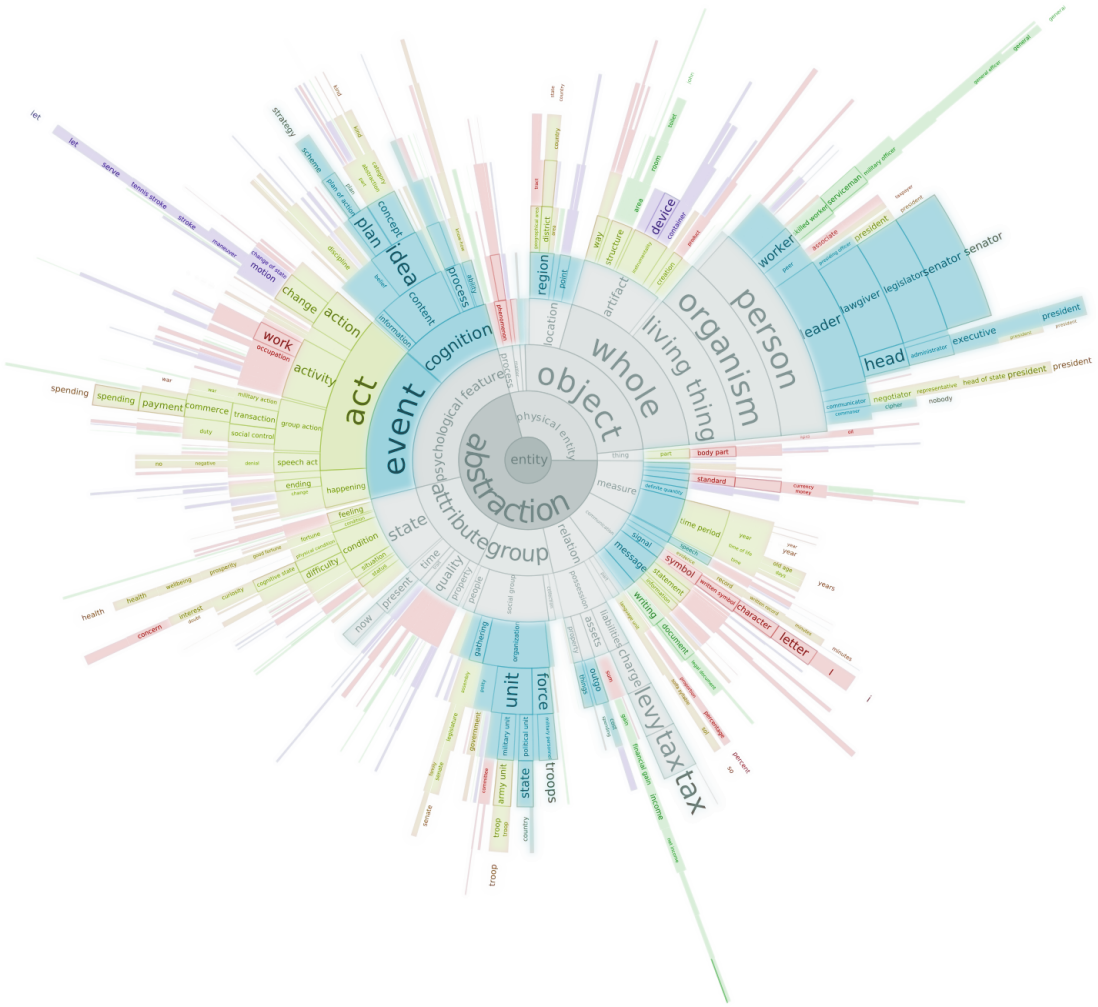

Fig. 1.1 Prioritizing nodes in hierarchical visualizations with the Tree Cut Model [47].

from a group of words that are semantically related to each other. We find this context dependency question relevant because data visualizations are usually contextual, where the data refer to a well-defined topic. Our hypothesis is that having an abbreviation presented within a context would be easier to interpret.

To address this problem we designed an adaptive crowdsourcing experiment. Crowdsourcing is defined as a phenomenon where web workers complete a set of small tasks, for micro-payments on the order of $0.01 to $0.10 per task. Micro-task markets lower the cost of recruiting participants, offering researchers almost immediate access to hundreds of diversified users [16]. By adaptive, we mean taking advantage of the easy and dynamic recruitment that a crowdsourcing platform gives us, and trying to evaluate in close to real time the word abbreviations created by the participants in our decoding task. In order to

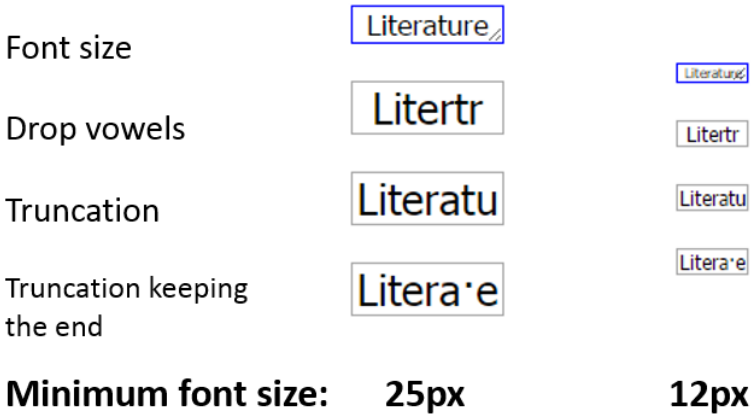

Fig. 1.2 Screenshot from prototype made to compare performance of abbreviation techniques.

achieve this, we are using a ranking algorithm that allows us to decide the most relevant abbreviation to test.

Based on the experiment results, we can understand the effect found on semantic context on decoding word abbreviations; When the decoding task is contextualized participants are more accurate, and their confidence level solving the tasks is higher. We also have compiled data on the dropping probability (probability of a letter being dropped) of letters regarding their position in a word and the letter that comes before it.

Finally, we successfully designed and implemented an automatic abbreviation technique based on the results of the experiment. This algorithm allows us to drop just as many letters as needed in order to fit the most amount of letters. Another upside of our algorithm is how it can be easily modified for different applications such as abbreviating words in a given number of characters or in the screen space available, or even for abbreviating tweets into 140 characters.

1.1 Motivation

This project was born from the observation of several visualizations, followed by literature review and a preliminary study focused on word abbreviation techniques.

1.1.1 Text Visualizations

The long word labeling issue has been present in many text visualizations. In order to picture it better, see Figures 1.1, 1.3 and 1.4.

In Figure 1.1, we have a different approach for labeling, where the font size gets smaller in order to make the word fit in the bounding box. In Figure 1.3, the long labels are displayed in the same font size as the shorter labels, which makes the text overflow the boxes and the labels to overlap. Lastly in Figure 1.4, we can observe three different techniques. If it detects the label is too long to fit, it breaks the text in lines. Then, if breaking in lines is not possible, it truncates the text. And finally, if the text is still too big to be displayed, it omits the label.

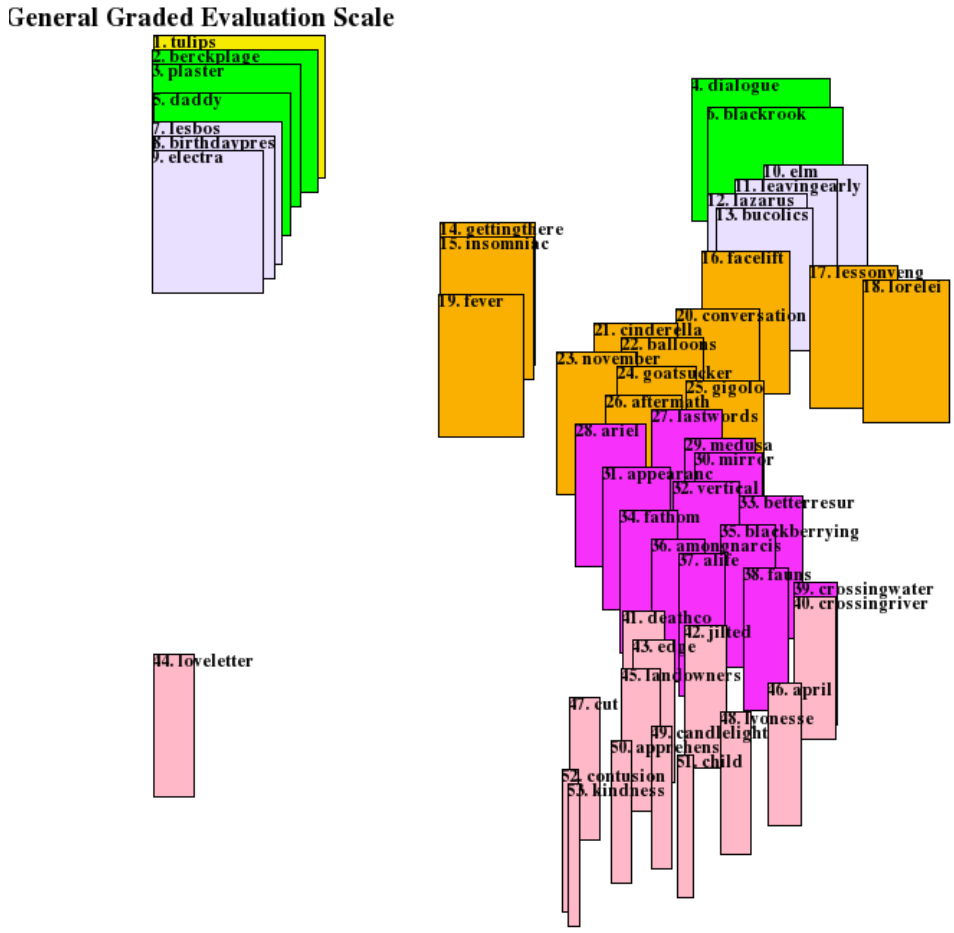

Fig. 1.3 Sylvia Plath's "General Graded Evaluation" visualization of poems clustered by SPARSAR [10] poetry analysis [38]. In this visualization we can observe that the text labels overflow the space and overlap each other making it difficult to read.

Even though the data visualization community tries to deal with long word labeling issues, to our knowledge there is no conventional solution for this problem yet. Our approach is inspired by SentenTree [20], a technique for visualizing the content of unstructured social media text. SentenTree displays frequent sentence patterns abstracted from a corpus of social

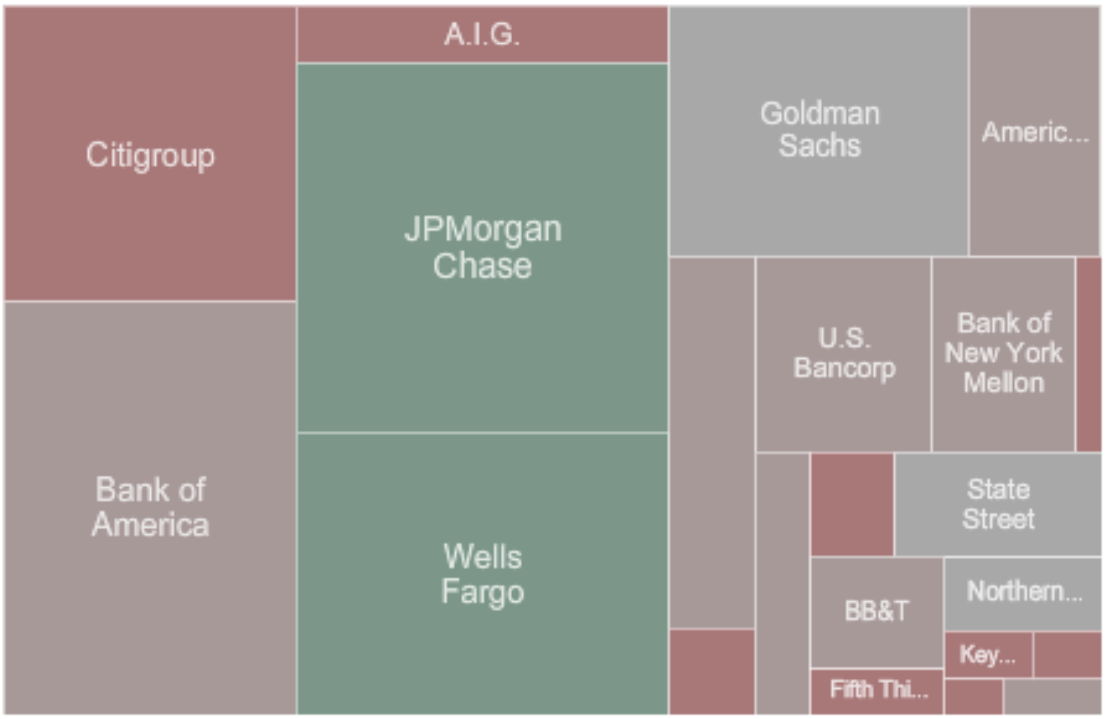

Fig. 1.4 How the Giants of Finance Shrank, Then Grew, Under the Financial Crisis [45]. In this picture we can see a number of techniques – text wrapping, acronyms, truncation and label omission – being applied to the text labels in order to make them fit.

media posts (Twitter[1]) and it can help people gain a rapid understanding of key concepts and opinions in a large social media text collection. Similar to SentenTree, we want to have access to a large enough amount of data that would allow us to find a dropping letter pattern within words instead of sentence patterns.

1.1.2 Typography

The SafeFont project aims to improve the inaccurate interpretation of medication labels by designing and testing a set of new approaches to typeface and display technology that is flexible enough to be used — onscreen and in print — by anyone across health-care settings [41]. Figure 1.5 displays a label design example from the SafeFont project.

The type of typography design that SafeFont presents has inspired us to investigate a possibility of designing a data-driven approach for typeface design for abbreviations. For example, making the consonants bigger than the vowels, or modifying the characters ligatures. We designed a sketch showed in Figure 1.6, which is a comparison between the long word, the abbreviated word and the font manipulated word labels.

[1] http://twitter.com. Accessed 2017-04-25.

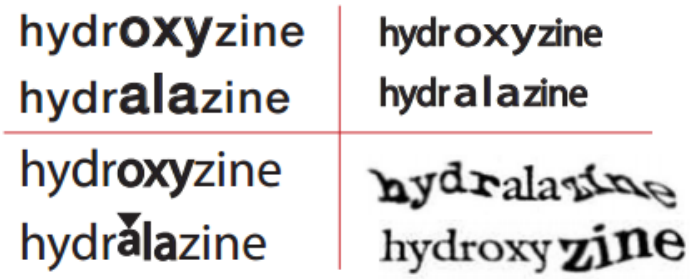

Fig. 1.5 Examples of possible type design strategies to increase disambiguation between similar names [42].

Fig. 1.6 Sketch of font manipulation.

1.2 Contributions

The main contribution of this work follow:

1. An adaptive crowdsourced study design (Chapter 3) which allowed us to combine human computing with recommendation systems in order to obtain more relevant data faster.

2. An investigation on effects of semantic context dependency of a group of words on decoding and encoding abbreviations (Chapter 4).

3. Tabulated and visualized data on dropped letters frequency, frequency of dropped letter by position within words and decoding accuracy (Chapter 4).

4. Designed and implemented a proof-of-concept recommendation algorithm "Abbreviation on Demand" for abbreviating words based on dropping letters probabilities (Chapter 5).

1.3 Organization

Chapter 2 presents a review of work related to text labeling, visual perception and typeface design in visualization, word abbreviation techniques and ranking algorithms. Chapter 3

discusses our pilot study, as well as our main experiment design. Experiment results and discussion can be found in Chapter 4, followed by design and application of our abbreviation algorithm on Chapter 5. Lastly, Chapter 6 addresses some limitations, conclusion and ideas for future work.

Chapter 2

Background

In this chapter we present a review of related work on labeling in visualization, manipulation of font attributes, typography design, existing word abbreviation techniques and ranking algorithms.

2.1 Long Words Labeling in Data Visualization

Labeling is a difficult challenge in text visualization [13]. Often long words or phrases are displayed in small font sizes, or they overlap with other labels (Figure 1.1 and 1.3) compromising the visualization readability [14].

Labeling in visualization is defined by Bertini et al. [5] as text labels attached to graphical marks to convey semantic information association to data items that along with visual features (e.g., color, size, shape, etc.) contain some textual description that characterizes the object.

Techniques to optimize label placement have been studied for decades, mostly for cartographic purposes [31]. Fekete and Plaisant [13] 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 (Figure 1.1). Other visualizations may apply truncation or omission of the text (Figure 1.4). 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 some cases where the text simply overflow/overlap with no special treatment (Figure 1.3).

SentenTree [20], a technique for visualizing the content of unstructured social media text, displays frequent sentence patterns abstracted from a corpus of social media posts (Twitter[1]). It can help people gain a rapid understanding of key concepts and opinions in a large social

[1] http://twitter.com. Accessed 2017-04-25.

media text collection. Like SentenTree, we want to have access to a large amount of data that would allow us to find frequent patterns of letters omitted within words instead of sentence patterns.

Visual search is a task that data visualizations often require (e.g. tag clouds [39] and cartography [34]), where visual clutter caused by long text labels can cause a serious problem. The perceptual psychology studies in visual word recognition [2, 3, 53] teach us about how people do not read letter by letter in a word, but they recognize its shape. Consequentially, the first as well as the last letters are more important for word recognition [2].

A study by Balota and Chumbley [3] reveals an effect of word frequency in a lexical decision task, where the person has to decide if a string of characters is or not a word. The more common the word is, the easier it is for people to recognize it. This relates to our study in terms of how we decided which words to use in our experiment. And also, drives us to believe that the most frequent letters in English would be easier assumed by people when we omit them in an abbreviation.

2.1.1 Font Attributes in Visualization

Afzal et al. [1] describe the role of labels in visual representations as a supporting role. However, a radical new idea is to generate graphics where the textual labels alone form the visual features — in other words, the labels become the image. It expresses the importance of keeping the readability of labels in a visualization, where the text itself, and the way it is displayed, can be the desired message to be delivered.

The FatFonts by Nacenta et al. [32] is a technique designed to bridge the gap between symbolic and visual representations through the manipulation of the shape of Hindu-Arabic numerals. It modifies the glyphs themselves to create a meaningful visual overview. Text and numbers are also part of modern visualizations, but they are usually separate elements (e.g., labels), and their typeface is influenced by legibility or space efficiency, not the image.

GreenArrow by Wong et al. [52] is a prototype system that comprises a suite of interactive and animated tools designed to visualize a graph with extended node and/or link labels. The label font starts out larger from the source node and shrinks gradually until it reaches the destination node. This relates to our work, as it is a typographic manipulation in order to optimize label between nodes.

Another visualization that uses typographic features is tag clouds. Tag clouds represent variables of interest (such as popularity) in the visual appearance of the keywords themselves — using text properties such as font size, weight, or colour [4]. One particular example of tag clouds is Wordle, which is a close relative of tag clouds, encoding word frequency information with font size. Even though Wordles are very similar to tag clouds, Wordles

look significantly different from a regular tag cloud. Besides presenting word frequency data, Wordles are more playful regarding the possibilities of color, typography, and composition [48].

The Word Tree is essentially an interactive form of the keyword-in-context (KWIC) technique. Taking a cue from the popularity of tag clouds, Word Tree uses font size to represent the number of times a word or phrase appears. The size is proportional to the square root of the frequency of the word [49].

Brath and Banissi [8] pointed out that cartography has centuries of history with innovative font encoding of information in documents hundreds of years old. There are many techniques for creating emphasis and differentiation with font, with various guidelines and conventions. They [8] also proposed and categorized several visualizations which have font manipulation as their main differential (e.g. type visualization on macro-views, type visualization on lists and type visualization on texts).

2.1.2 Typeface Design

Letters have often been used in many kinds of visual communication forms because letters have their own visual impact and aesthetic. The adjustment of letters in an empty page or screen is the most basic design challenge. A good typography will greatly help a design to get the message across. It attracts the audience, and it is valued more [44].

Jacko [23] defines typeface as all the individual characters or glyphs at a given size: letter forms, punctuation, numbers, mathematical symbols, diacritical marks, and other accessory needed to fully compose a text. The typeface choice affects legibility and readability, the ability to easy see and understand what is on the page.

Samara [40] explains how important it is to select a specific typeface for a particular visualization. The most suitable typeface, the one that presents all the desirable features (weight, style, posture, width, ligature, etc) for your application. In order to choose typeface, it is helpful to look at the images that accompany the text, or to think about objects or places related to the subject matter of the text, as inspiration.

Hurst et al. [21] highlighted the increasing use of micro-typography in practical applications. Micro-typography is concerned with the low-level composition of text: (a) kerning, which is the visual different spaces between letters, (b) line breaking and (c) line justification, where the possible techniques are to use alternative ligatures and glyphs, scale the font, scale inter-word spacing and scale inter-character spacing.

Zapf [54] created a software known as *hz*-program based on Gutenberg's 42-line Bible; which was printed about 1455, and is a known masterpiece of art. Gutenberg achieved this perfection by using several characters with different widths, combined with ligatures and

abbreviations, in his lines. He finally needed 290 characters for the composition of the 42-line Bible. The *hz*-program could not work with Gutenberg's unusual ligatures and abbreviations to get economical typesetting today — people are not familiar with the abbreviations of the 15th century. The *hz*-program scales the letters, expanding or condensing, and also manipulates the kerning.

Typefaces are frequently designed to solve issues of legibility and readability created by a technology. A typeface made for online use can increase page legibility, as well as the overall perception of approachability, quality of an interface, and ultimately product acceptance [23].

Keeping in mind the importance of choosing the most appropriate typeface for specific applications [40, 44], and exploring features of micro-typography [21], we assume our project is related to the *hz* typesetting program [54]. However, our approach is closer to the Gutenberg's Bible, once we see in the data analysis of this project an opportunity to explore this domain. We hope that with the help of typeface experts it is possible to blend abbreviation and typographic techniques based on the data collected in this study, in other words, a data-driven approach for typeface design. However, even though we discuss and explore this approach, it does not belong to the scope of this project, but it is listed in our future work section.

2.2 Word Abbreviation Techniques

Considerable effort has been put into observing and understanding the cognitive strategies employed by humans when generating abbreviations for words. Much of this work has been aimed at the identification of the most suitable methods for generating abbreviations, mostly, for reducing the number of keystrokes needed to type in a command while maintaining comprehensibility and recall, or for use in automated tactical systems.

Hodge and Pennington [19] conducted a study in which they analyzed natural abbreviations of words generated by study participants and categorized these abbreviations into truncation, contraction and unclassifiable. They observed that there was hardly any agreement between the abbreviations generated by the participants and that this variation in the abbreviations increased with an increase in word length. Another trend observed in the generated abbreviations was that contraction was favored with uncommon and short words whereas truncation was favored with common and long words. A similar trend of word length playing a role in the choice of employed abbreviation technique has been observed in subsequent studies by Moses and Potash [30] and Streeter et al. [43]. For testing decoding of abbreviations, the most frequently generated abbreviation of words during the encoding phase along with information about word length and part of speech were provided to another

set of participants. It was observed that the participants were able to correctly decode 67% of the abbreviations with common words being much easier to decode than uncommon ones.

Nawrocki [33] conducted a study similar to Hodge and Pennington [19] that tested an abbreviation technique which drops the most common letters of the words, identified on the basis of the naturally generated abbreviations. He controlled for the length of the abbreviations generated by the techniques being tested and observed that there was no significant difference in decoding when compared to vowel deletion and truncation.

Streeter et al. [43] conducted a study similar to the one conducted by Hodge and Pennington [19] in which they analysed naturally generated abbreviations and identified vowel deletion and truncation to be the most suitable techniques for encoding monosyllabic and polysyllabic words respectively. For decoding of abbreviations, they observed that the proposed modified vowel deletion technique (i.e., drop all vowels except the one(s) occurring before the first consonant) gave the best results. However, the length of the abbreviations generated by the different techniques being tested was not controlled for and thus the results could have been due to the fact that the abbreviations generated by the modified vowel deletion technique were on average longer than the ones generated by the other two techniques. Another important observation reported by the study was that the preference for the abbreviation technique changes depending on whether the task at hand is decoding or encoding.

Hirsh-Pasek [18] proposed and tested the performance of phonics (generating abbreviations that when pronounced sound similar to the actual word) and minimum-to-distinguish (a form of truncation where a word retains as many letters as needed to uniquely identify the abbreviation) compared to simple truncation and vowel deletion for both encoding and decoding of abbreviations in a limited lexicon. Participants were provided extensive training prior to being tested and it was observed that more time was spent on training with vowel deletion and phonics compared to the other techniques. The results of this study indicate that simple truncation is the most suitable technique for encoding whereas vowel deletion and phonics performed the best when decoding. However, the results obtained could have been influenced by the learning effects.

A study by Moses and Potash [30] compared the performance of simple truncation, truncation with the second letter dropped, vowel deletion, military abbreviations — abbreviations used in the military systems — and dropping letters according to their frequency of occurrence in written English for encoding and decoding. It was observed that vowel deletion and truncation were the preferred techniques for generating abbreviations when encoding whereas abbreviations formed by truncation performed better than or as well as other abbreviation techniques when decoding. Another study by Moses et al. [29] , conducted with Naval Sonar

operators as participants, compared the performance of simple truncation, vowel deletion and military abbreviations with the result that truncation and military abbreviations were the preferred methods for encoding whereas no significant difference in performance amongst the tested techniques was observed with decoding. The length of generated abbreviations was controlled for in all the techniques with the exception of known military abbreviations. Ehrenreich and Porcu [11] tested the effect of providing information about the abbreviation technique to the participants on encoding and decoding abbreviations. They observed that there was no significant difference in decoding performance, confirming the observations made by a pilot study by Moses et al. [29].

In summary, the majority of these studies indicate that simple truncation performs better than or as well as other abbreviation techniques for encoding. However, there is little to no agreement on the method best suited for decoding an abbreviation. It is the task of decoding an abbreviation that is of central importance to our research. We also believe that when a word has a well known abbreviation (e.g. 'V.P.' for 'Vice President'), it would be preferred to an algorithm-made abbreviation.

The website Abbreviations.com [27] also provides a dictionary of acronyms and abbreviations. Their resource is built from scrapping large amounts of data. This resource is very useful if we opt for supplement our algorithm with the dictionary in case where the word has a standard known abbreviation.

2.3 Ranking Algorithms

PageRank by Page et al. [37] is the biggest example of ranking algorithm. Once it was the base of Google's[2] web page ranking system [24]. PageRank [37] is based on the reputation of the web page, which is calculated according on how many other pages are pointing at it, also considers these other pages reputation.

The PageRank algorithm is known as the random surfer model. We know another ranking algorithm known as the biased surfer model, the WordRank by Kritikopoulos et al. [25]. WordRank takes the PageRank algorithm and adds content similarity to it by calculating the page score considering the possibility of surfers selecting page 1 from page 2.

The patent by Oliver et al. [36] describes a method for adaptive text recommendation systems (ATRS). The recommendation is made based on a statistic measure of relevance of the text for the agent that requests. The set of recommended documents is constantly updated as more documents are added to the set of documents of interest. The ATRS analyzes the text in the new documents inserted in the interest set, categorizing them into clusters based

[2] http://google.com. Accessed 2017-04-25.

on the similarity of the documents content (similarity of the words). After clustering the ATRS extracts keywords, which are the words that are more frequent from each cluster. Then the ATRS filters the documents using application parameters, followed by calculating the relevance scores of the eligible documents and ranking them. The top scoring documents can be chosen by any user or application criteria.

If two documents have no words in common, then the similarity will be $similarity(D_1, D_2) = 0$. However if they have at least one word in common it will be:

$$similarity(D_1, D_2) = \frac{\sum_{w \in D_1 \cap D_2} count(w, D_1) count(w, D_2)}{[\sum_{w \in D_1 \cap D_2} count(w, D_1)^2]^{\frac{1}{2}} [\sum_{w \in D_1 \cap D_2} count(w, D_2)^2]^{\frac{1}{2}}}$$

where $count(w, D)$ denotes the number of occurrences of word w in document D, and $w \in D_1 \cap D_2$ denotes a word that is present in both documents.

For each Word w in a cluster C, calculate the frequency of the word w in the interest set, $Frequency(w)$; and calculate the frequency of the Word w in cluster C, $Frequency(w, C)$. The keyword score is calculated using the equation:

$$KeywordScore(w, C) = logFrequency(w, C) - logFrequency(w)$$

For each eligible document D, count the number of times the keyword $w \in keywords(C)$ appears. We can calculate the relevancy score of document D into cluster C, using the equation:

$$relevance(D, C) = \frac{\sum_{w \in keywords(C)} count(w, D)}{\sum_{w \in keywords(C)} [count(w, D)^2]^{\frac{1}{2}}}$$

where $w \in keywords(C)$ denotes one of the keywords in cluster C.

These page ranking algorithms [25, 37] are related to our work in terms of choosing the most relevant page from a set of pages. In our case, we are trying to decide the most relevant version of a word abbreviation from a set of collected word abbreviations for a specific word. Consequently, our ranking algorithms would be between the encoding and the decoding tasks (for more information see section 3.2.5). The ATRS [36] also relates to our work. ATRS ranks text documents based on their relevance, and we want to do the same but with words. The ATRS patent also claims that this algorithm can be easily applied for other types of data such as video and audio, besides text documents. So it teaches us that we can apply it to word abbreviations as well.

Chapter 3

Investigation and Evaluation of English Word Abbreviations

In this chapter, we report our pilot study design and results, we explain the tasks design, the database design, the adaptive algorithm and the participants used in our main experiment.

3.1 Pilot Studies

To guide our research we first ran a pilot study to identify the most common abbreviation techniques. For the pilot study, we considered words of length 10 or more, extracted from academic articles. We included words having known/common abbreviations, uncommon/domain specific abbreviations, similar prefixes but different suffixes, a vowel as the first character, and a selection of different parts of speech. Our first pilot study involved two tasks designed to help us understand how people abbreviate words. First, participants of the pilot study were given 25 randomly chosen words to encode with the condition that the abbreviations should not exceed 7 characters. Second, participants were asked to rank (in order of preference) the abbreviations of 50 randomly chosen words, generated using a variety of techniques from the literature and without controlling for the length of the generated abbreviations. On the basis of the observations from this pilot study, four abbreviation techniques, namely, simple truncation, vowel deletion, modified vowel deletion (drop all vowels while preserving the first three letters) and modified truncation (truncate but preserve term ending) were chosen to be evaluated further.

The second pilot study was based on the methodology of Wobbrock et al. [51] and aimed at verifying that heuristics identified from naturally generated abbreviations could be used as a basis for generating abbreviations of terms for use in visualizations. Participants

for the study were recruited by email and self-screened for English literacy and the study was conducted using an online survey. Each participant was asked to perform three tasks each involving 20 words selected using the same strategy as in the pilot study. We had 21 participants for the encoding task, 18 for the decoding task and 20 for the ranking task.

The first of the three tasks, the encoding task, required the participants to provide their own abbreviations for 20 terms with the condition that the abbreviation for any term should not exceed 7 characters. A sample question for this task would be: Provide an abbreviation (7 characters or less) for the given word: "acknowledgement".

The second task, the decoding task, required the participants to decode abbreviations of 20 terms generated using the vowel deletion technique (drop all vowels except the first letter). The decision to evaluate only the vowel deletion technique for decoding was based on the analysis of the results obtained from the pilot study and a literature review. A sample question for this task would be: Identify the original word for the given abbreviation: "admnstrtn".

In the third task, participants ranked 20 different sets of word abbreviations generated using simple truncation, vowel deletion, modified vowel deletion (drop all vowels while preserving the first three letters) and modified truncation (truncate but preserve term ending). Participants were instructed to rank the abbreviations in the order of preference from 1–4, with 1 being the most preferred and 4 the least. Lengths of the abbreviations generated for a term using the simple truncation, modified vowel deletion and modified truncation techniques, were restricted to be the same length as that of the abbreviations generated using vowel deletion. A sample question for this task would be: Rank the following abbreviations for 'universally' from 1 (best) – 4 (worst): "universa","unvrslly", "univrsll", "univers.y".

After the data was collected and analysed, we were able to report that for encoding, truncation turned out to be the most preferred abbreviation technique, whereas when ranking the abbreviations, those obtained using the vowel deletion technique were the most preferred. Responses from the decoding task also indicated problems with maintaining uniqueness of abbreviations. Interpretation of the abbreviation for a term seems to be affected by the construction of the word itself. Some words like "admnstrtn" were easy, where others like "frtfctns" were difficult. When the abbreviation for a term sounds like the term itself when spoken out loud, the decoding process seems to be simpler and it becomes more probable to get a correct response.

We learned from feedback on this study and from the study design that text abbreviations may not be applicable to tag clouds. The goal of tag clouds is easy skimming, and in the paper we used tag clouds as the main application for abbreviation. The words used in the experiment must be more carefully chosen in order to have a fair comparison among the execution of the encoding/decoding task. Another issue was the fact that we did not evaluate

the readability of the abbreviation in the context of visualization, we only did the decoding task by itself, where no context was given.

From what we learned in these pilot studies, we design a third study, where the words were more carefully chosen and the experiment setup was on a crowdsoucing platform, where it is easier to get a greater amount of participants.

3.2 Study Design

The study (see high level diagram in Figure 3.1) has two types of tasks (1) encoding and (2) decoding abbreviations, and was deployed in form of a web application hosted by our laboratory server. The full experiment had a total of 80 tasks in the following order: 40 tasks to create abbreviations (encoding) and 40 tasks to guess the English word that has originated the abbreviations (decoding).

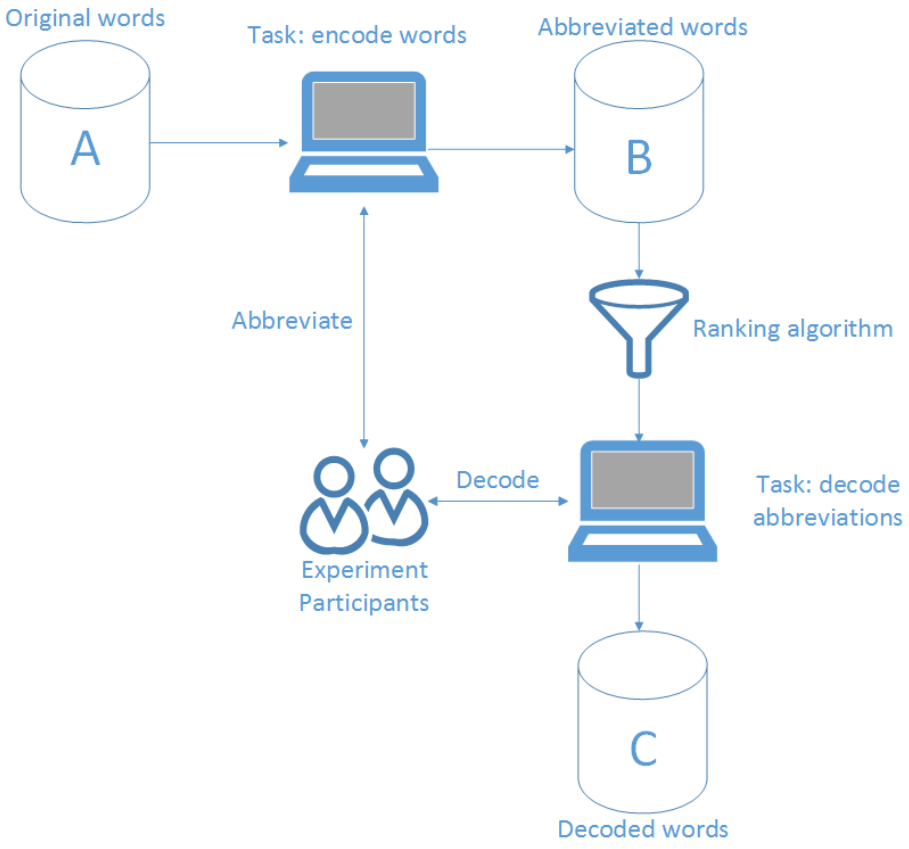

Fig. 3.1 Experiment design diagram.

The reason we kept the order of tasks as encoding first, then decoding is to keep the participants from learning any sort of abbreviation technique from the decoding task and having it to influence in their judgement when creating abbreviations in the encoding task.

There are tasks where human judgement, is needed and to solve this issue crowdsourcing computing is becoming more common for research studies [50, 55]. Crowdsourcing platforms are a socio-technical system that allow us to collect a considerable amount of data in a short period of time, and now we can embed other intelligent systems within the crowdsource platform to obtain faster human-computer type of computation [22]. This hybrid approach gave us the power we needed to design an experiment where we can collect and "smartly" evaluate abbreviations simultaneously, and saved us time and produce more relevant data. So instead of running two independent experiments where we first collect the encoded abbreviations, then tabulate the data and select the relevant set of abbreviations to be used in a second experiment to evaluate participants accuracy on decoding them. We simply use a recommendation system in between both experiments, which can ignore the least relevant abbreviations and feed the decoding task with the most relevant ones.

3.2.1 Task: Encoding Abbreviations

In this task participants were asked to create abbreviations (Figure 3.2) using the available space (no character restriction) for given words, while taking in to consideration that the created abbreviation should be understandable to most readers.

Also, for each word the participant had to choose a level of confidence regarding the their answer. The confidence levels varied from least confident "I don't think this word can be abbreviated", to neutral "It's my best guess", then to the most confident "I'm confident most people could understand it". When the participant declared that they did not think it could be abbreviated we allow them to leave the answer in blank, otherwise, the answer is required to proceed through the study.

Besides the collection of abbreviations, this task might contain a priming question regarding the context of words (described in Section 3.2.3).

3.2.2 Task: Decoding Abbreviations

In this task participants were asked to answer their best guess of what is the original word that originated the given abbreviations (Figure 3.3).

Also, for each abbreviation the participants had to choose a level of confidence regarding their answer. The confidence levels varied from least confident "I have no idea", to neutral "It's my best guess, but I'm not sure", then to the most confident "I'm confident that's the

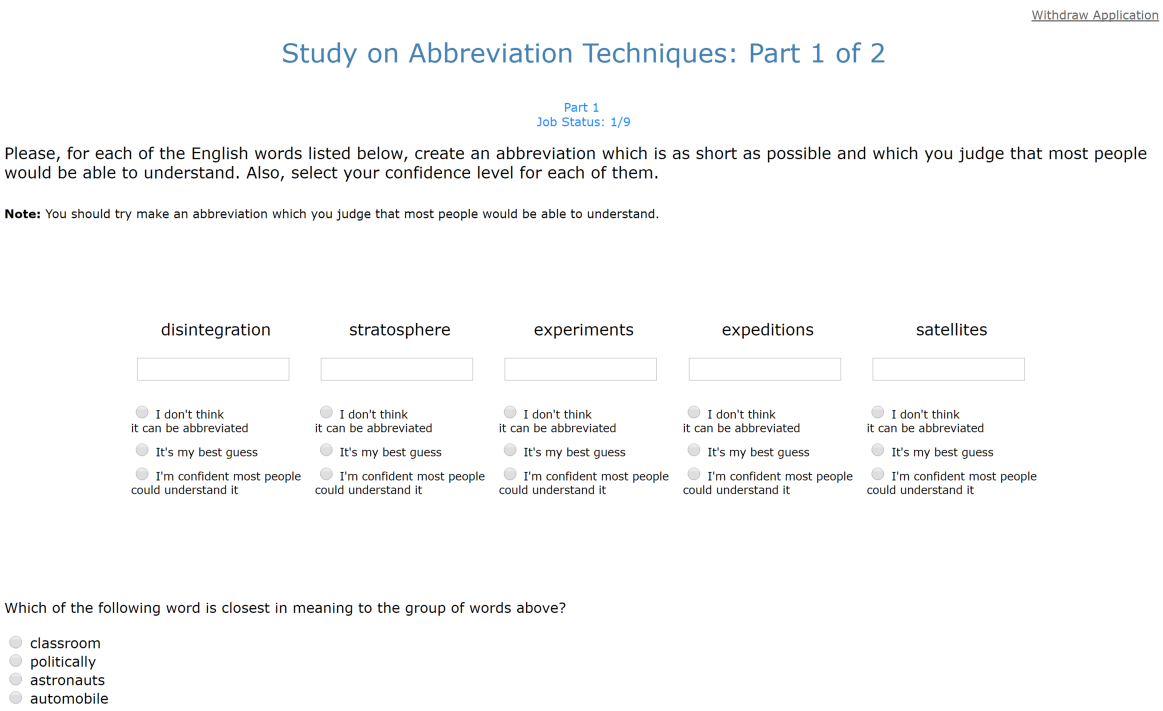

Fig. 3.2 Model screen for the encoding abbreviations task with context.

best answer". When the participant declared that they had no idea of what the answer was we allow them to leave the answer in blank, otherwise, the answer was required to proceed the study. In addition to that, we added an optional box for the participant to indicate they know their answer is correct, however they were not sure if they have spelt it right.

Besides the collection of decoded words, this task might contain a priming question regarding the context of words (described in Section 3.2.3).

3.2.3 Experiment Priming

We wanted to understand the effects of having semantic context when encoding or decoding abbreviations. In order to evaluate the context dependency we primed the participants for context in half of the tasks. Priming, means, we creating a priming question inserted in part of the tasks. The question is "Which of the following words is closest in meaning to the group of words above?"; then we list four options — each option represents one of our pre-defined contexts — as possible answers, but only one is correct and is semantically related to the words presented in either encoding or decoding task. This priming approach is based on Mohammad's [28] data collection for his Lexicon database that associates colour with word's semantic meaning.

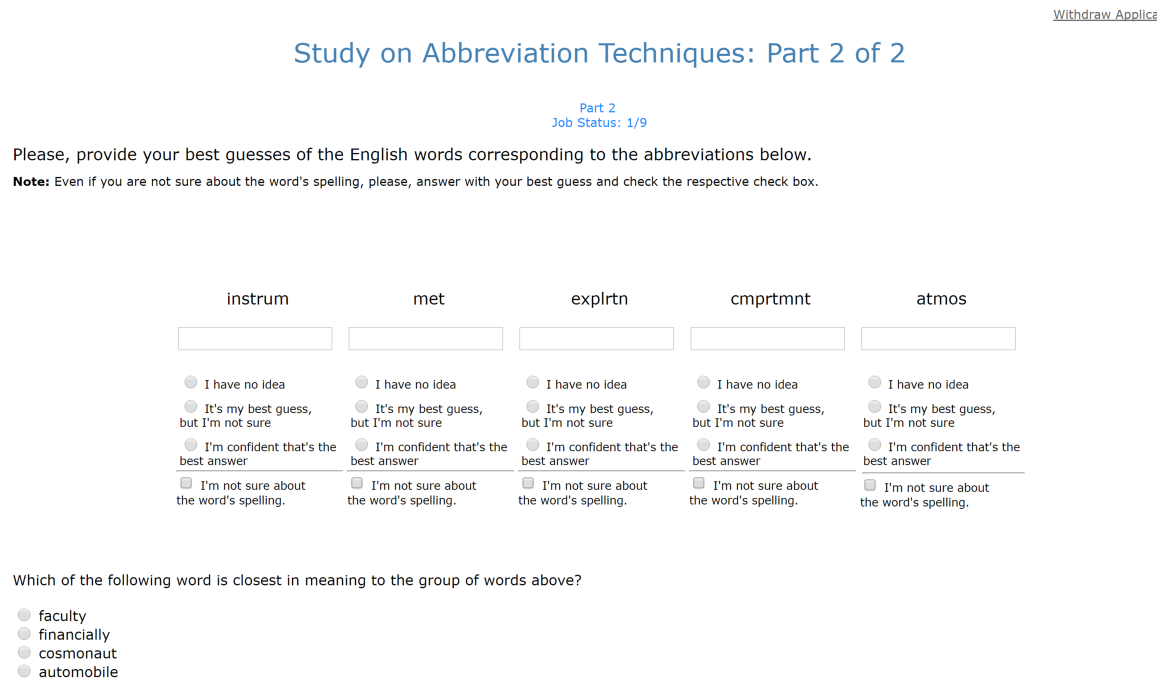

Fig. 3.3 Model screen for the decoding abbreviations task with context.

We assumed that when having a context participants would tend to worry about creating non-ambiguous abbreviations, or when decoding an abbreviation they would use the contextual information to maybe understand the abbreviation better.

The priming was applied to the two types of tasks in the same way. For each type of task, participants received 40 words to encode or decode. The words were divided equally in 2 sets of 20 words, one set is primed and the other is not.

At first, participants eitheror had all the 20 words in four groups of five words that are related through a common semantic context, after completing those, they are provided with the remaining 20 words in five groups of four words (the choice of words and how they were grouped is discussed further in this chapter), however these groups belong to different contexts. In other words, no context is defined, this was one scenario and the second scenario was where the participants received the set of words without context before the contextual set of words, in order to counter-balance the order of conditions (see Figure 3.4).

3.2.4 Word Database

From our previous pilot study on how people create abbreviations we learned that choosing the right type of words for the tasks is very important. So, this time we chose long words (10 characters or more) that are common in the English language using the 500K top most

Participant Starting WITH Context (80 words)			
Encoding Task (40 words)		**Decoding Task (40 words)**	
Contextualized	Non-contextualized	Contextualized	Non-contextualized
4 screens	5 screens	4 screens	5 screens
5 words (from same context)	4 words (one from each context)	5 words (from same context)	4 words (one from each context)
20 words	20 words	20 words	20 words
Participant Starting WITHOUT Context (80 words)			
Encoding Task (40 words)		**Decoding Task (40 words)**	
Non-contextualized	Contextualized	Non-contextualized	Contextualized
5 screens	4 screens	5 screens	4 screens
4 words (one from each context)	5 words (from same context)	4 words (one from each context)	5 words (from same context)
20 words	20 words	20 words	20 words

Fig. 3.4 The two different orders of the context effect counter-balanced.

frequent words in the English language by COCA [9] and in order to set context relations between words we used a Word2Vec API [46], which can give us a semantic similarity between words.

In order to create semantic groups, we had a two steps process where first we ran the Word2Vec API using the Google News model [15] on the words from the COCA database where we selected the top words being 10 characters or longer, resulting in around 5000 words. We created a triangular matrix with the semantic similarity of all the resulted words, in a total of 5000 x 5000 matrix.

As a second step, we arbitrarily chose four words that were semantically apart from each other and we used them as our contexts:

1. “astronaut”
2. “automobile”
3. “classroom”
4. “politically”

From those words we created four different word lists, each list contained 20 words that belong to the same context. In total the pool had 80 words (see Appendix A). These word lists were selected based on a threshold of similarity 0.2, where we found the words to be loosely related to the context. The words were semantically similar but not too close in the

spelling neither were synonyms of the context word. At this point, we hand picked 20 words out of the result. The same process was repeat for all four contexts.

Word Groups*			
Encoding Task		Decoding Task	
Group A	Group B	Group C	Group D
Group B	Group A	Group C	Group D
Group A	Group B	Group D	Group C
Group B	Group A	Group D	Group C
Group C	Group D	Group A	Group B
Group C	Group D	Group B	Group A
Group D	Group C	Group A	Group B
Group D	Group C	Group B	Group A

*each group has 20 words

Fig. 3.5 The 4 different groups of words were balanced in the experiment.

In order to balance the experiment regarding the words, we created four groups of words containing five words from each context resulting in eight different combinations as shown in Figure 3.5.

Consequentially, every participant was assigned with all the 80 words, however, the group words varied according to the order deployed to the participant's session.

3.2.5 Relevance Ranking Algorithm

In order to be able to have more relevant abbreviations decoded, which traditionally would mean running the encoding task as the first study, tabulating and analysing the data for us to be able to decide a trend on the abbreviations created. And then, those selected abbreviations could be used in a second study having people decoding them. Instead of that, we decided to substitute the human validation from this process by an algorithm that we designed to "intelligently" identify those trends and choose the abbreviations to be decoded without human interference. The intelligence of this algorithm is given by a formula we designed using factors we judge to be important when choosing relevant abbreviation to be decoded. Further in this section we discuss what are those factors and how we designed and implement this algorithm.

Consequentially, the word abbreviation used in the decode task was selected from the pool of abbreviations created by participants in the encode task and generated from three different abbreviation techniques we used in our pilot study: drop vowels, truncation and truncation while keeping the end (see examples in Figure 3.6). We are adding the abbreviations generated from our pre-defined techniques to solve the problem of the system's cold start, which is when the first participants start the decoding task and the algorithm does not have enough information for selecting relevant abbreviations.

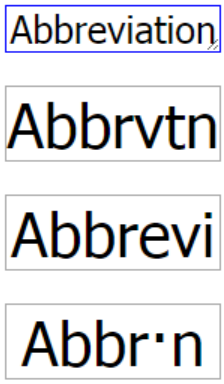

Fig. 3.6 The word “Abbreviation” and its abbreviation versions for the following techniques: drop vowels, truncation and truncation while keeping the end.

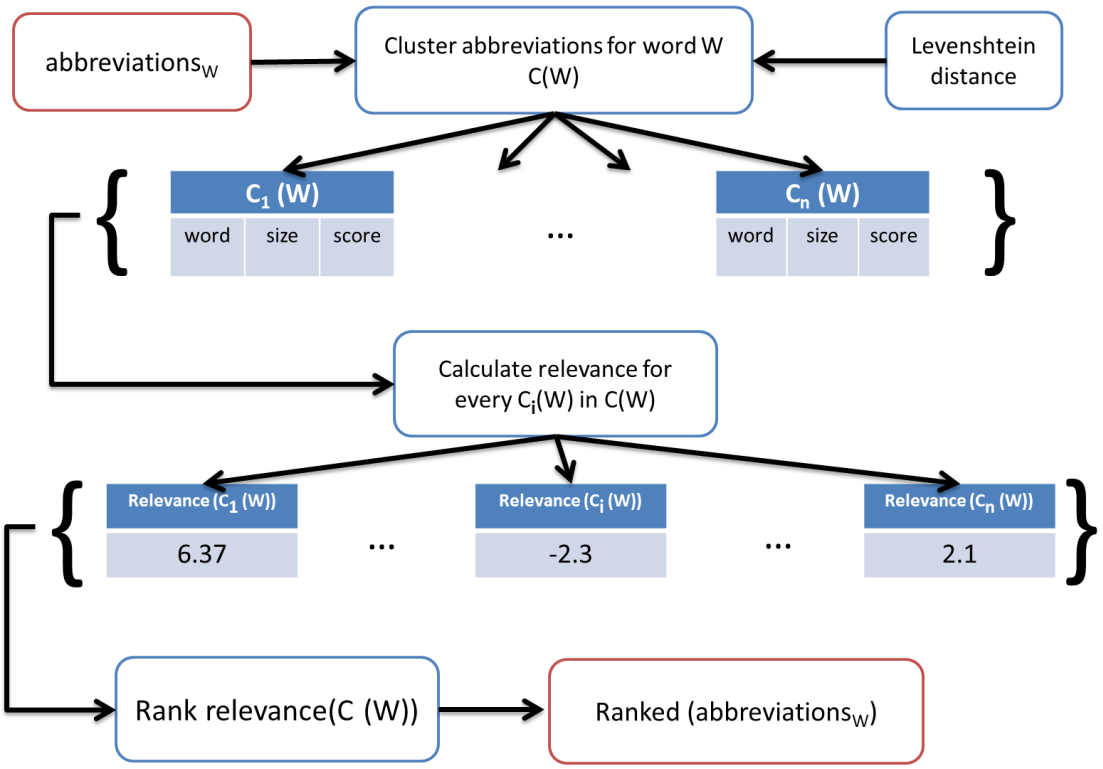

Fig. 3.7 Relevance ranking algorithm and how it works for ranking the abbreviations of a word W ($abbreviations_W$).

The overview of our relevance ranking algorithm can be seen in Figure 3.7. At first, we create clusters from $abbreviations_W$, which is the list of word abbreviations created for a word W, using a library [12] that classifies based on k-means algorithm [35]. The cluster $C(W)$ is calculated based on the Levenshtein distance score, which distinguishes different

words and takes the order of segments in a word into account [17]. To generate the scores we use the Levenshtein distance API [6] among the abbreviations in $abbreviations_W$. Generating a vector for each abbreviation in $abbreviations_W$. These vectors are then clustered by the cluster API [12] and will result in a collection of clusters $C(W)$. Each cluster $C_i(W)$ is an object that contains the set of properties:

$$C_i(W) = [word, size, score]$$

where *size* is the number of abbreviations in this cluster, and *score* is the average confidence for the abbreviations in the cluster that have been decoded already; *word* is the abbreviation that represents the $C_i(W)$ when the relevance of the cluster is calculated. The chosen *word* has a 55% probability (arbitrary variable) of being the abbreviation most similar to the original word W, otherwise it will be a random pick within $abbreviations_W$. We have chosen to add randomness to the algorithm so we can try to balance it for bias towards picking the same words within a cluster, because even though the words belong to the same cluster are very similar among themselves, we might want to vary choosing the representative of the cluster considering a small change in a choice of a letter to drop could influence the accuracy positively.

Afterwards, we have to measure the relevance of each cluster C_i in $C(W)$ by using the Levenshtein distance of $C_i(W)$ and word W, the amount of words inside the cluster $C_i(W)$ and in the case where this cluster has already been decoded before, the decoding confidence will also be considered. The Levenshtein distance returns the difference from 0 to word length. Where 0 is the same word and word length is the most different. Considering that we multiple it by -1, having the most negative as the most different as well. The confidence, if existent will vary from 0.0 to 2.0 and the relevance calculation is given by

$$relevance(C_i(W)) = [-1 * levenshtein(C_i(W).word, W) * C_i(W).score] + C_i(W).size + a$$

however, if it is the case where the cluster has not been decoded yet, then relevance is

$$relevance(C_i(W)) = [-1 * levenshtein(C_i(W).word, W)] + C_i(W).size + a$$

where in both cases a varies from 0 to 10, with a probability of 65% of being 0. The purpose of a is to add a random factor into the algorithm calculation to occasionally allow a cluster that does not have a good score to be chosen.

The set of relevant abbreviations for a word W is represented by

$$abbreviationRank_W(abbreviations_W) = rank[relevance(C_i(W))]$$

At last the algorithm is going to return the top 1 $abbreviationRank_W(abbreviations_W)$, allowing the study application to display it for the participant, who will now try to decode this abbreviation.

3.2.6 Participants

We recruited a total of 105 participants via CrowdFlower[1], which is a crowdsourcing platform to recruit web workers. CrowdFlower listed our study for their users, who then had access to the study description and instructions. If interested on the participating the user would have access to our web application after agreeing with the consent form. Users who participated on the study were recompensed $4.00 to complete the tasks.

Even though we did not take demographics of the participants, they were over 18 years old as speficfied by the CrowdFlower's agreement with its workers. As we displayed only English words and abbreviations, we preffered native English speakers so that language was not an extra barrier. However, CrowdFlower does not allow selection of participants by language proficiency, so we did our best to screen participants by setting up CrowdFlower to list our study only for participants from English speaking countries. The language proficiency criteria was also listed in the study description as well as in the consent form, and we relied on participants best intentions to not attempt the study otherwise.

[1] https://crowdflower.com. Accessed 2017-04/25.

Chapter 4

Study Results

In this chapter, we show our study results, methodologies used to analyse the data, as well as a brief discussion of the results and takeaways.

4.1 Data Cleaning

As mentioned in the previous chapter, we ran the study with 105 participants. However, we first piloted the study with five participants, which allowed us to detect and fix some implementation issues on the experiment platform. We noticed a number of participants that left many empty answers, so we omitted all participants who left more empty answers than the average of empty spaces plus two times the standard deviation of empty spaces. Furthermore, as mentioned on Chapter 3, we were also going to eliminate participants whose priming question accuracy was below the average priming accuracy. However, those who fit in this category were already eliminated by the empty space constraint. Finally, the data analysis was performed on the total of 85 participants' data.

4.2 Context Priming and Confidence Level

The primed tasks were balanced, as half of the participants started the experiment with primed tasks and the other half without, we also called them contextualized tasks and non-contextualized questions.

4.2.1 Priming Accuracy

The participants showed to be good at determining the context in the priming questions, with accuracy rate of 87.5%.

In Table 4.1 we can observe the priming accuracy rate between decoding and encoding tasks.

Table 4.1 The priming accuracy rate between decoding and encoding tasks.

Task		Percent (%)
Encoding	**Correct**	86.7
	Incorrect	13.3
Decoding	**Correct**	88.1
	Incorrect	11.9

4.2.2 Average Confidence

The confidence level was reported from 0 to 2, where 0 is the lowest and 2 is the highest. The scale for encoding: (0)"I don't think this word can be abbreviated", (1)"It's my best guess" and (2) "I'm confident most people could understand it"; and scale for decoding: (0)"I have no idea", (1)"It's my best guess, but I'm not sure", and (2) "I'm confident that's the best answer".

The average confidence level for the encoding tasks is 1.19 with a standard deviation of 0.56; and the average confidence for decoding tasks is 1.40 with a standard deviation of 0.59. We ran the paired sample statistical test that showed that the difference between the average confidence between encoding and decoding tasks is statistically relevant $t(1896) = 11.359$ and p<0.05.

We decided to investigate the average confidence of encoding and decoding tasks between contextual and non-contextual tasks. We found no statistical difference on the confidence level from the encoding task between contextual and non-contextual tasks ($t(482) = -1.172$ and p=0.242 (>0.05)). However, we found a statistical difference between contextual and non-contextual task on the decoding task confidence level as shown on Table 4.2 ($t(482) = 3.082$ and p<0.05).

Table 4.2 Average confidence for decoding task for contextual and non-contextual tasks.

	Decoding Confidence		
	Mean	**Std. Deviation**	**Std. Error Mean**
Context	1.4702	.57021	.02595
Non-Context	1.3532	.62999	.02867

In the decoding task, besides the confidence level we also had an option for participants to indicate independently of their confidence when they were not sure about the spelling of

the answer. We figured it would important to tell participants that even though they might not be sure of how to spell the answer, they could still give it a try. However, in the data analysis, we observed that very few participants (around 5% of answers) used this option.

4.3 Encoding Task

The average length of the encoded abbreviations is 6.7 characters long and standard deviation of 2.1 characters. It varied between 2 and 15 characters. The original words to be encoded were 10 to 15 characters long, and average length of 11.9 characters.

In the heat map visualization from Figure 4.1 we can see the probability of dropping a letter based on our study data. We can also notice the correlation of the most frequent letter in English having a high number, as well as vowels.

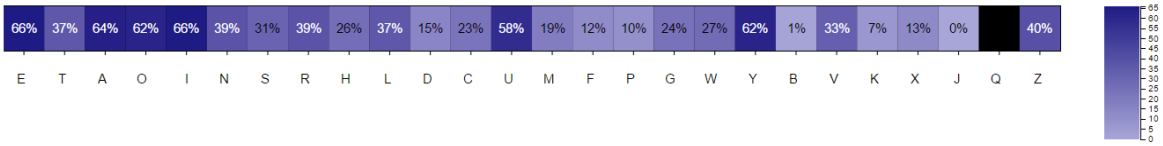

Fig. 4.1 Heat map visualization for the probability of dropping a letter from study data ordered by letter frequency in English [26]. The black cell means that the letter is not present in our study database, thus we has no data to display.

To understand how letters are dropped more deeply, we created a matrix visualization where we can see contextual dependency for instance, the probability of dropping a letter *A*, when it is position after a letter *B* (as shown in Figure 4.2). The letter in the rows represents the first letter followed by the letter in the column. In the right side, we can see a panel that presents more information about the selected digraph, including its frequency in English, and a small bar chart of the accuracy categories (this data is explained further on Section 4.4) associated to this digraphs. In this picture, we have **IE** selected, and black cells are digraph that are not present in our database.

In order to further understand the encoded abbreviations and also inspired by typography design, we created a visualization where each letter of the original word had its font size rendered proportionally to how frequent it was kept. For instance, in Figure 4.3, the first word “comprehensive” has its 4 first letters “comp” much bigger than the following ones. This shows that most of the created abbreviations for the word “comprehensive” drop the other letters many more times compared to the letters “comp”. Beside the abbreviation itself, we have the original word with all the letters in the maximum font size.

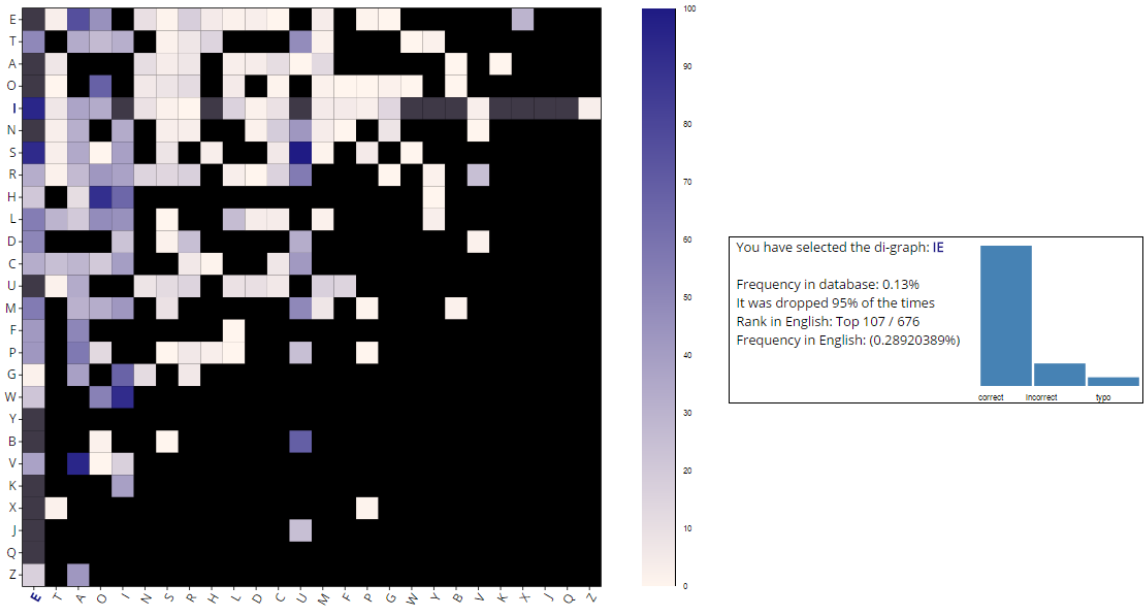

Fig. 4.2 Matrix heat map visualization for the probability of dropping a letter from a digraph, ordered by letter frequency in English [26]. In the miniature bar chart, we can see the proportion of the decoded words, where *E* was dropped after *I*, that were incorrect, typo or correct.

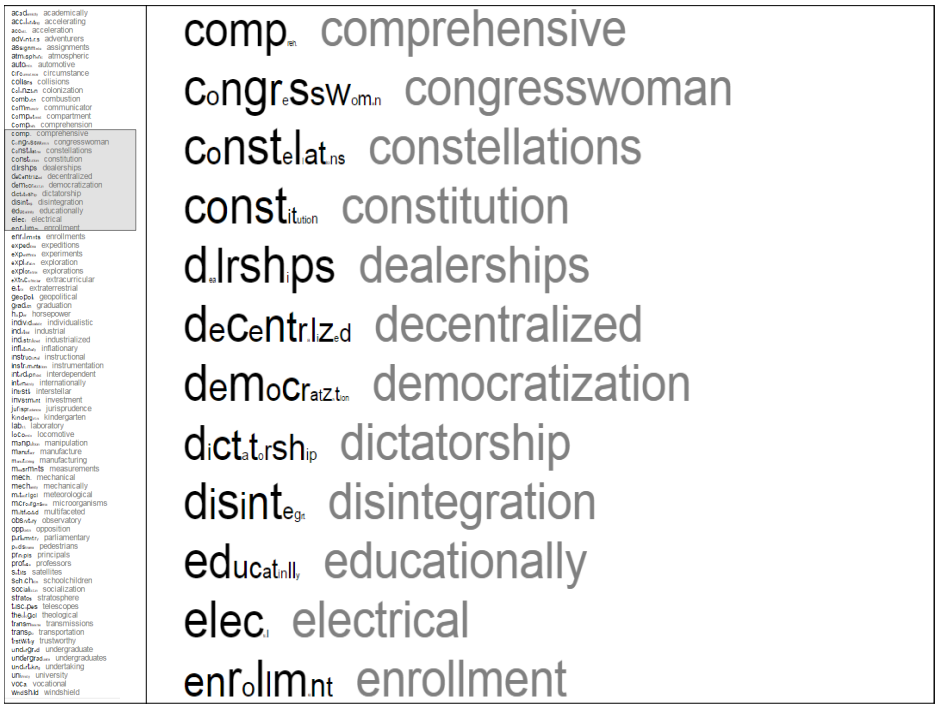

Fig. 4.3 Encoded words visualization thumbnail showing letters that were kept and dropped.

We decided that would be interesting to be able to visualize the words stressed syllables — stressed syllables are given by the syllable with most emphasis when pronouncing a word's phonemes. We were trying to understand if a letter from a stressed syllable was kept over others. It seems like the stressed syllable is usually kept, but its vowels tend to be dropped (see Figure 4.4).

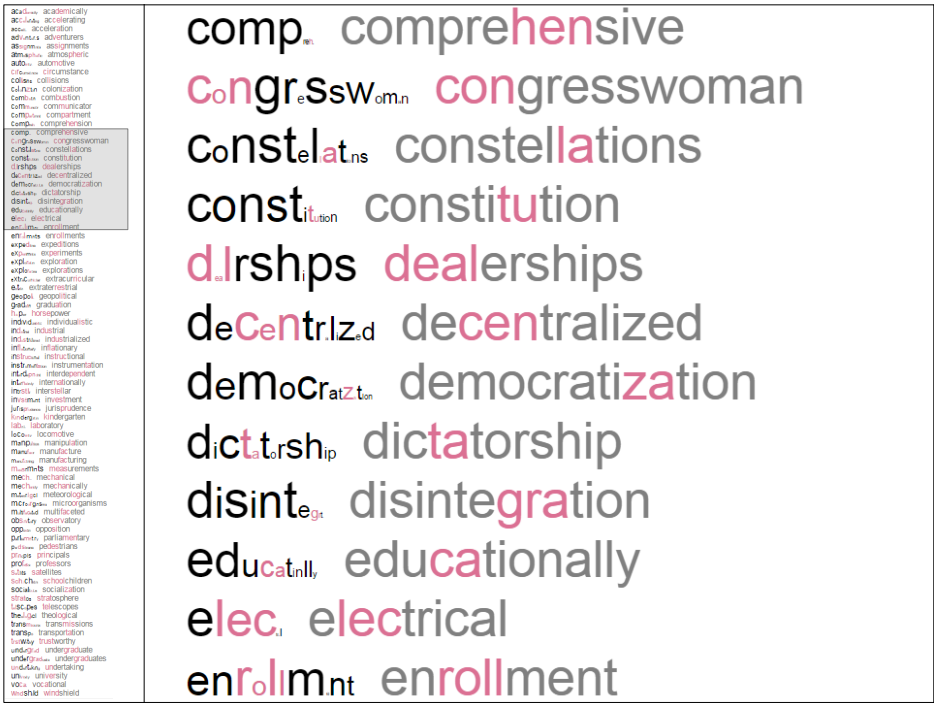

Fig. 4.4 Encoded words visualization thumbnail showing the letter that were kept and dropped, as well as their stressed syllables.

In Figure 4.5, we can observe a chart that shows the probability of keeping a letter in relation to its position within the word. In this chart we can observe the "U" like shape, which recalls the word recognition literature that mentions this "U" shape when describing that the first and last letter are the most important letters for people to recognize a word [3]. We generate this same chart for all the length we have in out study (10–15 characters) and for all the length we can observe the same "U" shape.

4.4 Decoding Task

In order to analyze the decoded words (abbreviation's interpretations), we went by finding the Levenshtein distance between the original word (right answer) and the decoded word answered. We found that 39% of the decoded words were perfectly matched with a Levenshtein distance of 0 (see Figure 4.6 for the full distribution).

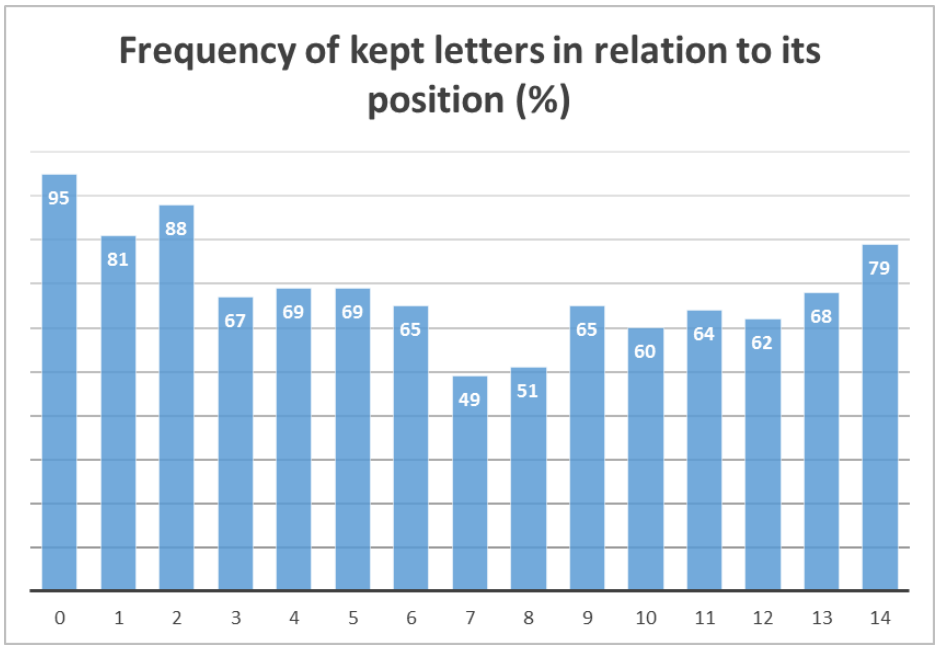

Fig. 4.5 The distribution of the probability of keeping a letter in relation to its position within a word.

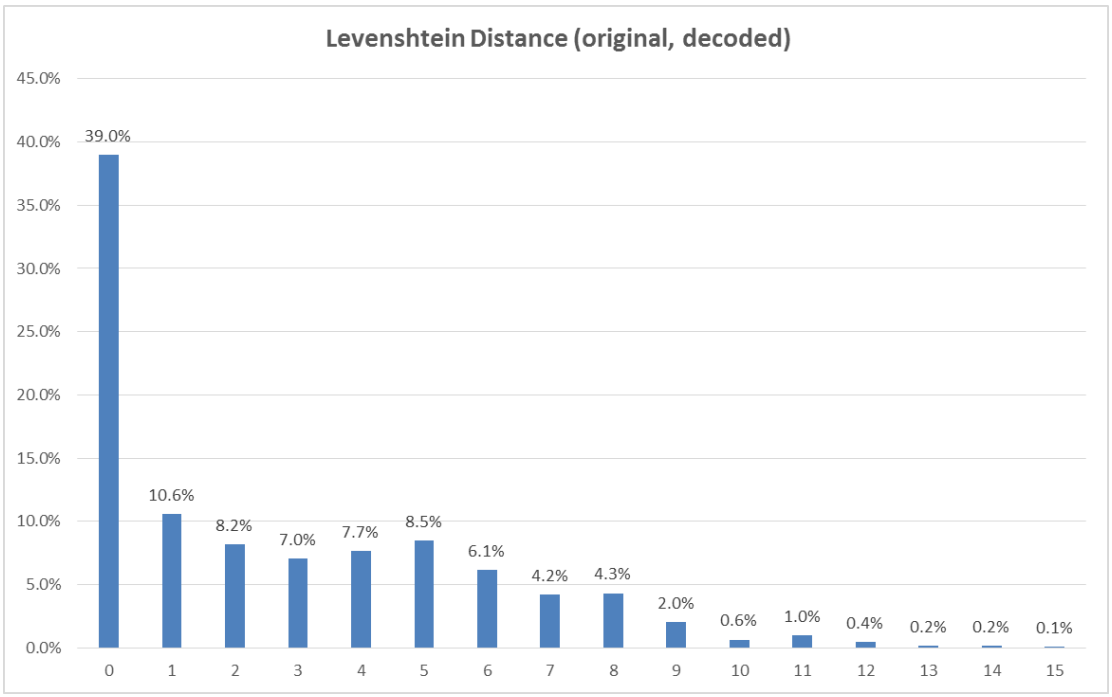

Fig. 4.6 The distribution of decoded words grouped by their Levenshtein distance with the original word.

Thus, to identify how "wrong" the decoded words with Levenshtein distance different from 0 were, we manually categorized the answers into the following categories:

correct: perfectly matches. Example: "abbreviation" from "abbrtn" (answer: "abbreviation").

typo: matches but has a typo. Example: "abbrevition" from "abbrtn" (answer: "abbreviation").

plural: its the correct answer either in its plural or singular form. Example: "abbreviations" from "abbrtn" (answer: "abbreviation").

semantic: has same semantic meaning as the original word. Example: "abbreviating" from "abbrtn" (answer: "abbreviation").

false: answer fits into given abbreviation but its not the original word, nor it is semantically related. Example: "universe" from "univ" (answer: "university").

synonym: its a synonym of the original word. Example: "car" from "auto" (answer: "automotive").

mislead: when the answer matches the abbreviation, however, the abbreviation does not follow the original word. Example: "button" from "btn" (answer: "abbreviation").

incorrect: answer is either a copy of the given abbreviation or does not fit into it. Example: "abbrtn" from "abbrtn" (answer: "abbreviation").

We can also find combination of those tags. For instance, the original word "expeditions", with the decoded word "expidition" and distance of 2; could be categorized as **typo** and **plural**.

In the Figure 4.7 we can see the distribution of decoded words with the accuracy categories by Levenshtein distance.

Considering the motivation of this study was to create abbreviations for data visualizations, we decided that besides the "correct" category we would also consider "plural", "typo", "synonym" and "semantic" as correct answers. It means the participant understood the meaning of the word, and this is what we need for visualizations, which are usually contextualized. That said, we can observe that those categories are mainly present up to Levenshtein distance of 2. All the other categories are considered incorrect, because we do not want people to be mislead from the visualization context when reading an abbreviation. The end result for the accuracy of the decoded words is 79% with standard deviation of 0.095.

In Table 4.3 we can see the difference between the accuracy rate between context and non-context for the decoding task. We ran a statistical t-test ($t(77) = 3.122$ and p<0.005) that showed the accuracy rate between context and non-context tasks are significantly different.

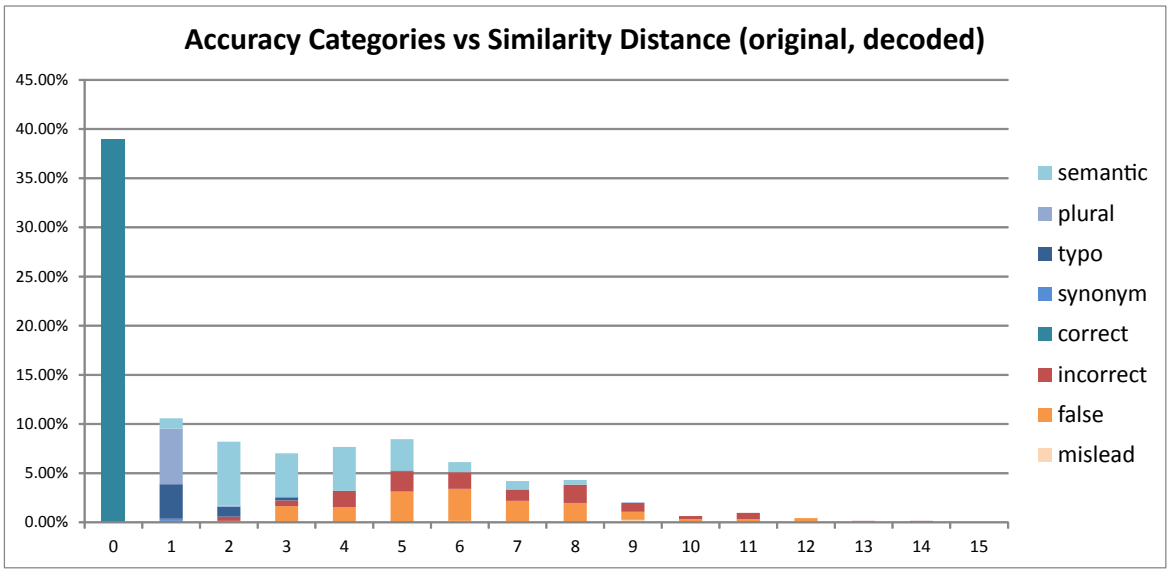

Fig. 4.7 The proportion of accuracy categories in relation to the Levenshtein distance between the answer and the original word.

This confirms our hypothesis that the context can significantly improve the participants' performance when decoding an abbreviation.

Table 4.3 The accuracy rate for context and non-context decoded words.

Accuracy by Context Dependency

	Mean	**Std. Deviation**	**Std. Error Mean**
Context	.815	.090	.010
Non-Context	.772	.095	.011

4.5 Discussion

We found a significant difference in the confidence levels between encoding and decoding tasks (higher) this tells us people are more comfortable decoding abbreviations. We can speculate that this occurs because when participants are guessing a word from an abbreviation it might feel like they are aiming for a more defined answer. On the other hand, when creating an abbreviation where they do not have many restrictions for invention, they do not know what to expect as right or wrong; which in our point of view seems normal. Furthermore, we found an effect on the confidence level between contextual and non-contextual for decoding tasks. This effect combined with the effect found on the accuracy rate from the decoding task can confirm our hypothesis that having an abbreviation within a semantic context can help

people to decoded abbreviations. This is positive in our case, considering data visualizations in general present data that are connected within semantic context (e.g., a visualization comparing bank's performance, all the words are related to finances and bank names).

The encoding task data provided us with some interesting numbers regarding the frequency of dropped letters, as monographes or even digraphes (Figures 4.1 and 4.8). This information along with the dropping probability in relation to the the letter's position (Figure 4.5) is crucial knowledge for us to understand the way people decide to keep and drop letters within a word. To explore this information further, we decided to visualize the dropping probability for digraph but scaled to the frequency of the digraph in the study database (Figure 4.8). This normalization fixes cases where a digraph appears to be dropped many times but we only have very few occurrences of it. For intance, **TI** is standing out for having high frequency in our database and in the same time it was dropped very frequently. This measure combined with the accuracy rate from the decoding phase is one of the main components of our abbreviation algorithm (see Chapter 5).

In the future we will collect more data which will allow us to extend the accuracy of our abbreviation algorithm. Considering that our approach to abbreviate is data driven, the more data we can obtain to calculate the dropping probability of letters, digraphs or even trigraphs, and position related, the better our algorithm will be. There are many other factors we could consider, such as nouns, verbs, analyze the root of words, etc.

The accuracy data compiled from the decoding task helped us to better understand how accurate we need people to be while decoding an abbreviation, by this we mean, we do not need people to decode the very same word from an abbreviation, but we do need them to be able to realize the contextual meaning the abbreviation brings.

The visualizations that we created for visualizing the abbreviation along with the dropped/kept letters (Figures 4.4 and 5.2) will be useful for typography design.

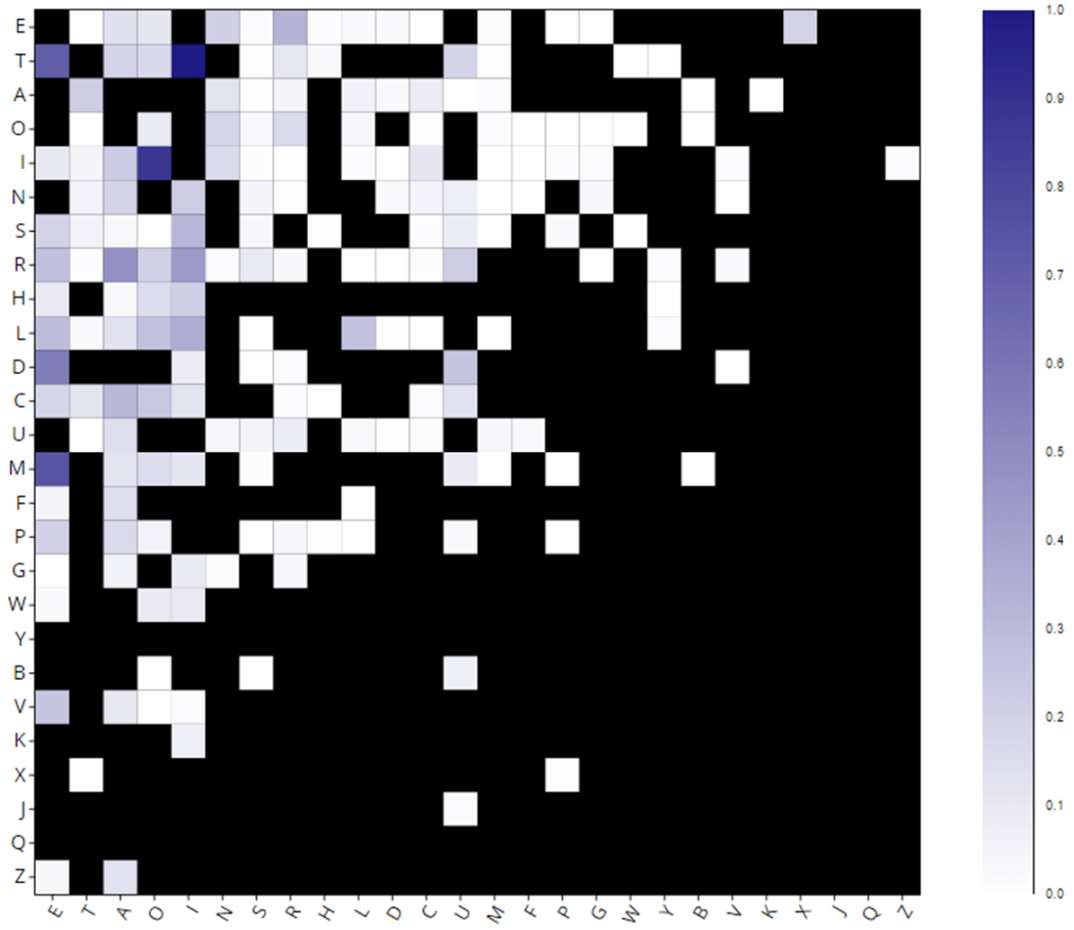

Fig. 4.8 Matrix visualization of digraph dropping probability normalized by digraph frequency in database.

Chapter 5

Abbreviation on Demand Algorithm

This chapter describes the design and implementation of our "Abbreviation on Demand" algorithm, which aims to drop as many letters as needed to abbreviate a word. This is our proposed solution for long text labels on data visualization. In addition to that, this solution is generic enough to be applied to short tweets.

5.1 Algorithm Design

The algorithm design is based on our study data. It is composed of two main parts a correlation measure of digraphs, and the probability of dropping a letter in relation to its position in a the word.

5.1.1 Correlation Matrix

We built a correlation matrix (see Figure 5.1) from elements of *char* x *char*, where *char* is the alphabet letters ordered by the English frequency. In other words, *char* has the alphabet order by the most frequent letter in the English language [26].

char = ["E","T","A","O","I","N","S","R","H","L","D","C","U","M","F","P","G","W", "Y","B","V","K","X","J","Q","Z"]

The $correlationMatrix$ measure is calculated based on three main elements. First, the $probDrop(char_i, char_j)$, which is the probability of a letter $char_j$ being dropped when it follows the letter $char_i$. Second, the $studyFreq(char_i, char_j)$ is how frequently the digraph $char_i char_j$ appears in the study data (encoded words). Third, the $studyAccuracy(char_i, char_j)$, which is the decoding accuracy of abbreviations where the digraph $char_i, char_j$ has been dropped at least once.

$$dropProb(char_i, char_j) = \frac{\sum dropped(char_j after char_i)}{\sum(char_i char_j)}$$

$$studyFreq(char_i, char_j) = \frac{\sum(char_i char_j)}{\sum_{i=0}^{n}[\sum_{j=0}^{n}(char_i char_j)]}$$

$$studyAccuracy(char_i, char_j) = \frac{\sum correctDropped(char_i char_j)}{\sum[correctDropped(char_i char_j) + incorrecDropped(char_i char_j)]}$$

$$correlationMatrix_{i,j} = probDrop(char_i, char_j) * studyFreq(char_i, char_j) * studyAccuracy(char_i, char_j)$$

The *correlationMatrix* measure is also normalized.

As an example of this equation applied is as follows

$$correlationMatrix_{0,1} = probDrop(\text{“}ET\text{”}) * studyFreq(\text{“}ET\text{”}) * studyAccuracy(\text{“}ET\text{”})$$

So, in order to calculate $correlationMatrix_{0,1}$, we have the probability of the letter “T” being dropped after a letter “E” ($probDrop(\text{“}ET\text{”})$), followed by how frequent the digraph “ET” is in our study database ($studyFreq(char_i, char_j)$); and how accurate the decoded abbreviations were when the digraphs “ET” was dropped ($studyAccuracy(char_i, char_j)$).

5.1.2 Probability of Dropping a Letter Based on Position

As discussed in the Chapter 4, from the encoded data we were able to get the probability of a letter being kept from a word, based on the letter's position within the word. So, in order to generalize this data we scale the probability found in the study which had a maximum length of 15 (index from 0 to 14) to word of length n (index from 0 to n-1).

mathtools

$$scale(i) = \lceil (n-1)/14 * (i-14) + (n-1) \rceil$$

where i is the index position from the original probability, so in this case it varies from 0 to 14. The $scale(i)$ will result in a new index from 0 to n-1. Afterwards, we round $scale(i)$ up to obtain an integer index. At this point, we vary in the index s in the new scale from 0 to n-1 so that, for every s $scaledProbability(s) = average(probability(i))$ where $scale(i) = s$.

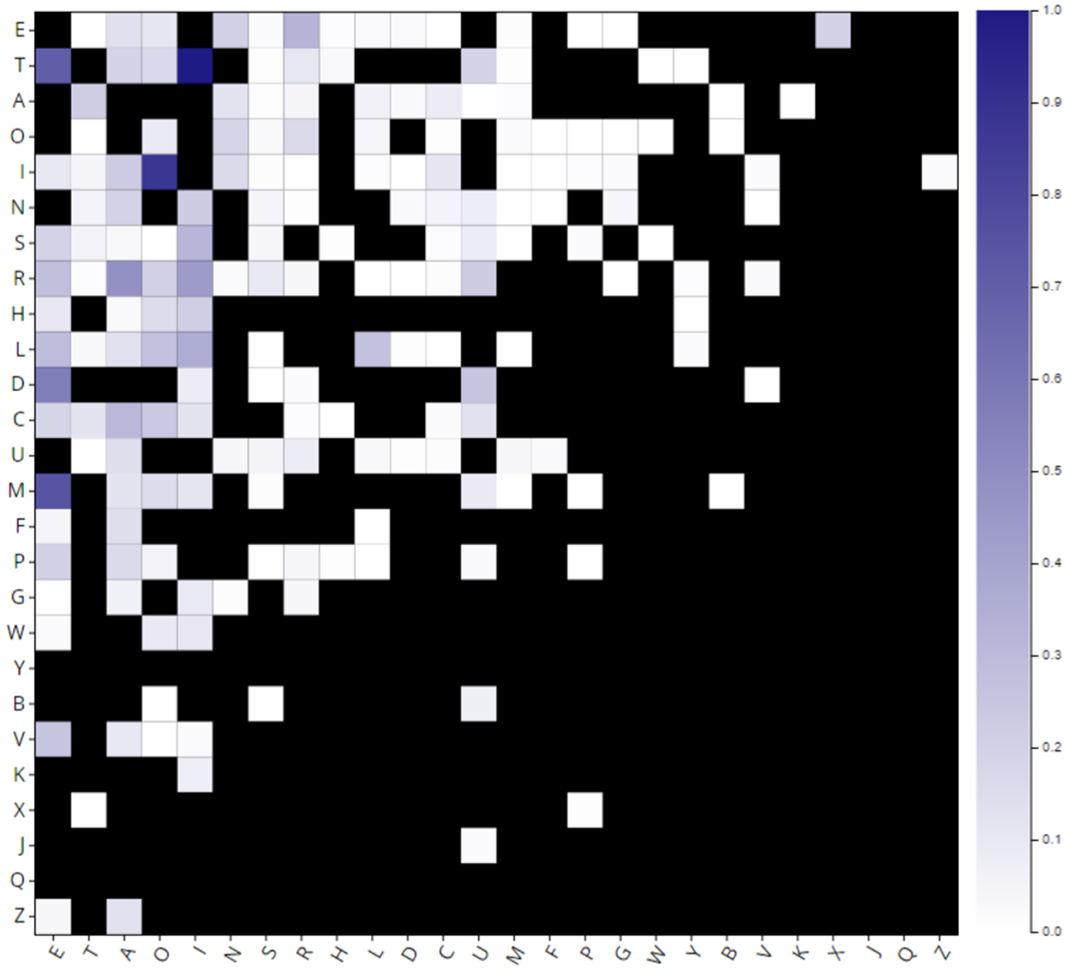

Fig. 5.1 Correlation matrix visualization. Black data points are digraphs that are not present in our study database.

See the example in Figure 5.2, where we are scaling the probability to a 10 letter word. Notice that we have the probability of keeping the letter in the word, so in order to find the the probability of dropping the letter, which would be more useful in our case, we just need to calculate:

$$probPositionDrop(s) = 100 - scaledProbability(s)$$

5.1.3 The "Abbreviation on Demand" algorithm

Inspired by mostly the drop vowels abbreviation technique, we decided deleting/dropping letters from a word is the best general approach. However we noticed that just dropping the

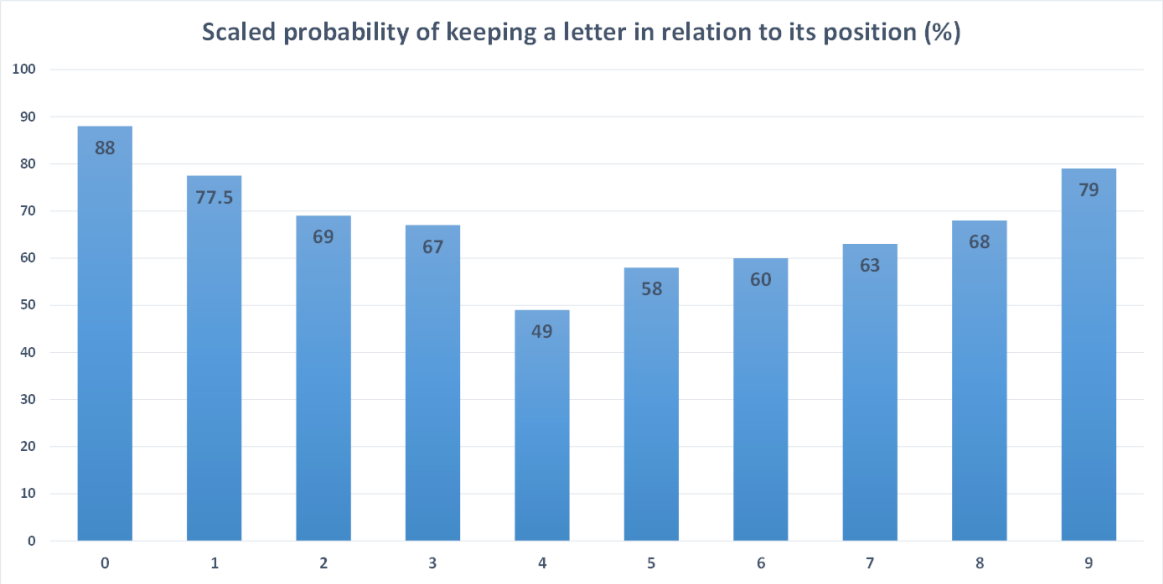

Fig. 5.2 The scaled probability of keeping a letter based on its position for a 10 letter word.

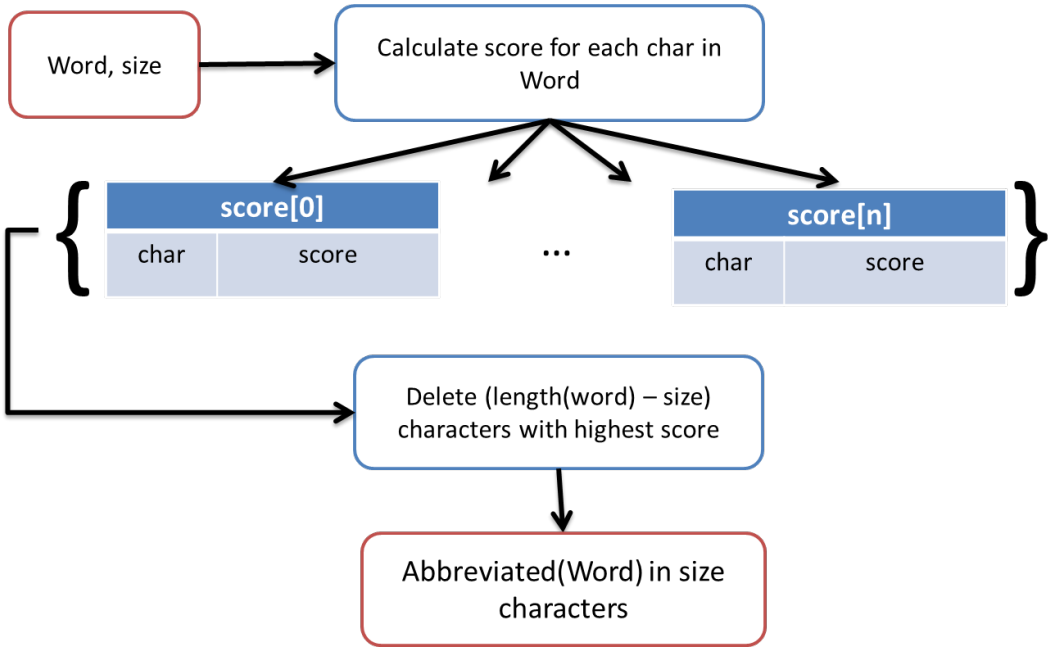

Fig. 5.3 How the abbreviation on demand works for a given *word* and the desired new length (*size*).

vowels might not be good enough, so we created the "Abbreviation on Demand" algorithm that given a word to be abbreviated in a specific size, the algorithm will drop the least important letter (many times it ends up being a vowel) considering a score calculated with its

position and correlation measure until the abbreviation's length in characters matches the specified size (see Figure 5.3).

For the score calculation we use the correlation measure given by $correlationMatrix(word[i-1]word[i])$ and the $probPositionDrop(i)$ probability of dropping a letter depending on its position. In a word, the score for each letter in position i from 0 to the word length is given by:

i>0

$$score_{word[i]} = correlationMatrix[word[i-1][word[i]] * probPositionDrop(i)$$

i=0

$$score_{word[i]} = monographDropProb(word[i]) * probPositionDrop(i)$$

where the $monographDropProb(word[i])$ 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 can not 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 the Figure 5.4 we present a modified version of our algorithm implementation 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 parametrized 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.

We ran our algorithm on the words from the study database side by side with our techniques from the literature: drop vowels, truncation and truncation while keeping the end. We also listed the top three most accurately decoded abbreviations created by participants in our study. A sample of those appear in Figure 5.5. The full list with the 80 words can be found in the Appendix B. We have found our algorithm to perform well: in the word "academically" abbreviates to "acdmcly" compared to the other techniques it is more intuitive. However, in some cases like for the word "automotive" we abbreviated it to "autmtv", which does not seem to be intuitive. Although we wish to have more data to improve the algorithm results, even without a formal evaluation we observed the current algorithm results and found it to perform well.

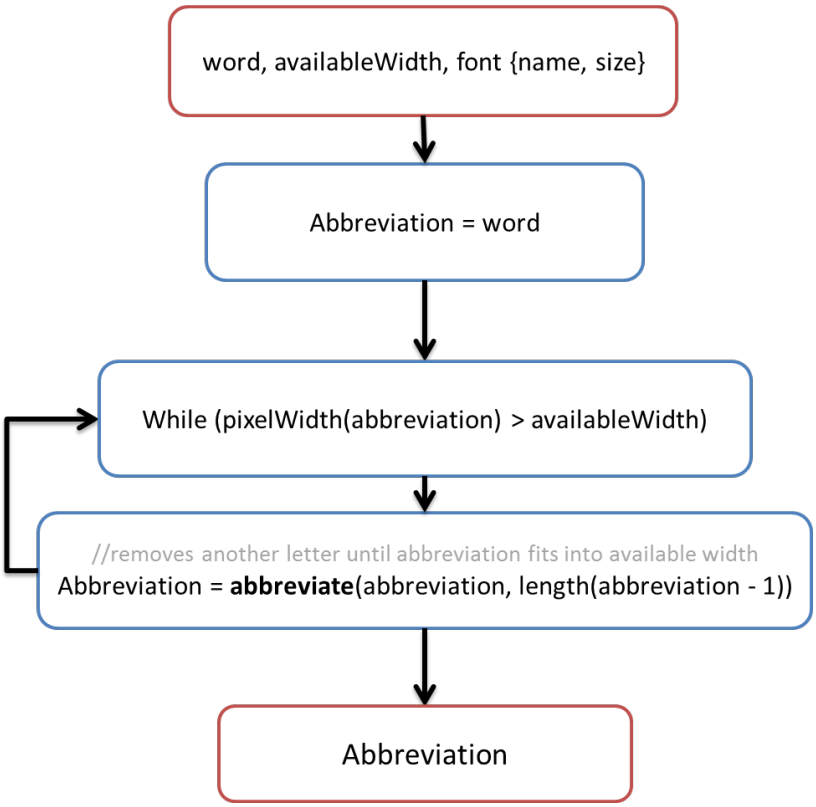

Fig. 5.4 The "abbreviation on demand" algorithm for a given *word*, constrained by screen size in pixels. Information about typeface and font size would be used in this case.

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

Fig. 5.5 Sample of 10 words from our study database. All abbreviations but, 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.

5.2 Application

In order to have our algorithm available, we have implemented a simple web interface (see Figure 5.6) where you can type in a word and the size of the desired abbreviation. We output the abbreviation based on the "Abbreviation on Demand" algorithm.

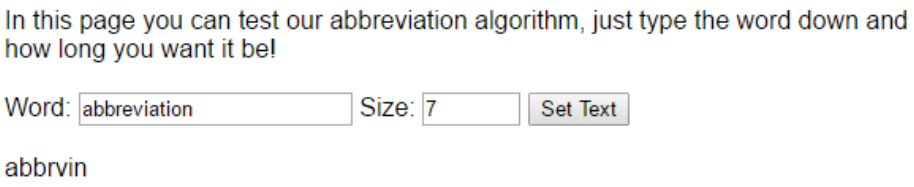

Fig. 5.6 Screenshot of prototype for abbreviating inputted words.

We have also implemented a variation of our algorithm to abbreviate words based on the available screen space screen (see Figure 5.7). In this prototype we can compare in real time how each of the algorithms (font size manipulation, abbreviation on demand, drop vowels, truncation and truncation while keeping the end) react to having more or less space on screen when we resize the text box that contains word.

In Figure 5.8 we can see the algorithm applied to a treemap visualization in D3 using our study words as the data set [7]. In Figure 5.9 we can see a closer comparison of the behaviour of our algorithm when the treemap gets rescaled.

Finally, we have an example of abbreviating a tweet in Figure 5.10. In this case we decided to analyze word by word, and cut one letter from each of the longest words until we can fit the message in 140 characters.

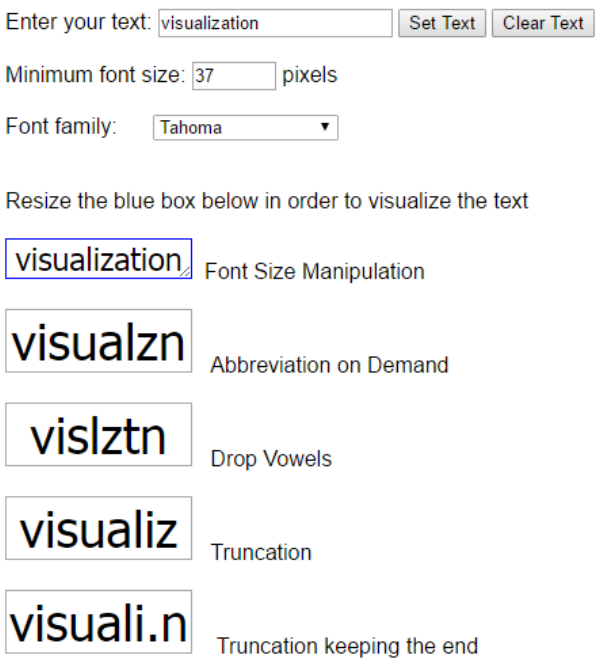

Fig. 5.7 Screenshot of resizable prototype.

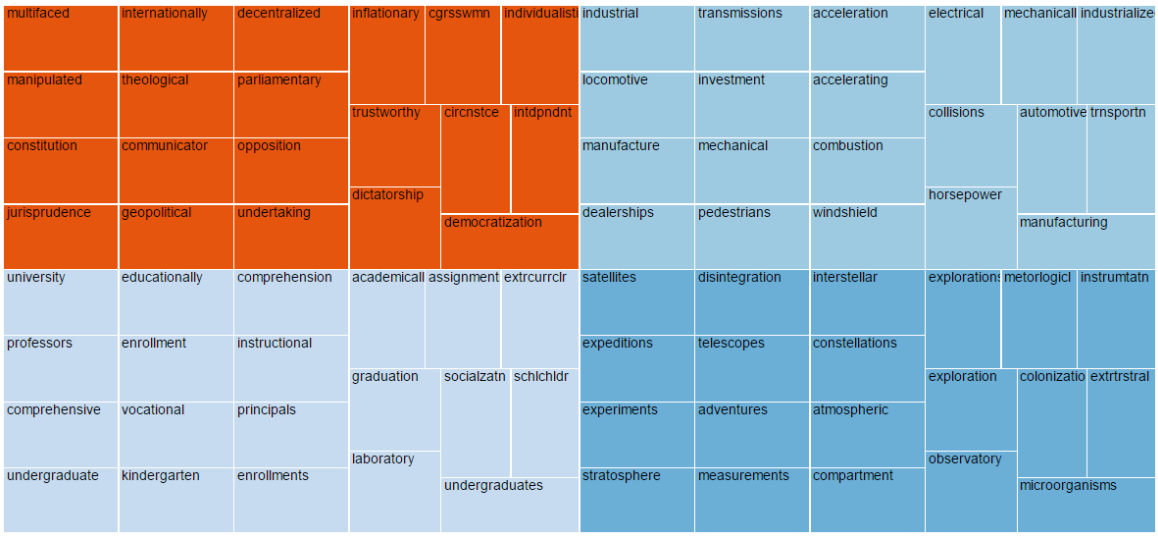

Fig. 5.8 Screenshot of treemap D3 visualization prototype.

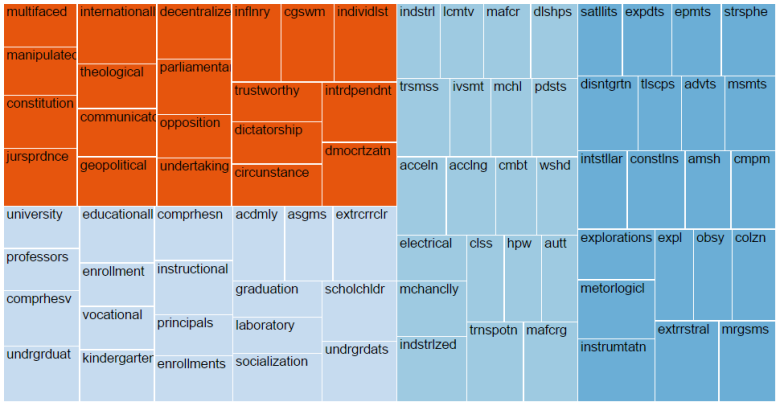

Fig. 5.9 Screenshot of rescaled treemap visualization.

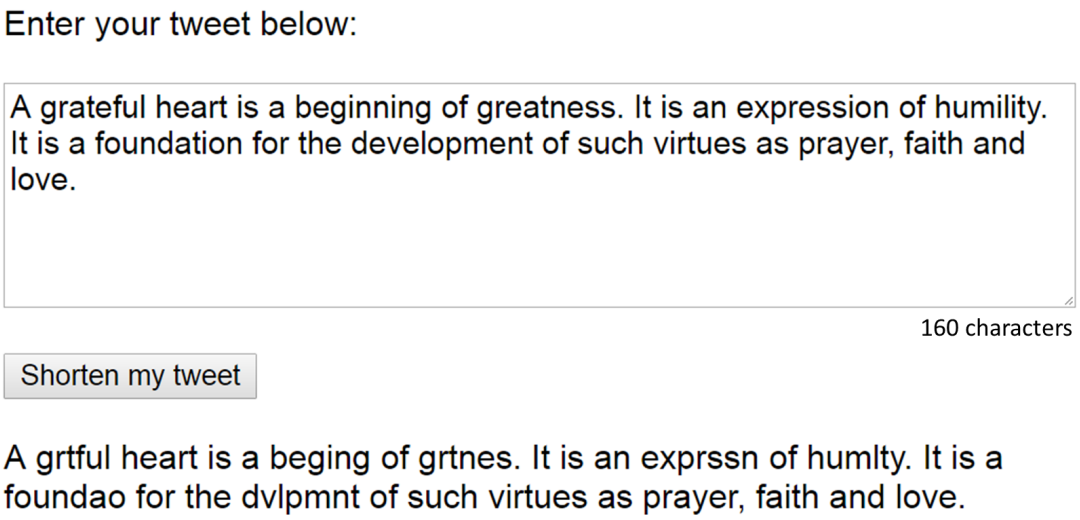

Fig. 5.10 Screenshot of prototype for abbreviating tweets into 140 characters. Original tweet had 150 characters.

Chapter 6

Conclusion

In this chapter we discuss our contributions, work limitations, conclusions and some ideas for future work related to this project.

6.1 Contributions

One of the contributions is our study methodology, that innovates by combining human cognitive power from a crowdsourced platform with a recommendation algorithm to maximize the relevance of the collected data. The study automatically adapts to the incoming data, so the specific tasks presented to users evolve over time. This design allowed us to collect all the necessary data in the same experiment. On the other hand, traditional approaches would lead us in two different experiments: an encoding study followed by data analysis and manual selection of relevant abbreviations to be then tested in a decoding study.

Furthermore, the results of our study added to the existing literature that tries to understand the process of human creating and understanding word abbreviations. We have provided data on the dropping probability of letters in relation to monographs, digraphs and position within a word.

Finally, we designed and implemented an abbreviation algorithm "Abbreviation on Demand" based on the probability of dropping letters regarding the letter itself or the character position. The "Abbreviation on Demand" algorithm works by dropping letters as needed in order to shorten words while maintaining their readability. We also demonstrated that in Chapter 5 that this algorithm can be easily adapted to different use cases, such as abbreviate based on screen size in pixels and abbreviation of short phrases like tweets.

6.2 Limitations

For our study design, we decided to use a database of words that do not repeat for the same participant, but that are used for every participant interpolating the order it appears in order to balance any effect for the used words. For this reason we were only able to use a very limited amount of words considering the size of the database affects directly on the study length. This database size decision gave us a generous amount of data on a small set of words, which showed to be enough to help us to create an algorithm to abbreviate words based on this data. However, we understand that if we had had more words our algorithm could perform better. Beyond that, in order to create a machine learning model to abbreviate any English word we would need many more different words abbreviated rather than having many abbreviations for a word.

Another limitation is that our algorithm creates abbreviations based on dropping the least relevant letters, and when compared against traditional known human created abbreviations, the latter would be more readable. We also do not consider standard abbreviations that are also highly readable.

For our study, we ideally wanted to recruit only English native speakers so that language skills would not affect on the study results. However, CrowdFlower does not have a screening option for language proficiency. Even though we explicitly wrote in our study description and consent form the crowd-worker needed to be native English speaker, and only accepted workers located in countries where English is the official first language, we still can not guarantee their English proficiency. Another limitation regarding language is that all the data collected and the algorithm designed is for the English language and can not be applied to other languages.

Crowdsourced studies in general offer a trade-off between time/amount of data and control. This trade-off also applies to our study we opted for the crowdsourced design in order to be able to collect higher volume of data. On the other hand, we were not able to control the experiment very well, we can not guarantee that participants did not use any other resources (e.g., search engines, forums, etc) when trying to decode or encode abbreviations. We also couldn't analyze the time of completion of the tasks considering crowd-workers often pause and resume tasks independently of what the task is.

Another limitation of this work is the fact that we did not run a formal evaluation of the abbreviations' created from our algorithm. This is a step for future work.

6.3 Future Work

As future work, we would like to collaborate with typography designers to design a data driven typeface that are more suitable for abbreviations.

We will also run a larger user study to collect many more word abbreviations, thus giving us the data needed to improve our algorithm and make it more powerful and generalizable. In the scope of this work, we could only investigate up to dropping probability of digraphs, however, with more data we can expand the findings for trigraphs, n-graphs, etc.

Collecting more data, gives us the possibility to create an artificial intelligence approach with machine learning, where we would train a model to learn how to abbreviate words.

In terms of exploring data, it might be interesting to run some data analysis on the encoded abbreviations and root of the words, to investigate the probability of keeping letters from the root chunk.

We could also consider making use of abbreviation dictionaries to try to optimize our algorithm. For instance, trying to fetch an abbreviation from the dictionary before creating a new one.

As future work, we could design a formal evaluation on readability comparing visualizations with and without abbreviated labels using our algorithm and the other common abbreviation techniques.

We can also adapt the "Abbreviation on Demand" algorithm, in order to create a library in JavaScript with functions that abbreviate based on different variables like number of letters, length in pixels, etc; and make it publicly available for other developers.

6.4 Conclusion

We can conclude that, the user study results lead to a promising new abbreviation algorithm. Even though we would want to have more data, the proposed solution showed to be a proof of concept that can easily be modularized and improved with the input of more data.

We strongly believe that our choice to design this experiment as an adaptive study gave us an advantage on the quality of data, considering that it is common to get a large amount of low quality data from crowdsourced studies due to lack of control on the experiment, the automatic filtering by our ranking algorithm guaranteed a good amount of trustworthy data by ignoring the noisy data. In addition, the data itself is an significant contribution for those who seek to understand word structure and letter relevance for readers.

In summary, this work described an opened question on how can we fit long words on space constrained screens, and through the design of an adaptive crowdsourced user study

we were to able to design and implement the algorithm “Abbreviation on Demand” that can abbreviate long words by dropping only as many letters as needed. We were also able to understand that context can improve people performance when trying to decode abbreviations, which is in favour to the use of abbreviations in text visualizations.

References

[1] Shehzad Afzal, Ross Maciejewski, Yun Jang, Niklas Elmqvist, and David S Ebert. Spatial text visualization using automatic typographic maps. *IEEE Transactions on Visualization and Computer Graphics*, (12):2556–2564, 2012.

[2] Andrew J Aschenbrenner, David A Balota, Alexandra J Weigand, Michele Scaltritti, and Derek Besner. The first letter position effect in visual word recognition: The role of spatial attention. *Journal of Experimental Psychology. Human Perception and Performance*, 2017.

[3] David A Balota and James I Chumbley. Are lexical decisions a good measure of lexical access? the role of word frequency in the neglected decision stage. *Journal of Experimental Psychology: Human Perception and Performance*, 10(3):340, 1984.

[4] Scott Bateman, Carl Gutwin, and Miguel Nacenta. Seeing things in the clouds: The effect of visual features on tag cloud selections. In *Proceedings of the Nineteenth ACM Conference on Hypertext and Hypermedia*, pages 193–202. ACM, 2008.

[5] Enrico Bertini, Maurizio Rigamonti, and Denis Lalanne. Extended excentric labeling. In *Computer Graphics Forum*, volume 28, pages 927–934. Wiley Online Library, 2009.

[6] Ionica Bizau. Npm: Levenshtein computing in javascript, 2016. URL https://www.npmjs.com/package/levdist. Accessed 2017-01-12.

[7] Mike Bostock. Treemap: D3 example, 2016. URL https://bl.ocks.org/mbostock/4063582. Accessed 2017-02-27.

[8] Richard Brath and Ebad Banissi. Using font attributes in knowledge maps and information retrieval. *Proceedings of Knowledge Maps and Information Retrieval (KMIR) at Digital Libraries*, 2014.

[9] Mark Davies. Word frequency data from the corpus of contemporary american english (COCA), 2011. URL http://www.wordfrequency.info. Accessed 2016-05-23.

[10] Rodolfo Delmonte. Visualizing poetry with SPARSAR–visual maps from poetic content. *on Computational Linguistics for Literature*, page 68, 2015.

[11] SL Ehrenreich and Theodora Porcu. Abbreviations for automated systems: Teaching operators the rules. *Directions in Human/Computer Interaction*, pages 111–135, 1982.

[12] Nathan Epstein. Npm: Kmeans clustering in javascript, 2016. URL https://www.npmjs.com/package/clusters. Accessed 2016-11-13.

[13] Jean-Daniel Fekete and Catherine Plaisant. Excentric labeling: dynamic neighborhood labeling for data visualization. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, pages 512–519. ACM, 1999.

[14] Pascal Goffin, Wesley Willett, Jean-Daniel Fekete, and Petra Isenberg. Exploring the placement and design of word-scale visualizations. *Visualization and Computer Graphics, IEEE Transactions on*, 20(12):2291–2300, 2014.

[15] Google. Google news pre-trained model, 2017. URL https://drive.google.com/file/d/0B7XkCwpI5KDYNlNUTTlSS21pQmM/edit. Accessed 2017-03-7.

[16] Jeffrey Heer and Michael Bostock. Crowdsourcing graphical perception: using mechanical turk to assess visualization design. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, pages 203–212. ACM, 2010.

[17] Wilbert Jan Heeringa. *Measuring dialect pronunciation differences using Levenshtein distance*. PhD thesis, Citeseer, 2004.

[18] Kathryn Hirsh-Pasek, S Nudelman, and ML Schneider. An experimental evaluation of abbreviation schemes in limited lexicons. *Behaviour and Information Technology*, 1(4): 359–369, 1982.

[19] Milton H Hodge and Florrie M Pennington. Some studies of word abbreviation behavior. *Journal of Experimental Psychology*, 98(2):350, 1973.

[20] Mengdie Hu, Krist Wongsuphasawat, and John Stasko. Visualizing social media content with sententree. *IEEE Transactions on Visualization and Computer Graphics*, 23(1): 621–630, 2017.

[21] Nathan Hurst, Wilmot Li, and Kim Marriott. Review of automatic document formatting. In *Proceedings of the 9th ACM Symposium on Document Engineering*, DocEng '09, pages 99–108. ACM, 2009.

[22] CrowdFlower Inc. Crowdflower and machine learning, 2016. URL https://www.crowdflower.com/machine-learning-with-crowdflower/. Accessed 2017-03-7.

[23] Julie A Jacko. *Human Computer Interaction Handbook: Fundamentals, Evolving Technologies, and Emerging Applications*. CRC press, 2012.

[24] Audun Jøsang, Roslan Ismail, and Colin Boyd. A survey of trust and reputation systems for online service provision. *Decision Support Systems*, 43(2):618–644, 2007.

[25] Apostolos Kritikopoulos, Martha Sideri, and Iraklis Varlamis. Wordrank: A method for ranking web pages based on content similarity. In *Databases, 2007. BNCOD'07. 24th British National Conference on*, pages 92–100. IEEE, 2007.

[26] LetterFrequency.com. Letterfrequency.com, 2016. URL http://letterfrequency.org/. Accessed 2017-04-17.

[27] STANDS4 LLC. Lagest resource of acronyms and abbreviations, 2017. URL http://www.abbreviations.com/. Accessed 2017-03-6.

[28] Saif M. Mohammad and Peter D. Turney. Crowdsourcing a word-emotion association lexicon. 29(3):436–465, 2013.

[29] FL Moses, CH Mendez, and SL Ehrenreich. The effect of learning and context on abbreviation intelligibility. In *88th Annual Convention of the American Psychological Association, Montreal, Canada*, 1980.

[30] Franklin L Moses and Lawrence M Potash. Assessment of abbreviation methods for automated tactical systems. Technical report, DTIC Document, 1979.

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

[32] Miguel Nacenta, Uta Hinrichs, and Sheelagh Carpendale. FatFonts: combining the symbolic and visual aspects of numbers. In *Proceedings of the International Working Conference on Advanced Visual Interfaces*, pages 407–414. ACM, 2012.

[33] L Nawrocki. Word abbreviations in man-computer communication systems. Technical report, ARI Working Paper MF-79-034, HF-74-04), Alexandria, VA: US Army Research Institute, 1979.

[34] Rudolf Netzel, Marcel Hlawatsch, Michael Burch, Sanjeev Balakrishnan, Hansjörg Schmauder, and Daniel Weiskopf. An evaluation of visual search support in maps. *IEEE Transactions on Visualization and Computer Graphics*, 23(1):421–430, 2017.

[35] Andrew Ng. Cs229 lecture notes. *CS229 Lecture notes*, 1(1):1–3, 2000.

[36] J.J. Oliver, W.L. Buntine, and G. Roumeliotis. System and method for adaptive text recommendation, jan 2005. URL https://www.google.com/patents/US6845374. US Patent 6,845,374.

[37] Lawrence Page, Sergey Brin, Rajeev Motwani, and Terry Winograd. The pagerank citation ranking: bringing order to the web. 1999.

[38] Sylvia Plath. General graded evaluation scale of Sylvia Plath's corpus, 2014. URL https://sparsar.wordpress.com/authors/sylvia-plath/. Accessed 2015-11-19.

[39] Anna W Rivadeneira, Daniel M Gruen, Michael J Muller, and David R Millen. Getting our head in the clouds: toward evaluation studies of tagclouds. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, pages 995–998. ACM, 2007.

[40] Timothy Samara. *Type Style Finder: The Busy Designer's Guide to Choosing Type*. Rockport Pub, 2006.

[41] Kate Sellen. SafeFont: typography design for the healthcare field. URL http://www.ocadu.ca/research/health-research/safe-font.htm. Accessed 2015-07-16.

[42] Katherine Sellen, Richard Hunt, and Greg Van Alstyne. Legibility to disambiguation: Typographic design strategies to prevent misreading, 2014. Poster at 2014 International Symposium on Human Factors and Ergonomics in Health Care: Leading The Way.

[43] Lynn A Streeter, JM Ackroff, and Glen A Taylor. Human factors and behavioral science: On abbreviating command names. *Bell System Technical Journal*, 62(6):1807–1826, 1983.

[44] Shauma Hayyu Syakura, Nur Ulfa Mauledevi, and Riama Maslan Sihombing. Typography expert system for graphic design purpose in the usage of typeface. *Jurnal Sarjana ITB bidang Teknik Elektro dan Informatika*, 1(1), 2012.

[45] New York Times. How the giants of finance shrank, then grew, under the financial crisis - interactive - nytimes.com, 2015. URL http://www.nytimes.com/interactive/2009/09/12/business/financial-markets-graphic.html. Accessed 2015-11-19.

[46] Top3. Github: Word2vec api, 2016. URL https://github.com/3Top/word2vec-api. Accessed 2017-03-7.

[47] Rafael Veras and Christopher Collins. Prioritizing nodes in hierarchical visualizations with the Tree Cut Model, 2014. Poster at Proc. of IEEE Conference on Information Visualization (InfoVis).

[48] Fernanda B Viegas, Martin Wattenberg, and Jonathan Feinberg. Participatory visualization with Wordle. *Visualization and Computer Graphics, IEEE Transactions on*, 15(6): 1137–1144, 2009.

[49] Martin Wattenberg and Fernanda B Viégas. The word tree, an interactive visual concordance. *Visualization and Computer Graphics, IEEE Transactions on*, 14(6): 1221–1228, 2008.

[50] Doug Wightman. Crowdsourcing human-based computation. In *Proceedings of the 6th Nordic Conference on Human-Computer Interaction: Extending Boundaries*, pages 551–560. ACM, 2010.

[51] Jacob O Wobbrock, Meredith Ringel Morris, and Andrew D Wilson. User-defined gestures for surface computing. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, pages 1083–1092. ACM, 2009.

[52] Pak Chung Wong, Patrick Mackey, Ken Perrine, James Eagan, Harlan Foote, and Jim Thomas. Dynamic visualization of graphs with extended labels. In *Information Visualization, 2005. INFOVIS 2005. IEEE Symposium on*, pages 73–80. IEEE, 2005.

[53] Melvin J Yap and David A Balota. Visual word recognition. In *The Oxford Handbook of Reading*, page 26. Oxford University Press, USA, 2015.

[54] Hermann Zapf. About micro-typography and the hz-program. *Electronic Publishing*, 6 (3):283–288, 1993.

[55] Yuxiang Zhao and Qinghua Zhu. Evaluation on crowdsourcing research: Current status and future direction. *Information Systems Frontiers*, 16(3):417–434, 2014.

Appendix A

List of Words extracted from COCA

Table A.1 List of words used for experiment (Group A and Group B)

Group	Context	Word
Group A	astronaut	extraterrestrial
Group A	astronaut	constellations
Group A	astronaut	colonization
Group A	astronaut	observatory
Group A	astronaut	exploration
Group A	automobile	transportation
Group A	automobile	accelerating
Group A	automobile	automotive
Group A	automobile	horsepower
Group A	automobile	collisions
Group A	classroom	schoochildren
Group A	classroom	instructional
Group A	classroom	socialization
Group A	classroom	laboratory
Group A	classroom	graduation
Group A	politically	internationally
Group A	politically	jurisprudence
Group A	politically	constitution
Group A	politically	manipulation
Group A	politically	multifaceted
Group B	astronaut	instrumentation
Group B	astronaut	meteorological
Group B	astronaut	explorations
Group B	astronaut	compartment
Group B	astronaut	atmospheric
Group B	automobile	industrialized
Group B	automobile	mechanically
Group B	automobile	electrical
Group B	automobile	windshield
Group B	automobile	combustion
Group B	classroom	extracurricular
Group B	classroom	assignments
Group B	classroom	academically
Group B	classroom	enrollments
Group B	classroom	principals
Group B	politically	democratization
Group B	politically	decentralized
Group B	politically	geopolitical
Group B	politically	communicator
Group B	politically	theological

Table A.2 List of words used for experiment (Group C and Group D)

Group	Context	Word
Group C	astronaut	microorganisms
Group C	astronaut	interstellar
Group C	astronaut	measurements
Group C	astronaut	adventurers
Group C	astronaut	telescopes
Group C	automobile	manufacturing
Group C	automobile	acceleration
Group C	automobile	pedestrians
Group C	automobile	mechanical
Group C	automobile	investment
Group C	classroom	undergraduates
Group C	classroom	comprehension
Group C	classroom	kindergarten
Group C	classroom	vocational
Group C	classroom	enrollment
Group C	politically	individualistic
Group C	politically	congresswoman
Group C	politically	inflationary
Group C	politically	undertaking
Group C	politically	opposition
Group D	astronaut	disintegration
Group D	astronaut	stratosphere
Group D	astronaut	experiments
Group D	astronaut	expeditions
Group D	astronaut	satellites
Group D	automobile	transmissions
Group D	automobile	dealerships
Group D	automobile	manufacture
Group D	automobile	locomotive
Group D	automobile	industrial
Group D	classroom	educationally
Group D	classroom	undergraduate
Group D	classroom	comprehensive
Group D	classroom	professors
Group D	classroom	university
Group D	politically	interdependent
Group D	politically	parliamentary
Group D	politically	circumstance
Group D	politically	dictatorship
Group D	politically	trustworthy

Appendix B

Comparison between Abbreviation on Demand Algorithm and Literature Techniques

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
combustion	**cmbust**	cmbstn	combus	combu.n	combstn	cmbstn	combtn	10	6
communicator	**communr**	cmmnctr	communi	commun.r	cmmnctr	com	comcator	12	7
compartment	**cmprtmt**	cmprtmn	compart	compar.t	compart	cmprtmn	cmprtmnt	11	7
comprehension	**comprhes**	cmprhnsn	comprehe	compreh.n	compreh	comp	cmprhns	13	8
comprehensive	**comprhsv**	cmprhnsv	comprehe	compreh.e	compreh	comp		13	8
congresswoman	**cgrsswmn**	cngrsswm	congress	congres.n	cngrssw	congrsswomn	congres	13	8
constellations	**costlans**	cnstlltn	constell	constel.s	constel	constellat	constlations	14	8
constitution	**costttn**	cnstttn	constit	consti.n	constit	constitution	contution	12	7
dealerships	**dalshps**	dlrshps	dealers	dealer.s	dlrshps	dealrshps	dlrships	11	7
decentralized	**dctrlzed**	dcntrlzd	decentra	decentr.d	dcntrlz	decentr	decenlized	13	8
democratization	**dmocrzatn**	demcrtztn	democrati	democrat.n	democra	democ	dmcrtzt	15	9
dictatorship	**dictshp**	dcttrsh	dictato	dictat.p	dictato	dcttrsh	dicttrship	12	7
disintegration	**disntgrn**	dsntgrtn	disinteg	disinte.n	disinte	disint	dsntgrt	14	8
educationally	**edctnaly**	edctnlly	educatio	educati.y	edu	educati	edctnll	13	8
electrical	**elcrcl**	elctrc	electr	elect.l	elec	elecl	electri	10	6
enrollment	**enrlmt**	enrllm	enroll	enrol.t	enrllmn	enrollm	enrllmnt	10	6
enrollments	**enrlmts**	enrllmn	enrollm	enroll.s	enrolmts	enrlmnts	enrllmnts	11	7
expeditions	**expdits**	expdtns	expedit	expedi.s	expedit	exped	expdtns	11	7
experiments	**expemts**	exprmnt	experim	experi.s	exper	exprmnt	experim	11	7
exploration	**explrtn**	explrtn	explora	explor.n	explrtn	explora	explr	11	7
explorations	**explrtns**	explrtn	explora	explor.s	explora	explorat	explrtn	12	7
extracurricular	**extrcrlar**	extrcrrcl	extracurr	extracur.r	extrcrr	xtracular	xtrcrriculr	15	9
extraterrestrial	**extrrstral**	extrtrrstr	extraterre	extraterr.l	et	extrtrr	extrate	16	10
geopolitical	**gepoltl**	gepltcl	geopoli	geopol.l	geopoli	geopol	gepltcl	12	7
graduation	**grdatn**	gradtn	gradua	gradu.n	gradutn	grad	gradtn	10	6
horsepower	**hspowe**	hrspwr	horsep	horse.r	hp	hspwr	horspwr	10	6
individualistic	**idvidlstc**	indvdlstc	individua	individu.c	individ	individual	indvdstc	15	9
industrial	**indstl**	indstr	indust	indus.l	indstrl	indtrial	indust	10	6
industrialized	**indstzed**	indstrlz	industri	industr.d	indstrlized	indstrl	industri	14	8
inflationary	**inflary**	infltnr	inflati	inflat.y	inflationary	infl	infltnr	12	7

Fig. B.1 Table with comparison of Abbreviation on Demand algorithm against the literature techniques.

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
instructional	**instrctl**	instrctn	instruct	instruc.l	nstrucnal	instrctnl	instructi	13	8
instrumentation	**instrumtn**	instrmntt	instrumen	instrume.n	instrmntation	instrmntatn	instrument	15	9
interdependent	**intdpdnt**	intrdpnd	interdep	interde.t	intrdpn	intrdpndnt	interde	14	8
internationally	**intnanaly**	intrntnll	internati	internat.y	intnly	intrntn	interna	15	9
interstellar	**inttlar**	intrstl	interst	inters.r	intrstl	instlr	intrstllar	12	7
investment	**ivstmt**	invstm	invest	inves.t	invstmn	invstmnt	invstment	10	6
jurisprudence	**jursprdc**	jrsprdnc	jurispru	jurispr.e	jurispr	jrsprdn	jurisdence	13	8
kindergarten	**kidgart**	kndrgrt	kinderg	kinder.n	kinderg	kinder	kndergrtn	12	7
laboratory	**labory**	lbrtry	labora	labor.y	lab	laborat	labrtory	10	6
locomotive	**lcmtve**	locmtv	locomo	locom.e	loco	locomtv	lcmtv	10	6
manipulation	**manpuln**	manpltn	manipul	manipu.n	manip	manipul	manpultion	12	7
manufacture	**manufcr**	manfctr	manufac	manufa.e	manufac	manfctr	manu	11	7
manufacturing	**manufcrg**	mnfctrng	manufact	manufac.g	mnfctrn	mftg	manufac	13	8
measurements	**msurmts**	msrmnts	measure	measur.s	msrmnts	measrmnts		12	7
mechanical	**mchacl**	mchncl	mechan	mecha.l	mech	mec	mechanica	10	6
mechanically	**mchacly**	mchncll	mechani	mechan.y	mech	mechani	mechally	12	7
meteorological	**mtolgicl**	metrlgcl	meteorol	meteoro.l	mtrlgcl	meteological	meteorlgcl	14	8
microorganisms	**mirgasms**	mcrrgnsm	microorg	microor.s	microorgs	mcrrgns	microor	14	8
multifaceted	**multftd**	mltfctd	multifa	multif.d	mfaceted	mltfctd	multfactd	12	7
observatory	**obsvary**	obsrvtr	observa	observ.y	obsrvtry	obsrvtr	obstory	11	7
opposition	**oppost**	oppstn	opposi	oppos.n	oppostn	opp	oppstion	10	6
parliamentary	**parlmtry**	prlmntry	parliame	parliam.y	prlmntr	parliam	parlmt	13	8
pedestrians	**pedstrs**	pdstrns	pedestr	pedest.s	pdstrians	pdstrns	peds	11	7
principals	**prcpls**	prncpl	princi	princ.s	prncpls	princip	prin	10	6
professors	**prfsso**	prfssr	profes	profe.s	profs	prof	profess	10	6
satellites	**satlts**	stllts	satell	satel.s	satllts	stlts	sate	10	6
schoolchildren	**schchldr**	schlchld	schoolch	schoolc.n	shlchilden	schlchl	schchdrn	14	8
socialization	**socilzan**	socilztn	socializ	sociali.n	sociali	socialztn	soclztn	13	8
stratosphere	**strsphe**	strtsph	stratos	strato.e	stratos	strato	strat	12	7
telescopes	**tlscps**	tlscps	telesc	teles.s	telscps	telscopes	tlscpes	10	6
theological	**theolgl**	thelgcl	theolog	theolo.l	thelgcl	thelgicl	theolog	11	7
transmissions	**trnsmsss**	trnsmssn	transmis	transmi.s	transmi	trnsmss	transmisses	13	8
transportation	**trnspotn**	trnsprtt	transpor	transpo.n	transp	transpo	trans	14	8
trustworthy	**trswthy**	trstwrt	trustwo	trustw.y	trstwrthy	trstwrt	trstwrty	11	7
undergraduate	**undgrdat**	undrgrdt	undergra	undergr.e	undergrad	undrgrd	ndrgrad	13	8
undergraduates	**undgrdas**	undrgrdt	undergra	undergr.s	undergrad	undergrads	undrgraduats	14	8
undertaking	**undtkig**	undrtkn	underta	undert.g	undrtkn	undrtaking	underta	11	7
university	**unvsty**	unvrst	univer	unive.y	uni	univst	unvrsty	10	6
vocational	**vocnal**	voctnl	vocati	vocat.l	voca	vocatio	voc	10	6
windshield	**wdshld**	wndshl	windsh	winds.d	wndshld	wndshild	wdshld	10	6

Fig. B.2 (Cont.) Table with comparison of Abbreviation on Demand algorithm against the literature techniques.